МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) __Факультет автоматизації і комп'ютерних систем__
Кафедра _Інформаційних технологій, штучного інтелекту і кібербезпеки_

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту (декан факультету)	Завідувач кафедри
_____Андрій_ФОРСЮК_____	_____Сергій ГРИБКОВ_____
(підпис) (прізвище та ініціали)	(підпис) (прізвище та ініціали)
«___» _____ 20__ р.	«___» _____ 20__ р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності _____122 «Комп'ютерні науки»_____
(код та назва спеціальності)

освітньо-професійної програми____Комп'ютерні науки_____

на тему: Створення інформаційної системи підтримки обліку складу готової продукції ТДВ "Яготинський маслозавод"

Виконав: здобувач 4 курсу, групи КН-4-2.

Харченко Богдан Анатолійович _____
(прізвище, ім'я та по батькові повністю) (підпис)

Керівник ст.викл. Грама Михайло Петрович _____
(прізвище, ім'я та по батькові повністю) (підпис)

Консультанти _____ _____
(прізвище та ініціали) (підпис)

_____ _____
(прізвище та ініціали) (підпис)

_____ _____
(прізвище та ініціали) (підпис)

Рецензент _____ _____
(прізвище та ініціали) (підпис)

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

Здобувач_____
(підпис)

Київ - 2024

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інформаційних технологій, штучного інтелекту і кібербезпеки
Сергій ГРИБКОВ

" 15 " квітня 2024 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА
Харченко Богдан Анатолійович
(прізвище, ім'я, по батькові)

1. Тема роботи Створення інформаційної системи підтримки обліку складу готової продукції ТДВ "Яготинський маслозавод"

Керівник роботи ст.викл Грама Михайло Петрович PhD
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від 15 квітня 2024 року № 279-кс
2. Строк подання здобувачем роботи 03.06.2024 р.
3. Вихідні дані до роботи
Організація роботи відділу складу готової продукції, задачі відділу
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1. Розробка бази даних mySQL в середовищі phpMyAdmin
2. Проектування інтерфейсу користувача в середовищі Mivrosoft Visual Studio 2022

5. Перелік графічного матеріалу
6 таблиць, 23 ілюстрації, 4 додатки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Ст.викл. Грама М.П. PhD	19.02.2024	20.02.2024
2	Ст.викл. Грама М.П. PhD	26.03.2024	28.03.2024
3	Ст.викл. Грама М.П. PhD	05.04.2024	06.04.2024
4	Ст.викл. Грама М.П. PhD	29.04.2024	02.05.2024

7. Дата видачі завдання_____15 квітня 2024 року_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Системний аналіз процесу	20.02.2024	
2	Розробка технічного завдання	28.03.2024	
3	Реалізація задач інформатизації процесу діяльності складського відділу підприємства	06.04.2024	
4	Оформлення пояснювальної записки	01.05.2024	
5	Створення презентації	18.05.2024	

Здобувач _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему створення інформаційної системи підтримки обліку складу готової продукції ТДВ “Яготинський маслозавод” складається з 54 сторінок, 6 таблиць, 23 рисунків, 4 додатків, 29 літературних джерел.

В даній роботі наведено результати досліджень роботи підприємства ТДВ “Яготинський маслозавод” та самого об’єкта дослідження, а саме відділу складу готової продукції.

Графічний інтерфейс користувача було реалізовано в середовищі Visual Studio 2022, мовою C#, набором бібліотек Windows Forms.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, ВІДДІЛ СКЛАДУ, БАЗА ДАНИХ, ГРАФІЧНИЙ ІНТЕРФЕЙС, ТДВ “Яготинський маслозавод”.

ANNOTATION

The qualification work on the topic of creating an information system for accounting for the compliance of finished products of TDV "Yagotinsky Maslozavod" consists of 54 pages, 6 the table, 23 drawings, 4 appendices, 29 literary sources.

This paper presents the results of the research of the TDV Yagotynskyi Maslozavod enterprise and the research object itself, namely the department of conformity of finished products.

The graphical user interface was implemented in the Visual Studio 2022 environment, in the C# language, with a set of Windows Forms libraries.

KEY WORDS: INFORMATION SYSTEM, WAREHOUSE DEPARTMENT, DATA BASE, GRAPHIC INTERFACE, TDV "Yagotinsky Maslozavod".

ЗМІСТ

ВСТУП

Продуктивність будь-якого виробничого підприємства визначається його автоматизованістю як в цілому, так і окремо його відділів. Автоматизація підприємства впливає на ефективність, зниження витрат на працю, підвищення якості та дає мінімізувати людський фактор.

На сьогоднішній день є велика кількість засобів для автоматизування тих, чи інших відділів, як то від обліку, так і до самого процесу виробництва.

Для коректного та швидкого управління відділом складу та обліком готової продукції необхідна автоматизована система його діяльності.

Основні задачі, які має покривати автоматизована система обліку складу готової продукції, це:

- облік готової продукції;
- створення заявок на відвантаження;
- облік відвантажень;
- облік персоналу відвантажень;
- звітність.

Метою даної кваліфікаційної роботи є проведення дослідження роботи складу з готової продукції, розробка інформаційної підтримки для даного відділу та її інтеграція.

Завданням кваліфікаційної роботи є: розробка інформаційної системи для підтримки роботи складського відділу.

РОЗДІЛ 1. СИСТЕМНИЙ АНАЛІЗ ОБ'ЄКТА ДОСЛІДЖЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ ДО ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

1.1. Загальна характеристика

Яготинський маслоробний завод був введений в експлуатацію в червні 1956 року, ставши одним з найбільших молокопереробних підприємств в області. У 1960-ті роки завод був реконструйований, оснащений новим обладнанням і переведений на природній газ що дозволило збільшити його переробні потужності до 100 т молока на добу [31]. В цілому, за радянських часів, маслозавод входив до числа найбільших підприємств міста.

У лютому 1994 року перетворене у відкрите акціонерне товариство "Яготинський маслозавод". Наразі акціонерами є фізичні та юридичні особи, які набули право власності на акції товариства у процесі приватизації, на вторинному ринку цінних паперів, а також у порядку спадкування громадян, правонаступництва юридичних осіб та у інших випадках, передбачених чинним законодавством. Держава акціями даного товариства не володіє [10].

На офіційному сайті підприємства наведено таку інформацію про асортимент продукції:

- молочні коктейлі;
- масло безлактозне;
- какао на молоці;
- йогурти;
- сухе молоко;
- безлактозне молоко;
- паста сиркова;
- кефір;

Та ще багато інших видів продукції. Загалом виробництво налічує приблизно 36 найменувань продукції.

З цього можна дізнатись, що дане виробництво є досить великим та важливим для пересічного споживача. Також це підприємство займає велику нішу на ринку молочних товарів.

1.2. Організаційна структура підприємства

Організаційна структура підприємства складається з переліку відділів, які відповідають за функціонування всього підприємства. На чолі кожного відділу є керівник, який і управляє цими відділами.

Організаційна структура даного підприємства представлена на рисунку нижче (див. Рисунок 1.1):

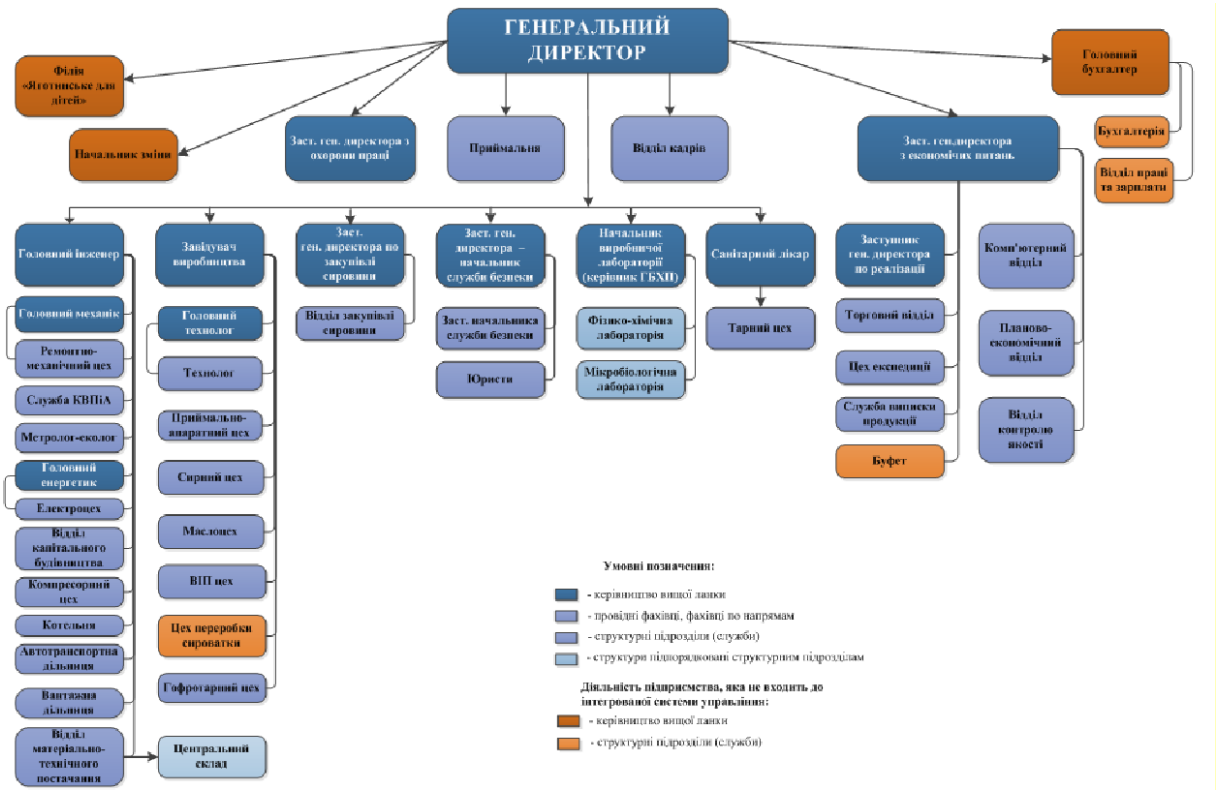

Рисунок 1.1 – Організаційна структура підприємства

Кожен відділ підприємства є важливим для підтримки його діяльності. Автоматизація та інформатизація цих відділів напряму впливає на продуктивність та ефективність всього підприємства.

Слід виділити складський відділ, який відіграє не останню роль на підприємстві. Його правильне та швидке функціонування може зберегти багато матеріальних та часових ресурсів.

Організаційна структура складскього відділу готової продукції (див. Рисунок 1.2):

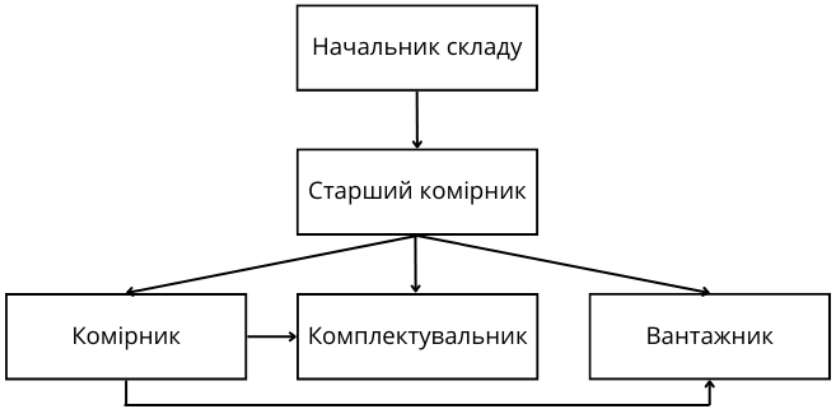

Рисунок 1.2 - Організаційна структура складського відділу

Аналізуючи організаційну структуру складського відділу, можна визначити функції того, чи іншого співробітника.

Начальник складського відділу виконує такі функції:

- керує роботою складу;
- відповідає за організацію роботи персоналу;
- вирішує технічні та організаційні питання;
- слідкує за надходженням та відвантаженням продукції.

Старший комірник:

- координує роботу комірників;
- відповідає за організацію та контроль правильного зберігання;
- здійснює контроль за веденням обліку продукції на складі.

Комірник:

- приймає надходження на склад;
- здійснює відвантаження відправок відповідно до заявок.

Комплектувальник:

- збирає товари зі складу відповідно до створених заявок;
- комплектує створену заявку продукцією, відповідно якості та кількості.

Складський відділ готової продукції також взаємодіє з іншими відділами підприємства. (див Таблиця 1.1.):

Таблиця 1.1. Функціональні зв'язки складського відділу з іншими відділами

Функція складського відділу	**Взаємодія з іншими відділами**	**Опис взаємодії**
Прийом сировини та готової продукції	Відділ якості	Відділ якості здійснює контроль якості надходження
Управління запасами	Відділ планування, виробничий відділ	Відділ планування допомагає прогнозувати потреби у сировині. Виробничий відділ інформує про потреби сировини
Комплектація замовлень на відвантаження	Відділ продажів, відділ логістики	Відділ продажів надає замовлення для комплектації, а відділ логістики планує доставку готової продукції клієнтам.
Відвантаження продукції	Відділ логістики, відділ економічних питань	Відділ логістики координує відвантаження та доставку продукції. Відділ економічних питань займається документацією.
Забезпечення якості	Відділ якості	Відділ якості перевіряє дотримання стандартів якості та зберігання продукції на складі

1.3. Аналіз стану інформатизації складського відділу підприємства

На даний момент, для обліку готової продукції використовується електронний складський журнал Excel.

Дане рішення має перелік недоліків, які є незадовільними:

- Ручне заповнення даних (ручне заповнення даних вимагає багато часу та уваги до заповнюваної інформації);
- Високий ризик помилок (через людський фактор є великий шанс зробити помилку в обліку даних);
- Відсутність оновлення даних в режимі реального часу;
- Низькій рівень автоматизації (даний метод обліку є досить старим та не продуктивним);
- Ризик втрати даних, так як вся інформація зберігається одним файлом.

Зрештою, можна сказати, що даний метод обліку не відповідає жодним критеріям автоматизованого обліку.

Це є дуже критичним фактором для великих виробництв, через який втрачається продуктивність як складського відділу, так і всього підприємства в цілому.

Найголовнішою метою даної кваліфікаційної роботи розробити інформаційну систему підтримки, яка покращила б продуктивність та автоматизованість складського відділу, мінімізуючи наявні недоліки та додаючи зручний спектр функціоналу для обліку на даному підприємстві.

1.4. Розроблення функціональної моделі існуючих бізнес-процесів

Для виявлення основних бізне-процесів було розроблено функціональну модель “Діяльність відділу складу готової продукції” в онлайн середовищі для побудування IDEF0 моделі (див. Рисунок - А.1 у додатку А).

Для створеної моделі входами є:

- готова продукції;

- інформація про продукцію;
- працівники складу.

Виходами даної моделі є:

- створення відвантаження;
- звітність;
- продукція до зберігання.

Механізмами є:

- облік продукції;
- керівник.

Засоби управління:

- нормативні документи;
- приймання продукції;
- регламент роботи.

Описуючи побудовану діаграму, можна побачити, що облік продукції, створення відвантажень є важливим аспектом в діяльності відділу складу з готової продукції.

Після створення функціональної моделі було проведено її декомпозицію до першого рівня для більш точного огляду стану речей (див. Рисунок - А.2 у Додатку А). Створена декомпозиція поділяється на наступні блоки:

- приймання готової продукції;
- зберігання продукції;
- відвантаження продукції;
- облік і звітність.

Виходячи з побудованої декомпозиції першого рівня деталізації, слід зазначити, що засіб обліку відіграє не останню роль в діяльності складського відділу готової продукції. На даний момент часу, використовуваний обліковий засіб є досить не ефективним, через певні проблеми, такі як:

- проблема зі швидкодією інформації;
- проблема з орієнтацією в інформаційному просторі;
- можливість допускання помилок при обліку.

Задачею автоматизації засобу обліку готової продукції є мінімізування та вирішення даних проблем. А саме, створення інформаційної системи, що буде їх вирішувати. Розроблювана система вирішить такі проблеми, та надасть певні можливості:

- швидкість обліку;
- легкий пошук потрібної інформації;
- мінімізування своєрідних помилок в облікуванні;
- зручність в користуванні.

Через спосіб введення, та обробки даних, інформаційна система мінімізує допускання помилок в облікуванні. В свою чергу, виведена інформація є актуальною та чіткою з боку її перегляду та доступу в інформаційному просторі. Швидкодія даної системи впирається лише на продуктивність серверної частини.

1.5. Існуючі рішення для вирішення виявлених недоліків

На даний момент часу, існує велика кількість повноцінних готових систем під схожі задачі. Це можуть бути як професійні рішення саме для складу, так і рішення, в яких є певні необхідні функції. Розглянемо декілька з них:

1.5.1. IBS виробництво

Ця інформаційна система є спеціалізованою саме під виробництво. Через те, що в ній є багато версій (експрес, бізнес) її можна використовувати як для малого бізнесу, так і для великого виробництва. Воно забезпечує ефективне управління виробництвом, від обліку сировини до аналізу ефективності робочих ліній. Це програмне забезпечення надає можливість автоматизувати облік як готової продукції так і сировини [13].

Також слід перечислити основні функції цього ПЗ:

- детальний довідник готової продукції;

- автоматизація планування виробництва;
- аналітика та звітність;
- інтеграція з іншими системами;
- модульність;
- облік по партіях;
- повна інформація про рух продукції на будь-який момент часу;
- облік якості на кожну партію продуктів;
- можливість установити нормативні показники виходу готової продукції;
- списання та прихід готової продукції.

Як і всі програмні рішення, цей продукт має свої переваги та недоліки, тож почнемо з переваг:

- скорочення витрат часу на документально-інформаційне забезпечення бізнес-процесів до мінімуму;
- полегшення виявлення випадків крадіжок на підприємстві;
- володіння повною інформацією про стан справ підприємства;
- гнучкість налаштування.

Недоліки:

- висока початкова вартість як для великого підприємства;
- процес впровадження може бути складним, зрівнюючи з спеціалізованими рішеннями під певне підприємство;
- потреба в навчанні персоналу;
- необхідність регулярного оновлення.

1.5.2. Вигляд інтерфейсу IBS Виробництво

Інтерфейс програмного забезпечення IBS Виробництво розроблений з урахуванням потреб користувачів, які вимагають глибокої інтеграції та контролю над виробничими процесами..Також даний інтерфейс може виявитись дещо

складним для опанування. Далі проаналізовано приклад цього інтерфейсу в середині самого середовища (див Рисунок 1.3).

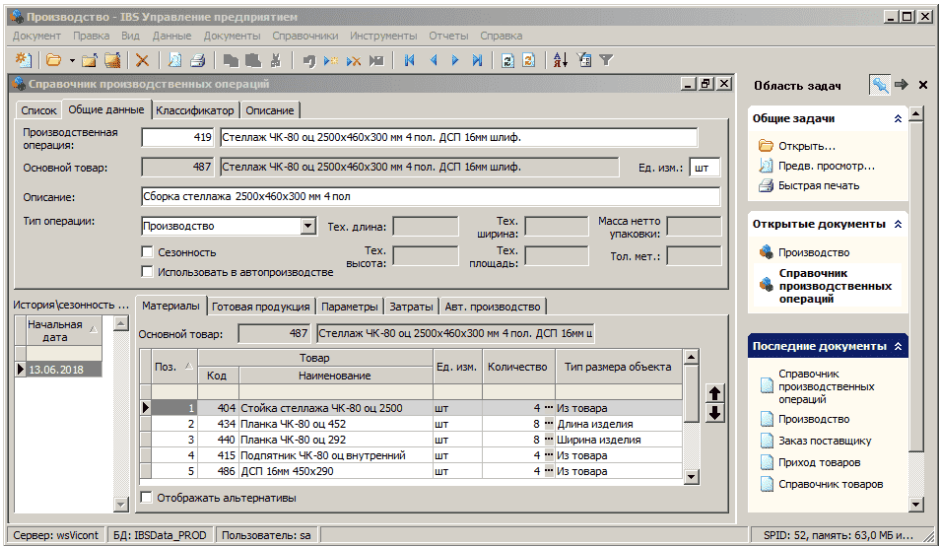

Рисунок.1.3 - Приклад інтерфейсу IBS Виробництво

Як можна побачити із скріншоту вище, програма включає в себе ряд складних елементів управління, таких як:

- меню навігації;
- панель інструментів;
- робоча область;
- інформаційні панелі.

Цей інтерфейс демонструє високий рівень гнучкості та налаштувань, дозволяючи користувачам адаптувати робоче середовище під свої індивідуальні потреби.

Водночас, складність інтерфейсу може потребувати додаткового часу на навчання нових користувачів, щоб вони могли ефективно використовувати всі можливості цього програмного забезпечення. Однак, інвестиції в час та в навчання працівників, як правило, швидко окупаються за рахунок зростання продуктивності та ефективності робочих процесів.

1.5.3. Oracle Warehouse Management Studio (WMS)

Oracle Warehouse Management Studio (WMS) є частиною Oracle Supply Chain Management (SCM) Cloud і представляє собою інтегроване рішення для управління складськими операціями. Воно дозволяє підприємству оптимізувати більшість процесів від приймання продукції на склад до відвантаження кінцевому споживачу [12].

Основні функції даної системи:

- управління прийманням та відвантаженням продукції;
- відстеження запасів в реальному часі;
- оптимізація використання складського простору;
- оптимізація та управління трудових ресурсів на складі;
- звітність та аналітика.

Ця інформаційна система має свої переваги та недоліки, тож розглянемо їх, починаючи з переваг:

- тісна інтеграція з Oracle SCM, що забезпечує єдину платформу для управління всіма ланцюгами постачання;
- гнучкість налаштування;
- широкі можливості інтеграції;
- розширені аналітичні можливості;
- підтримка клієнтів та оновлення;
- ця система підтримує як малі так і великі підприємства;
- рішення дозволяє автоматизувати багато складських процесів, включаючи зберігання, приймання та відвантаження продукції.
- програма допомагає ефективніше розподіляти складський простір;
- забезпечення точних запасів;
- висока кількість функцій.

Недоліки:

- висока вартість впровадження;

- складність конфігурації (велика кількість функцій може вимагати значних зусиль для налаштування та оптимізації під конкретні потреби);
- велика кількість підтримуваних функцій також призводить до складності впровадження та налаштування цієї системи;
- потрібність кваліфікованого навчання для використання цієї системи.

1.5.4. Вигляд інтерфейсу Oracle WMS

Інтерфейс Oracle WMS розроблено з урахуванням зручності користувача, надаючи легкий доступ до всіх необхідних функцій та інструментів управління складом. В даній системі інтерфейс виглядає не перенавантаженим як в минулій програмі, але для його опанування потрібні певні професійні здібності.

Інтерфейс цього середовища включає в себе такі функції:

- навігація;
- панель інструментів;
- індивідуальне налаштування;
- інтерактивні звіти та аналітика.

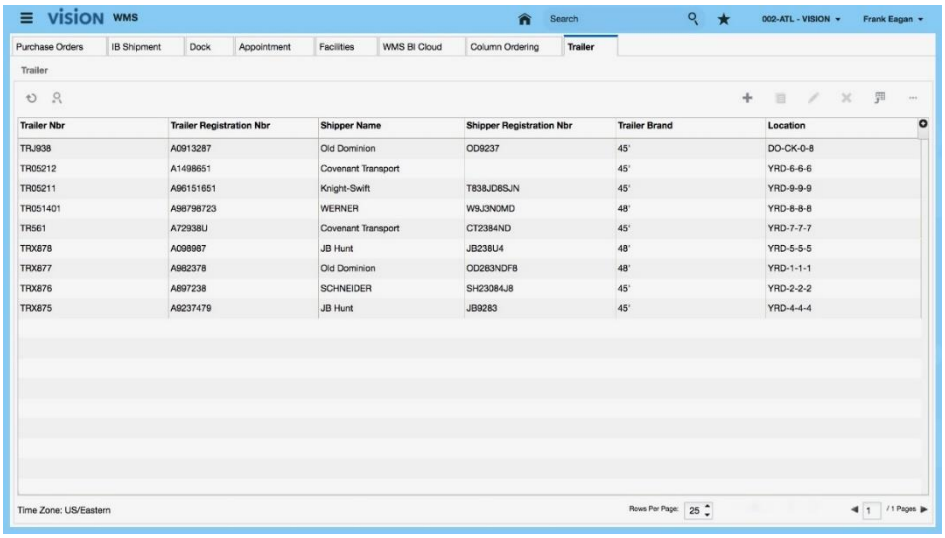

Рисунок 1.4 - Вигляд інтерфейсу Oracle WMS

1.5.5. Порівняльна характеристика систем для управління проектами

Аналізуючи перелічені вище програмні засоби, а саме “IBS Виробництво” та “Oracle WMS Cloud”, було створено порівняльну таблицю цих двох продуктів (див Таблиця 1.5.1).

Таблиця 1.2. Порівняльна таблиця програмних засобів

Функція	**Oracle WMS Cloud**	**IBS Виробництво**
Технічні вимоги	Вимагає високої продуктивності, а саме великої кількості пам’яті	Дозволяє використовувати менш потужне обладнання
Гнучкість налаштування	Вимагає значних зусиль в налаштуванні	Висока гнучкість, яка залежить від плану, який використовується
Складність користування	Потребує підготовку персоналу, так як має високу складність	Середня складність системи
Вартість	25 000 доларів	До 30 000 гривень (залежить від вбудованих функцій), та +600 грн за нове робоче місце, оренда стартового пакету – 750 грн на місяць.
Інтерфейс	Зручний інтерфейс	Дещо складний інтерфейс, який можна налаштувати під користувача

Вибір між даними двома системами залежить від конкретних потреб підприємства, його розміру, галузі, чи спеціалізації. Побудована таблиця демонструє загальні відмінності між аналізованими системами. Обидві системи надають широкий спектр функцій, але їх основна спеціалізація та призначення дещо різниться:

- Oracle WMS Cloud більш націлений на управління та автоматизацію складським відділом.

- IBS Виробництво орієнтовано на виробництво та на інтеграцію з виробництвом.

Oracle WMS ідеально підходить для компаній, які шукають ефективне управління складом і логістикою, тоді як IBS Виробництво краще відповідає потребам виробничих підприємств, зосереджених на оптимізації виробничих процесів.

1.6. Обґрунтування доцільності створення інформаційної системи підтримки обліку складу готової продукції

За результатом огляду систем аналогів, можна підвести певні підсумки. Обидві системи можуть надати певні переваги, які будуть потрібні відділу складу, але ці системи мають великий спектр функцій, що можуть не знадобитись даному відділу.

Найголовнішою метою розробки є створення інформаційної системи, яка б мала всі основні необхідні функції. Дана система має бути простою, як в плані графічного інтерфейсу, що буде зручним в користуванні, так і в плані набору функціоналу, де користувач не буде плутатись в тих, чи інших опціях та можливостях.

Після інтеграції створюваної системи, отримаємо такі можливості:

- швидке редагування інформації щодо продукції, відвантажень, тощо;
- збільшення рівню автоматизації відділу в цілому;
- робота користувача системи та керівника полегшиться;
- матеріальні та часові витрати на інтеграцію зменшаться.

1.7. Економічний та соціальний ефект від створення та впровадження інформаційної системи підтримки обліку складу готової продукції

При створенні та впровадженні такого роду інформаційної системи, важливим фактором є економічний ефект, який може бути досягнутий. Цей ефект визначає на скільки розробка такої системи є доцільною.

Економічним ефектом після впровадження даної системи стане:

1. підвищення продуктивності та ефективності відділу складу в цілому;
2. підвищення ефективності управління готовою продукцією;
3. зменшення фінансових витрат через помилки в обліку;
4. ефективніше управління витратами сировини, через прозорий облік продукції.

Дані чинники збережуть багато матеріального ресурсу підприємства. Також слід зазначити, що дана інформаційна система також має в собі і соціальний ефект від впровадження. Чинниками соціального ефекту можна вважати:

1. покращення умов праці через автоматизацію процесу обліку;
2. оптимізація процесу обліку готової продукції зменшить навантаження на працівників;

Таким чином, з боку економічного та соціального ефекту ця система є доцільною до впровадження.

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ

2.1. Загальні положення.

2.1.1. Найменування системи: «Інформаційна система підтримки обліку складу готової продукції ТДВ “Яготинський маслозавод»

2.1.2. Результати робіт зі створення системи оформлюються згідно з вимогами ДСТУ на відповідні етапи розробки. Порядок оформлення і передачі результатів у даному випадку визначається змістом і календарним планом виконання розробки.

2.1.3. У випадку необхідності на наступних стадіях робіт по створенню системи окремі положення можуть уточнюватися та допрацьовуватись.

2.2. Призначення і цілі створення системи.

2.2.1. Призначення системи.

Система призначена для підтримки роботи складського відділу готової продукції підприємства. Система автоматизує відділ складу, формування даних про продукцію, відвантаження, створення звітів. Система має пошук по даним про продукцію, її найменування, дату виробництва, тощо. Також дані про відвантаження, та вся додаткова необхідна інформація.

2.2.2. Цілі створення системи.

Основною метою створення інформаційної системи є полегшення обліку готової продукції та її відвантаження, що забезпечить вищу продуктивність всього відділу в цілому.

2.3. Характеристика об’єкта інформатизації.

2.3.1. Короткі відомості про об’єкт автоматизації.

Об’єктом інформатизації є діяльність відділу складу з готової продукції підприємства “Яготинський маслозавод”. Базовий об’єкт впровадження – ТДВ “Яготинський маслозавод”.

2.4. Вимоги до системи.

2.4.1. Вимоги до системи в цілому.

- створення сучасної інформаційної системи, яка б забезпечила швидкий облік на підприємстві;
- система має бути надійною;
- повинна мати простий та інтуїтивний інтерфейс.

2.4.1.1. Вимоги до структури і функціонування системи.

- база даних має бути чіткою та зрозумілою;
- всі вікна інформаційної системи мають забезпечити зручну навігацію;
- мати можливість пошуку та фільтрації даних;
- створення звітів.

2.4.1.1.1. Система повинна мати клієнт-серверну архітектуру, що використовує базу даних MySQL (надалі – БД).

2.4.1.1.2. Діагностика функцій системи повинна включати відхилення від нормального процесу виявлення та усунення проблем і невідповідностей у роботі комп’ютерів і технічних засобів, а також програмних помилок і надавати користувачам відповідну діагностичну інформацію.

Зв'язок між підсистемами повинен здійснюватися на інформаційному рівні через єдину БД із використанням певних технічних засобів локальних комп’ютерних мереж

2.4.1.1.3. При розробці та впровадженні даної інформаційної системи, розвиток та модернізація системи має здійснюватись методом додавання, уточнення, або зміною виконуваних її функцій та технологій. Структура і технологія програмного забезпечення системи повинні забезпечити простоту їх модернізації та розвитку, з можливістю збільшення розмірності задач і масивів інформації, а також можливості реалізації їх на нових ПК.

Програмно-технічні засоби функціонування системи повинні мати текстові, табличні та графічні відображення даних. Програмна та інформаційна сумісність має забезпечуватися загальносистемним протоколом обміну, використанням

проблемно-орієнтованих пакетів прикладних програм міжмашинних зв’язків і єдиною системою класифікації і кодування.

2.4.1.1.4. Функціонування створеної інформаційної системи має забезпечувати діалогову та мережну обробку даних.

2.4.1.2.1. Персонал, який буде використовувати створену інформаційну систему підтримки обліку складу готової продукції, повинен дотримуватись наступних вимог:

- пройти навчання і отримати навички роботи на ПК;
- пройти ознайомлення з системою;
- дотримуватись технологічних інструкцій при роботі системи в діалоговому режимі;
- дотримуватись умов експлуатації ПК у відповідності вимог експлуатації;
- дотримуватись техніки безпеки при роботі з ПК.

2.4.1.2.2. Користувачами системи може виступати як начальник складу, старший комірник, так і комплектувальник відвантаження. Вхід у систему повинен здійснюватись методом вводу логіну та паролю користувача системи, який буде відображати його роль. В залежності від ролі, користувач може отримати різні рівні переліку функціоналу системи.

2.4.1.3. Показники призначення.

2.4.1.3.1. Відповідно до п. 2.1, показники призначення повинні характеризувати ступінь та якість обліку даних та управлінської діяльності на підприємстві для його оптимального функціонування. Перелік і допустимі значення показників, при яких зберігається цільове призначення системи, повинні бути визначені на стадії технороботочого проектування.

2.4.1.3.2. У процесі модернізації та розвитку, створювана система, повинна мати можливість до адаптування при зміні організаційних процесів та методів управління відділом.

2.4.1.4. Вимоги до надійності.

2.4.1.4.1. Система має великий спектр функцій і призначена для використання протягом робочого дня. Всі функції системи виконуються окремо. Кожна функція системи оцінюється на надійність згідно вимог ДСТУ 2226-93. Враховуючи особливості системи та її функцій, показником надійності є надійність СУБД, на якій вона реалізована, та технічних засобів, на яких вона експлуатується.

L_i — ймовірність безвідмовного виконання задачі в заданий термін (імовірність того, що і-тей запит буде виконаний);

$K_г$ — коефіцієнт готовності ПТК (програмно-технічного комплексу);

$T_в$ — середній час відновлення ПТК;

$T_е$ — мінімальний час між двома відмовами за календарний місяць.

2.4.1.4.2. Комплекс технічних засобів повинен передбачати:

- можливість запуску і функціонування з різних робочих станцій;
- можливість переходу на локальний режим роботи.

Для гарантування надійності програмного та інформаційного забезпечення необхідно передбачити використання:

- модульного, структурного, об’єктно-орієнтованого програмування;
- запобігання перевірки введеної інформації користувачем, та при помилці виводити виявлені помилки.

2.4.1.5. Вимоги до безпеки.

Для забезпечення безпеки при експлуатації, налагодженні, монтажі, обслуговуванні і ремонті технічних засобів системи потрібно дотримуватись вимог ДСТУ: ДСТУ 2293:2014, ДСТУ EN ISO 7010:2019, ДСТУ 12.0.230:2008, ДСТУ 7237:2011, ДСТУ 7238:2011, ДСТУ 7239:2011; по доступним рівням освітленості, вібраційних і шумових навантажень слід дотримуватися вимог відповідно ДСТУ Б А.3.2-15:2011, ДСТУ EN 14253:2018, ДСТУ 2867-94.

2.4.1.6. Вимоги з ергономіки та технічної естетики.

Загальні ергономічні і естетичні вимоги до системи повинні відповідати держстандартам ДСТУ 8604:2015, ДСТУ 7298:2013. Освітленість робочого місця повинна відповідати ДСТУ EN 12464-1:2016, ДБН В.2.5-28:2018.

Засоби відображення повинні розміщуватися таким чином, щоб кут спостереження екрану складав не більше, ніж 45 градусів, мінімальна відстань спостереження екрану — 0,3 м, рекомендована — 0,5 м.

При створенні програмного продукту слід подбати про зручність та інтуїтивність інтерфейсу користувача для полегшення його роботи.

2.4.1.7. Вимоги по експлуатації, технічного обслуговування, ремонту і зберігання компонентів системи.

2.4.1.7.1. Види обслуговування системи визначаються у відповідності з ДСТУ EN 13306:2019. Загальні вимоги по експлуатації, технічному обслуговуванню і ремонту повинні відповідати ДСТУ 3576-97.

2.4.1.7.2. Для розміщення технічних засобів системи необхідні площі, визначені в ДБН В.2.2-9:2018. При цьому слід дотримуватися вимог, зазначених в експлуатаційній документації. Напруга живлення технічних засобів системи 220 В змінного струму, частотою (50±1) Гц. Допустиме відхилення вхідної напруги від +10% до -15%, тривалість перерв у живленні не повинна перевищувати 0,001 с.

2.4.1.7.3. Склад, розміщення і умови зберігання компонентів технічних засобів системи визначається рекомендаціями, зазначеними в експлуатаційній документації на ці елементи.

2.4.1.7.4. Регламент обслуговування повинен відповідати їх рівню і умовам роботи, щоб у випадку відмови системи забезпечити роботу в аварійному режимі.

2.4.1.8. Вимоги до захисту інформації від несанкціонованого доступу.

Для надійності збереження і доступу до інформації необхідно використовувати засоби захисту:

1) локальної мережі та програми захисту в мережі Firewall.
2) клієнт-серверної СУБД:
    - тригери, представлення;

- процедури та функції;
- встановлення груп користувачів і ролей використання.

Крім цього, кожен сеанс роботи системи має розпочинатись з введення користувачем логіну та паролю для продовження роботи. Для надійного захисту від несанкціонованого доступу кожен із користувачів системи повинен мати персональний пароль. Крім того, деякі таблиці треба захистити від можливого редагування, доповнення чи вилучення інформації.

2.4.1.9. Вимоги щодо збереження інформації при аваріях.

2.4.1.9.1. Необхідно передбачити засоби резервного збереження бази даних і можливість завантажити БД у випадку її руйнування.

2.4.1.9.2. Резервний файл і БД мають знаходитись на різних машинних носіях чи пристроях.

2.4.1.10. Вимоги по захисту від впливу зовнішніх діянь.

2.4.1.10.1. Електрична складова електромагнітного поля завад в приміщеннях не повинна перевищувати 0,3 В/м^2 в діапазоні частот від 0,15 до 300 МГц. Для захисту від впливу електромагнітних полів та індустріальних завад слід передбачити різноманітні екрани та фільтри.

2.4.1.10.2. Засоби, які виключають вплив шкідливих факторів на функціонування комплексу технічних засобів, повинні бути запроектовані згідно з ДБН В.2.2-9:2018. Обчислювальні засоби по стійкості до зовнішніх впливів повинні відповідати ДСТУ 2506-94.

2.4.1.11. Вимоги до патентної чистоти.

При створенні даної системи патентні дослідження не проводяться.

2.4.1.12. Вимоги по стандартизації і уніфікації.

У системі кодування інформації необхідно проводити за світовим класифікатором і стандартом.

2.4.2. Вимоги до функцій.

2.4.2.1. Перелік функцій із зазначенням вхідної та вихідної інформації наведено в таблиці (див. Таблиця 2.1.).

Система повинна підвищити продуктивність обліку готової продукції. Функції системи мають забезпечити заповнення БД новою інформацією, формування звітів і виконання інших функцій, визначених чинним документом. При цьому, пріоритетом є зручне використання цієї системи за рахунок інтуїтивного інтерфейсу.

Таблиця 2.1. Перелік функцій, вхідної та вихідної інформації

№ п/п	**Найменування функції**	**Вхідна інформація**	**Вихідна інформація**
1	Формування відвантажень та їх виведення	Таблиці БД «Відвантаження», «Водії», «Продукти»	Форма з редагуванням відвантажень
2	Формування та виведення готової продукції	Таблиці БД «Продукти»	Формування інформації про готову продукцію
3	Пошук серед готової продукції	Запит користувача, «Продукти»	Результат пошуку в таблиці на формі
4	Пошук серед відвантажень	Запит користувача, «Відвантаження»	Результат пошуку в таблиці на формі
5	Пошук серед водіїв	Запит користувача, «Водії»	Результат пошуку в таблиці на формі
6	Авторизація	Таблиці БД «Користувачі»	Авторизація в системі
7	Видалення даних	Таблиці БД «Продукти», «Водії», «Відвантаження», «Продукти до відвантаження»	Видалені дані
8	Формування звіту по даті виробництва	Таблиця «Продукти»	Звіт по виробництву
9	Формування звіту по відвантаженням	Представлення «Деталі відвантаження»	Звіт по відвантаженням

2.4.3. Вимоги до видів забезпечення.

2.4.3.1. У вимогах до математичного забезпечення (МЗ) система не вимагає спеціального математичного забезпечення для реалізації покладених на неї функцій. Достатньо можливостей обраної СУБД.

2.4.3.2. Вимоги до інформаційного забезпечення (ІЗ).

2.4.3.2.1. Інформаційне забезпечення системи має містити дані, достатні для виконання всіх покладених на систему функцій. ІЗ повинно гарантувати раціональну організацію зберігання інформації та доступу до неї.

Заповнення БД інформацією покладається на замовника за методиками і формами, створеними розробниками системи.

Склад, структура і спосіб організації інформації представляються у логічній моделі БД і можуть уточнюватись на етапі технічного проектування.

2.4.3.2.2. Використовувати та створювати резервні копії БД задля запобігання руйнування інформації.

2.4.3.3. Вимоги до лінгвістичного забезпечення (ЛЗ).

2.4.3.3.1. Для розробки програмних продуктів, які забезпечують виконання функцій і обслуговування користувачів, повинні використовуватись мови програмування високого рівня для створення програмної логіки та мова запитів для взаємодії з базою даних.

2.4.3.3.2. Взаємодія користувача і системи має відбуватись через вікна меню і підказок, орієнтованих на виконувані функції.

Запити користувача мають задаватись переважно природною мовою.

2.4.3.4. Вимоги до програмного забезпечення (ПЗ).

2.4.3.4.1. Загальносистемне ПЗ має забезпечувати якісне та надійне виконання функціональних можливостей системи. Загальносистемним ПЗ вважають:

- операційна система (далі ОС) – Windows версії 10,11,8,8.1,7;
- система управління БД (далі СУБД) – MySQL.

2.4.3.4.2. Загальні вимоги до системного ПЗ можна описати так:

- мінімальні вимоги до ресурсів технічних засобів;

- максимальна швидкодія та продуктивність;
- надання повного переліку необхідних функцій системи.

2.4.3.4.3. Вимоги до ОС:

- мінімальне використання ресурсів технічних засобів для власних потреб;
- максимальна швидкодія при використанні зовнішніх пристроїв управління;
- ОС сервера – Windows 10, ОС клієнта – Windows 10.

2.4.3.4.4. Вимоги до СУБД:

- виконання потрібних функціональних задач;
- надійність системи;
- ефективне управління потрібного обсягу і структури;
- швидкість виконання запитів користувачів системи;
- мінімальні вимоги до ТЗ.

2.4.3.4.5. Програмні засоби введення та виведення даних повинні забезпечувати:

- виведення необхідних даних на екран у вигляді відповідних форматів;
- під час введення даних сповіщати користувача про наявні помилки та давати можливість їх виправлень;
- керований комп’ютером діалог при введені даних;
- виведення даних у відповідному вигляді за запитом користувача.

2.4.3.4.6. При розроблені спеціального ПЗ слід виконати наступні вимоги:

- використані програми мають бути сумісні між собою та загальносистемним ПЗ;
- ПЗ має розроблятись засобами об’єктно-орієнтованого програмування;
- забезпечити відповідність інтерфейсу користувача стандартам Windows;
- необхідна модульна структура програми;

- повинна бути передбачена можливість розширення складу задач у відповідності з новими функціональними потребами;
- ПЗ не має залежати від типу зовнішніх пристроїв;
- діалог користувача та системи має відбуватись за допомогою клавіатури та миші з поясненням виконуваних дій.

2.4.3.5. Вимоги до технічного забезпечення.

2.4.3.5.1. Технічні засоби системи (див Таблиця 2.2) повинні забезпечувати виконання функцій, перерахованих в таблиці 2.1.

2.4.3.5.2. Засоби обчислювальної техніки повинні забезпечувати обмін інформації в об’ємах, приведених вище в п. 2.4.3.2.

Таблиця 2.2. Вимоги до технічного забезпечення системи

№ п/п	**Основні характеристики комп’ютера**
Технічне забезпечення для сервера	
1	HP Proliant DL 380 Gen9 2 x Intel Xeon 8 Core 2,4 GHz\16 Gb\2 TB\ LAN 1 Gbit
Технічне забезпечення для клієнта	
1	I3-6100, 3.7 GHZ; RAM: 4 GB; HDD: 500 Gb;
2	Монітор для виведення зображення
3	Миша USB
4	Клавіатура USB

2.4.3.6. Вимоги до метрологічного забезпечення.

Ця система не вимагає наявності вимірювальних каналів, вимірювального обладнання чи приладів, тому вимоги до такого обладнання не ставляться.

2.4.3.7. Вимоги до організаційного забезпечення.

2.4.3.7.1. Організаційне забезпечення системи розробляється з вимогами державного стандарту АСУП.

2.4.3.7.2. При інтеграції системи не передбачається збільшення штатної чисельності підприємства. Територіальне розміщення робочих місць, на яких буде встановлена система. Визначається самим підприємством.

2.4.3.7.3. До функціонування системи висуваються наступні вимоги:

- наказом керівника підприємства визначається список співробітників, які матимуть доступ до системи;
- контроль і прийняття рішень при аварійних ситуаціях при експлуатації системи здійснює відповідальний за систему.

2.5. Склад і зміст робіт по створенню системи.

2.5.1. Стадії створення системи і терміни виконання робіт в таблиці нижче (див. Таблиця 3).

Таблиця 2.3. Найменування робіт при створенні системи

№ п/п	Найменування робіт	Строки виконання робіт
1	Передпроектне дослідження об’єкта автоматизації	23.02.2024
2	Технічне завдання	18.04.2024
3	Технічний проект	20.05.2024
4	Оформлення документації	25.05.2024

2.6. Порядок контролю і приймання системи

2.6.1. Система вводиться на діючому підприємстві ТДВ “Яготинський маслозавод”. При введенні в експлуатацію система повинна пройти приймальні випробування згідно ДСТУ 3974-2000.

2.6.2. Випробування проводяться розробником і замовником для перевірки працездатності системи та визначення її готовності до введення в експлуатацію.

Програма випробувань складається розробником і затверджується замовником.

2.6.3. Введення в експлуатацію здійснюється відповідно до умов та інструкцій з експлуатації. За результатами пусконалагоджувальних робіт розробляється план поліпшень і рекомендований термін його виконання.

2.6.4. Введення в дію системи оформляється актом здачі-прийому.

2.7. Вимоги до складу і змісту робіт з підготовки до введення системи в дію

Перед введенням в дію замовник виконує певний комплекс робіт з підготовки об'єкта, який включає в себе:

- укомплектування технічних засобів;
- організацію ознайомлення та навчання користувачів системи роботі на ПК і вивчення інструкцій з її експлуатації;
- проводить дослідну експлуатацію і вводить систему в дію.

2.8. Вимоги до документації

2.8.1. На систему розробляється комплекс документації у складі: технічне завдання та технічний проект.

2.8.2. Документація на систему розробляється у відповідності з вимогами Державних стандартів серії 19 «Єдина система програмної документації» та серії 24 «Єдина система стандартів систем управління».

2.9. Джерела розробки

2.9.1. При розроблені технічного завдання на систему використано наступні документи:

- ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання;
- ДСТУ 3973–2000 Система розроблення та поставлення продукції на виробництво;
- ДСТУ Б В.2.5–82:2016 Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом.

РОЗДІЛ 3. ОПИС КОМПЛЕКСУ ЗАДАЧ АВТОМАТИЗАЦІЇ

3.1. Інформаційне забезпечення системи

Інформаційна система такого роду не може існувати без власної бази даних. Опис бази даних відділу складу готової продукції слід почати з її моделі.

Побудована модель бази даних є реляційною та була побудована в середовищі phpMyAdmin (див. Рисунок В.1 в додатку В)[4]. Для побудови якісної та зрозумілої бази даних варто використовувати ключі відносин між таблицями.

Ключі відносин в базі даних дозволяють забезпечити унікальність та зв'язок між записами у різних таблицях. Вони забезпечують цілісність даних і дозволяють ефективно здійснювати пошук, зміну та видалення даних.

Існує декілька видів вище згаданих ключів, а саме:

- primary key (ключ, де один, або декілька стовпців таблиці мають унікальні значення кожного створеного рядка);
- foreign Key (зовнішній ключ таблиці, де стовпець таблиці містить дані з primary ключа іншої таблиці).

Отже, слід оглянути побудовані сутності бази даних:

- таблиця «product» (первинний ключем є стовпець «id»), яка містить інформацію про продукцію;
- таблиця «drivers» (первинним ключем є стовпець «id»), яка містить інформацію про водіїв відвантажень;
- таблиця «shipment» (первинним ключем є стовпець «ShipmentID»), яка містить інформацію про відвантаження готової продукції;
- таблиця «users» (первинним ключем є «UserId»), яка містить інформацію про користувачів системи;
- таблиця зв’язку «productstoshipment» (має зовнішні ключі «ShipmentID», «ProductId», «Номер_партії»), яка містить в собі інформацію про відвантаження та продукцію до відвантаження.

Більш детальний опис побудованих таблиць та представлень можна переглянути в таблиці(див Таблиця Б.1. в додатку Б).

3.2. Алгоритмізація та реалізація комплексу задач по створенню інформаційної системи

Для створення клієнтського додатку було використано середовище програмування Microsoft Visual Studio 2022 з використанням вбудованого фреймворку Windows Forms, який використовує об’єктно орієнтовану мову програмування C#[5-7]. Обране середовище з даним фреймворком надає можливість до використання графічного конструктора форм, засоби компілювання та інше.

В створений проект було додано бібліотеки для робити із сервером MySQL, після чого було прописане підключення до створеної локальної бази даних в окремому класі.

Виконання кожної функції, яка задіює базу даних відбувається через підключення до бази даних через створений клас, та виконання SQL запиту.

3.3. Функціонал клієнтської частини

Після створеної БД стояло питання розробити клієнтський додаток для користувача [1-3, 9]. Був створений проект програми, в яку додавались функції. Основний перелік функцій програми:

- облік готової продукції;
- облік відвантажень;
- облік водіїв;
- облік користувачів системи;
- створення звітів;
- авторизація в системі;
- резервне копіювання та завантаження бази даних.

3.3.1. Вікно авторизації користувача.

При відкритті програми, користувач має авторизуватись в системі. Це було реалізовано через таблицю «users» методом виконання SQL запиту. З вікна авторизації передаються дані з полів, по яким виконується запит на пошук цих полів в таблиці. Також з таблиці дістається і роль користувача в системі. Даний SQL запит та його виконання виглядає так:

```
DB db = new DB();
 DataTable table = new DataTable();
 MySqlDataAdapter adapter = new MySqlDataAdapter();
 MySqlCommand cmd = new MySqlCommand("SELECT Логін, Пароль, Роль FROM 
`users` WHERE Логін = @log AND Пароль = @pass", db.getConnection());
 cmd.Parameters.Add("@log", MySqlDbType.VarChar).Value = login;
 cmd.Parameters.Add("@pass", MySqlDbType.VarChar).Value = password;
 adapter.SelectCommand = cmd;
         adapter.Fill(table);
  if (table.Rows[0]["Роль"] != DBNull.Value)
  {
     role = table.Rows[0]["Роль"].ToString();
     this.Close();
     th = new Thread(openForm);
     th.SetApartmentState(ApartmentState.STA);
     th.Start();
  }
  else
  {
     MessageBox.Show("Помилка входу");
          }
```

Після виконання даних запитів, якщо авторизація пройшла успішно, в головну форму програми передаються дані авторизованого користувача (його логін, пароль та роль).

3.3.2. Головне меню програми

Після авторизації відкривається головне меню програми. В залежності від ролі авторизованого користувача доступний перелік функцій. Будемо розглядати перелік функцій для користувача з максимальним доступом прав.

3.3.2.1. Облік продукції.

Облік готової продукції був реалізований трьом вкладками в меню (Додавання продукції, Видалення продукції та Редагування продукції). Облік продукції відбувається виконанням супутніх SQL запитів до бази даних. Під час обліку зчитуються поля вводу інформації та передаються до відповідних таблиць БД.

3.3.2.2. Облік відвантажень.

Для обліку відвантажень була створена окрема форма, яка відкривається з кнопки на головній формі. На формі відвантажень було створено три кнопки, а саме «Створити заявку», «Продукти до замовлення» та «Водії». Кожна кнопка відкриває окрему вкладинку для обліку заявок, продуктів, які відвантажуються та обліку водіїв.

На вкладинках «Водії» та «Створити заявку» реалізовний пошук через не повне співпадіння, який здійснює пошук по різним колонкам, наприклад: Назва_відвантаження, Номер_авто.

Пошук був реалізовний через подію TextChanged текстового поля. Приклад коду пошуку виглядає наступним чином:

```
public void itemSearch()
 {

    DB db = new DB();
    DataTable table = new DataTable();
    MySqlCommand command = new MySqlCommand("SELECT Назва_відправки, Дата, Імя, Прізвище, Номер_авто, Деталі FROM shipment WHERE Назва_відправки LIKE @SearchText;", db.getConnection());
    command.Parameters.AddWithValue("@SearchText", "%" + SearchBox.Text + "%");
    MySqlDataAdapter adapter = new MySqlDataAdapter();
    adapter.SelectCommand = command;
    adapter.Fill(table);
    dataGridView1.DataSource = table;
    db.closeConnection();
        }
```

3.3.2.3. Створення звітів.

Створення звітів реалізовано на головному меню програми, де при натискані, відкривається вікно з генерацією обраних звітів. Дані звіти були реалізовані через фреймворк Microsoft Report Services. В програмі всього два можливих звіти (звіт з готової продукції по даті виробництва та звіт з відвантажень). Звіт з відвантажень був виконаний через представлення з бази даних «shipmentdetails». Це представлення включає в себе повний список інформації про відвантаження, водія та продукції.

3.3.2.4. Облік користувачів.

Облік користувачів здійснюється користувачем з максимальним рівнем допуску (адміністратор, або керівник). На даній вкладці, з головного меню, можна проводити облік користувачів, а саме додавати та видаляти їх. Слід зазначити, що видалення користувачів з максимальним рівнем прав не здійснюється.

3.3.2.5. Резервне копіювання бази даних.

Для запобігання втрати інформації та бази даних в цілому була створена вкладка «Імпорт/експорт» бази даних. При експорті бази даних робиться резервний файл цієї БД, а при імпорті цей файл завантажується на сервер. Слід відмітити, що ручне управління цією функцією є тільки в користувача з максимальними правами до системи, однак, відбувається автоматичне копіювання бази даних через 20 хвилин користування системою.

Реалізація цієї функції відбувалась через додатковий фреймворк для роботи з базами MySQL, а саме «MySqlBackup».

3.3.3. Обробка помилок

3.3.3.1. Обробка помилок введення даних.

На всіх передбачених формах та вкладках створена перевірка помилок вводу інформації. Є різні обробники, а саме:

- помилка при введенні пустого текстового поля;
- блокування текстових символів в текстових полях, де передбачається введення цифр;

Ці обробники забезпечують правильне експлуатування системи користувачем, що є дуже важливим фактором.

Приклад реалізації таких обробників:

```
if (string.IsNullOrWhiteSpace(NameBox.Text) ||
string.IsNullOrWhiteSpace(SurnameBox.Text) ||
```

```
string.IsNullOrWhiteSpace(lastNameBox.Text)                    ||
string.IsNullOrWhiteSpace(CarnumberBox.Text)                   ||
string.IsNullOrWhiteSpace(PhoneBox.Text))
 {
    MessageBox.Show("Заповніть всі поля");
 }
 else
        {
        //Виконання коду, якщо всі поля заповнені
        }
if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
 {
    e.Handled = true;  // Відхилити введення, якщо символ не є цифрою або клавішею управління
        }
```

3.3.3.2. Обробка помилок при зверненні до серверу.

Обробки помилок при зверненні до серверу включать в себе побудовані конструкції «try - catch». Ці конструкції використовуються під час звернення програми до серверу, як то введення інформації, виведення та інші операції із серевером.

3.4. Інструкція користувача

Для початку роботи з системою слід відкрити її «Warehouse.exe», після чого буде відкрито вікно авторизації користувача (див Рисунок 3.1).

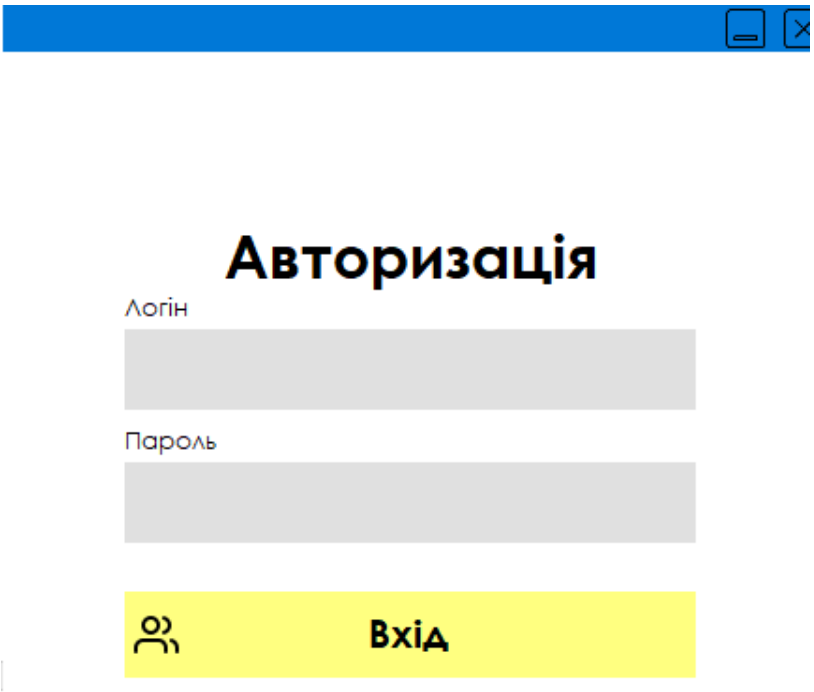

Рисунок 3.1 - Вікно авторизації

Для подальшої роботи із системою користувач має ввести свої дані для входу. Кожен користувач має свої певні права в системі, які надають перелік доступного функціоналу.

Після успішної авторизації користувача буде доступне головне меню системи, залежно від його прав відкриються різні головні меню (див. Рисунок 3.2 – 3.4).

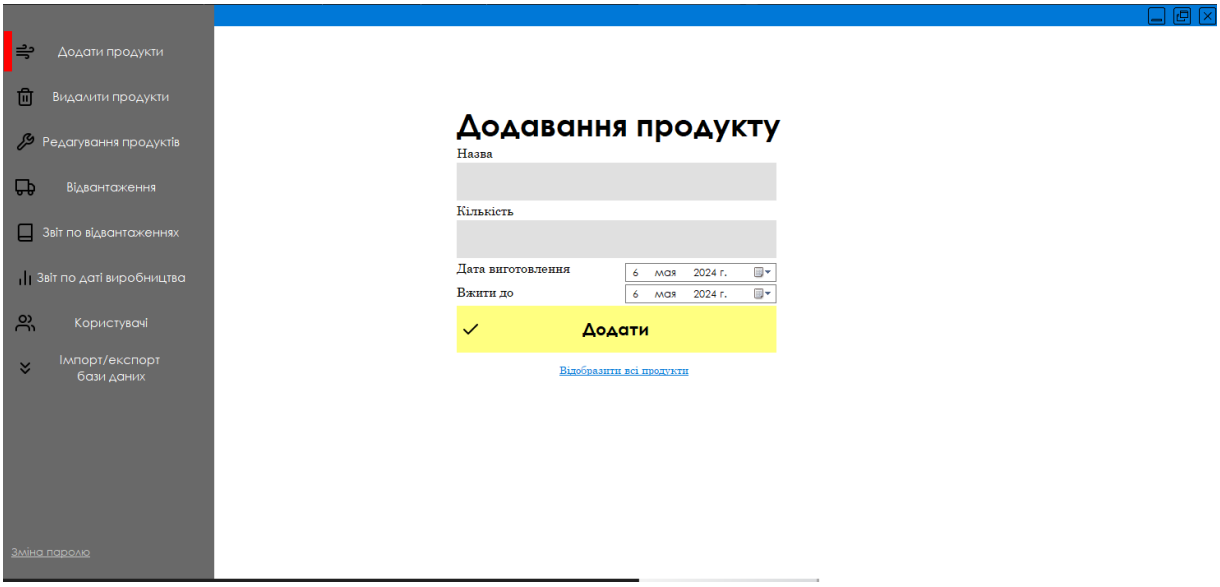

Рисунок 3.2 - Вигляд меню з повним доступом до всіх функцій

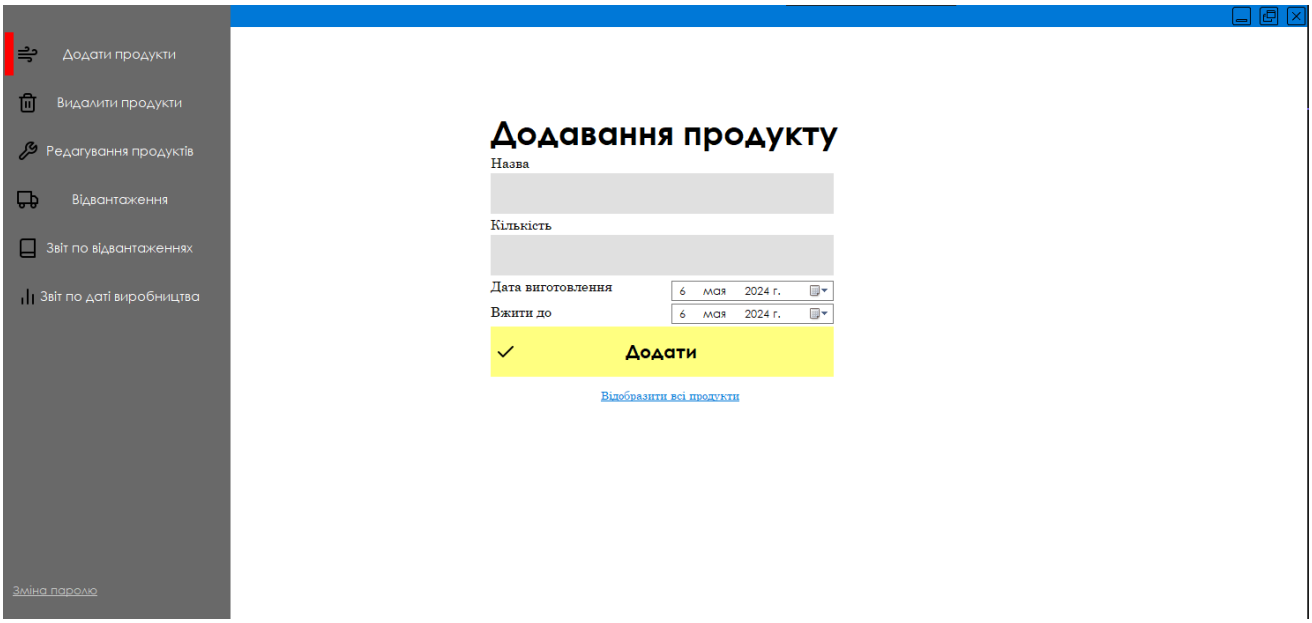

Рисунок 3.3 - Меню для звичайного користувача

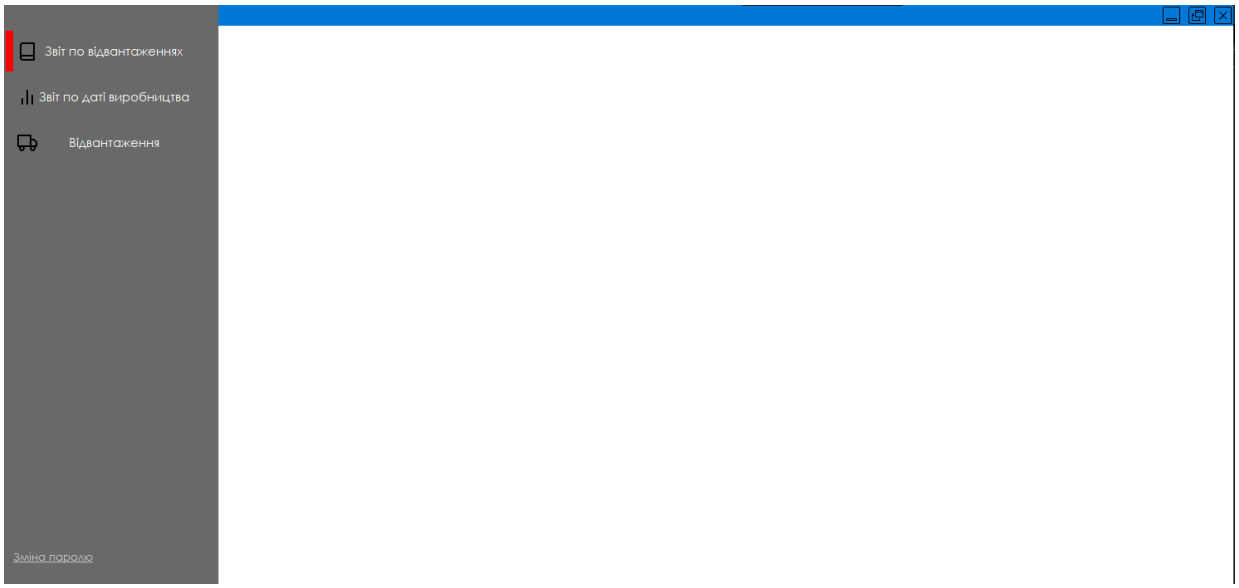

Рисунок 3.4 - Головне меню з обмеженими функціями

Головне меню системи з всіма правами (з більш обмеженими правами доступу, функції просто стають не доступні) має такі вкладки:

- додати продукти;
- видалити продукти;
- редагування продуктів;
- відвантаження;
- звіт по відвантаженням;
- звіт по даті виробництва;
- користувачі;
- імпорт/експорт бази даних;
- зміна паролю.

Деякі із вкладень відкривають інші окремі меню, а саме «Відвантаження», «Звіт по даті виробництва», «Звіт по відвантаженням», ці окремі меню зображені на рисунках нижче (див. Рисунки 3.5 – 3.7).

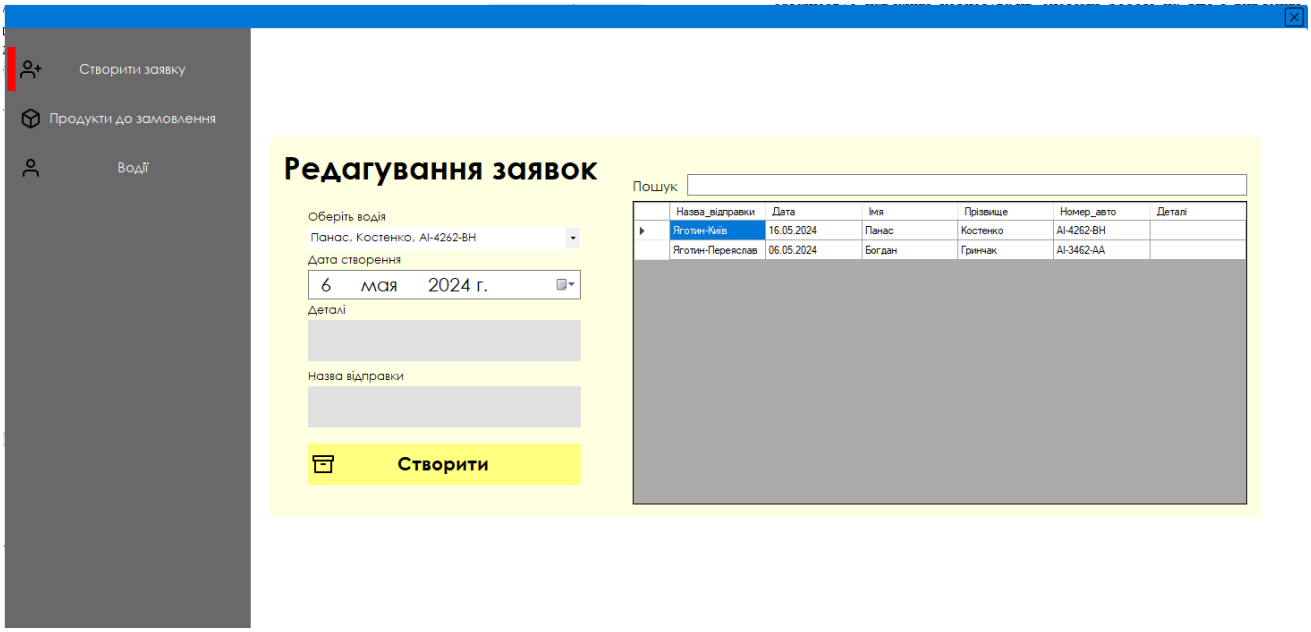

Рисунок 3.5 - Меню відвантажень

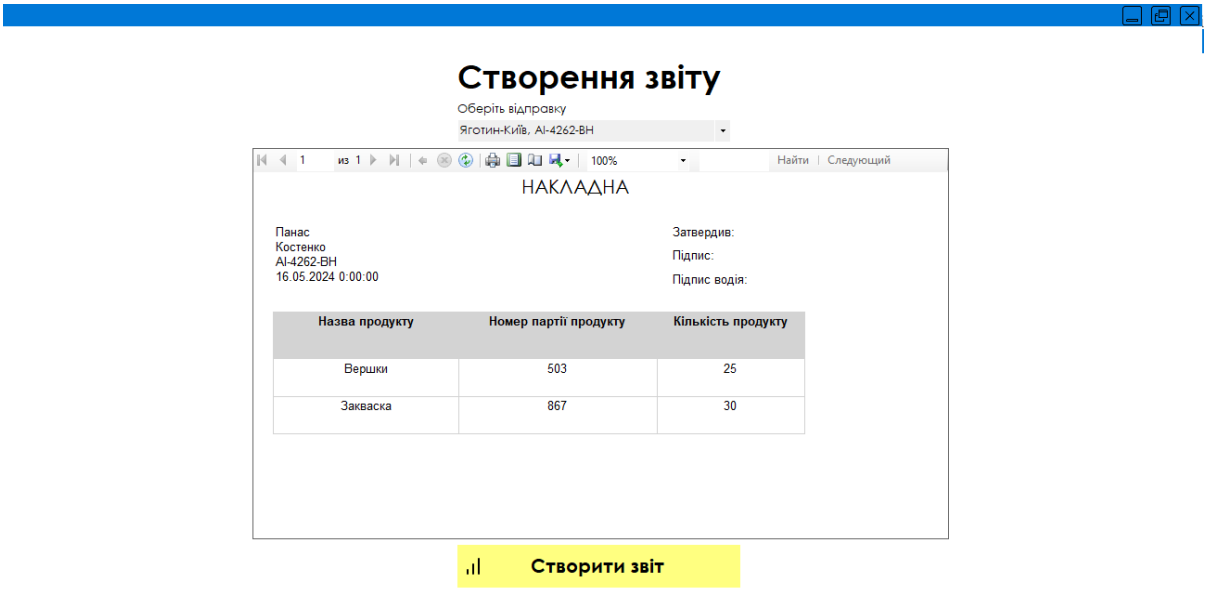

Рисунок 3.6 - Меню «Звіт по відвантаженнях»

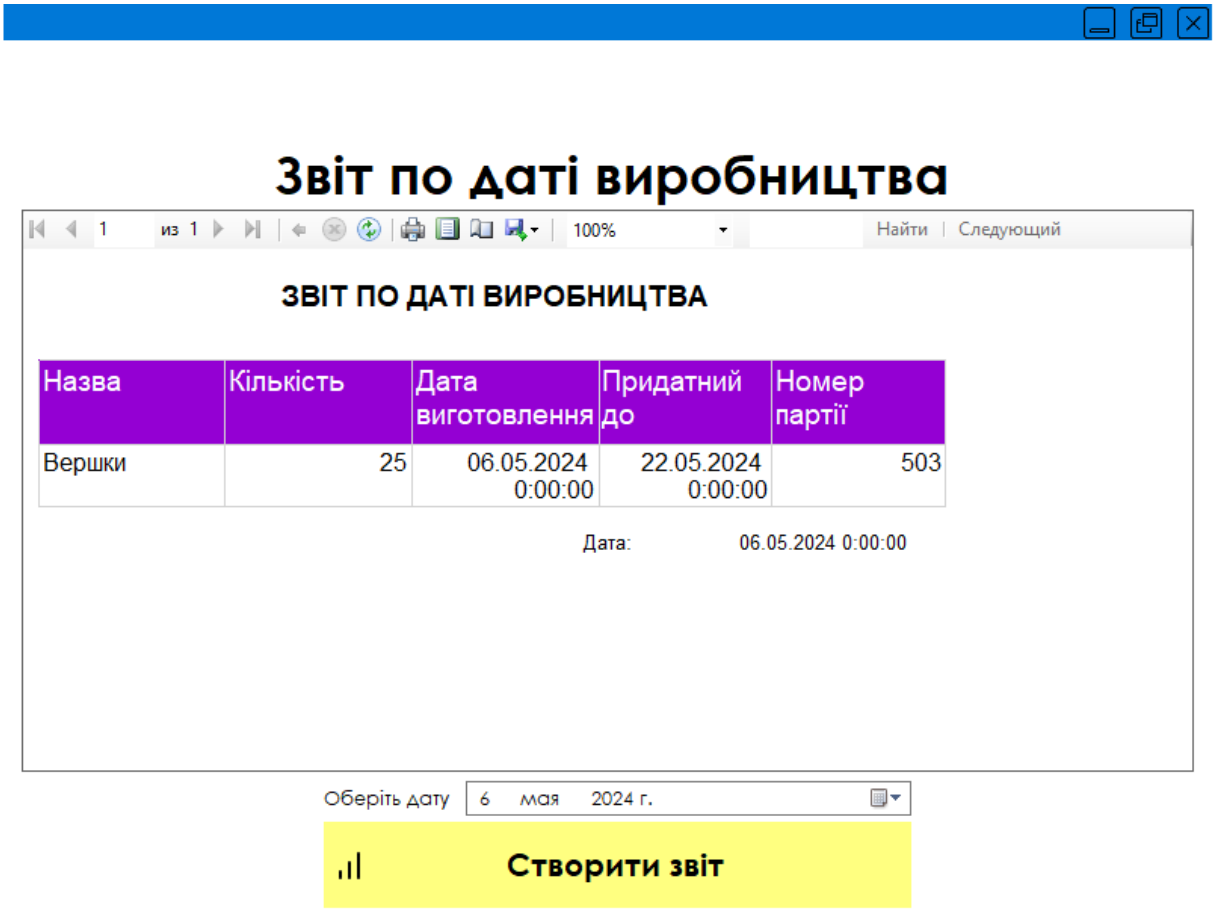

Рисунок 3.7 - Меню «Звіт по даті виробництва»

Отже, в меню відвантажень ми маємо такі вкладки:

- створити заявку;
- водії;
- продукти до замовлення.

В цих вкладках користувач може створити заявку на відвантаження, видалити та редагувати їх. Також додати/редагувати/видалити водіїв до відвантажень та редагувати продукти до замовлень відвантажень.

Вигляд цих вкладок наведено на рисунках нижче (див. Рисунки 3.8 – 3.10).

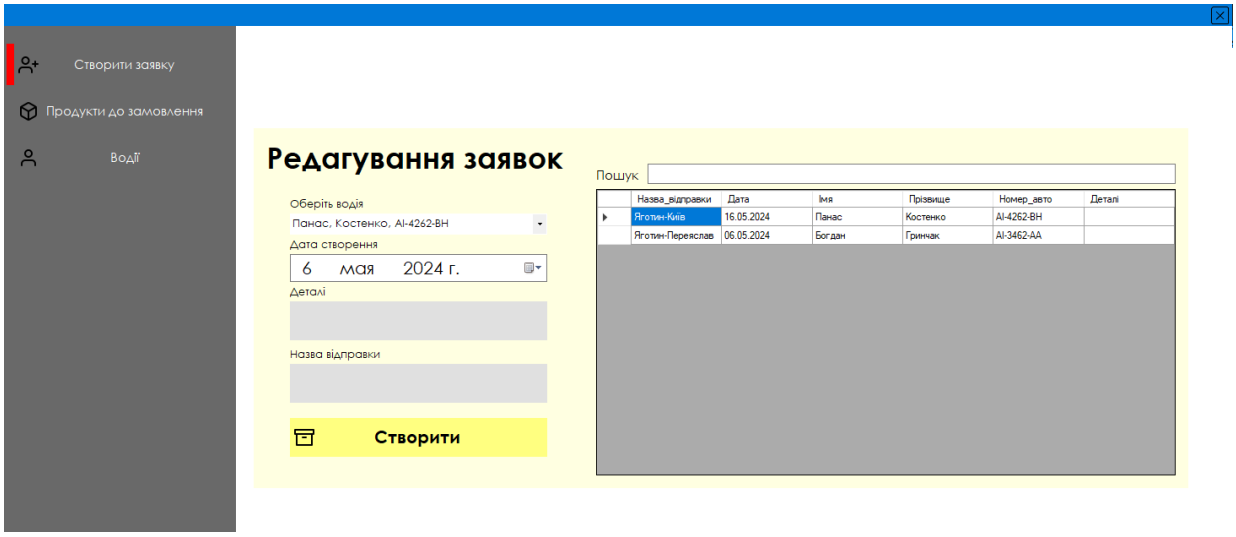

Рисунок 3.8 - Створення заявки

На даній вкладці користувач може редагувати заявки на відвантаження. Для того, щоб додати заявку, користувач має заповнити всі поля та натиснути кнопку «Створити», після чого, заявка буде створена. Для видалення заявки слід її виділити на натиснути клавішу «Delete» на клавіатурі.

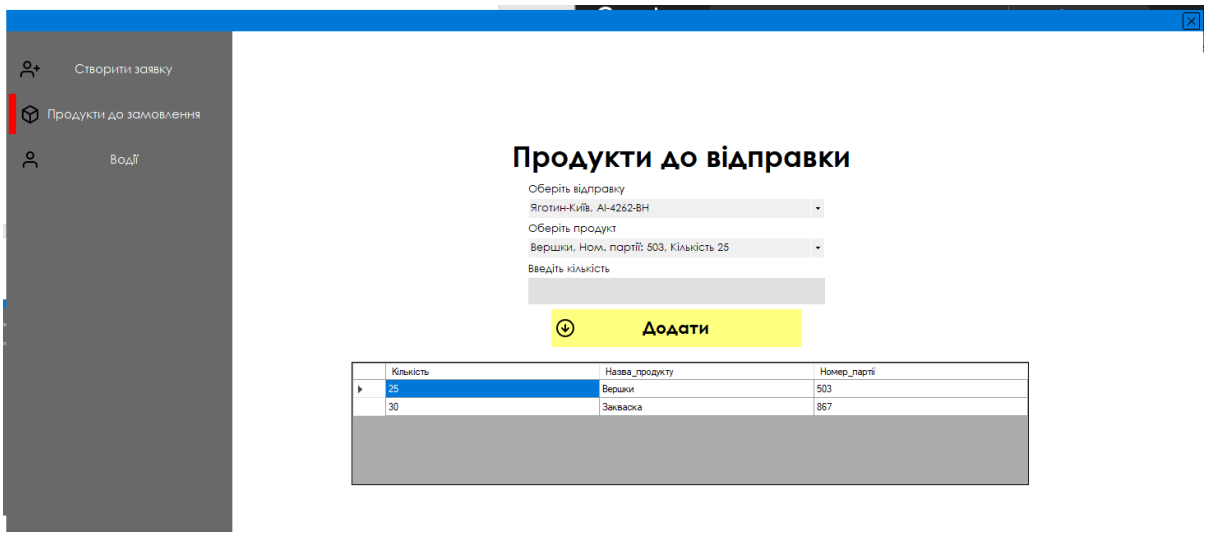

Рисунок 3.9 - Продукти до замовлення

Тут користувач може додавати до певного відправлення продукцію, яка буде відправлена. Для того, щоб додати продукцію, слід заповнити всі поля та натиснути «Додати». Для видалення продукції слід її виділити та натиснути клавішу «Delete» на клавіатурі.

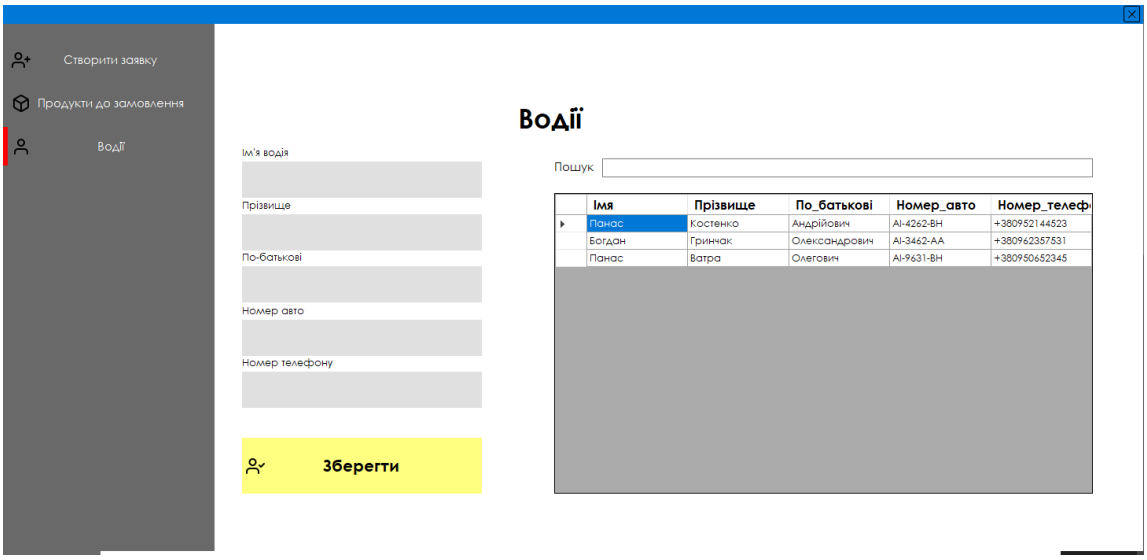

Рисунок 3.10 - Редагування водіїв

На вкладці «Водії», користувач може редагувати інформацію про водіїв, для того, щоб додати водія, слід заповнити всі поля інформацією та натиснути «Зберегти». Для видалення водія слід виділити запис та натиснути клавішу «Delete» на клавіатурі.

Отже, тепер повернемось до головного меню системи, де буде продемонстровано вигляд вкладок цього меню. Вигляд вкладок представлено на рисунках нижче (див. Рисунки 3.11 – 3.16).

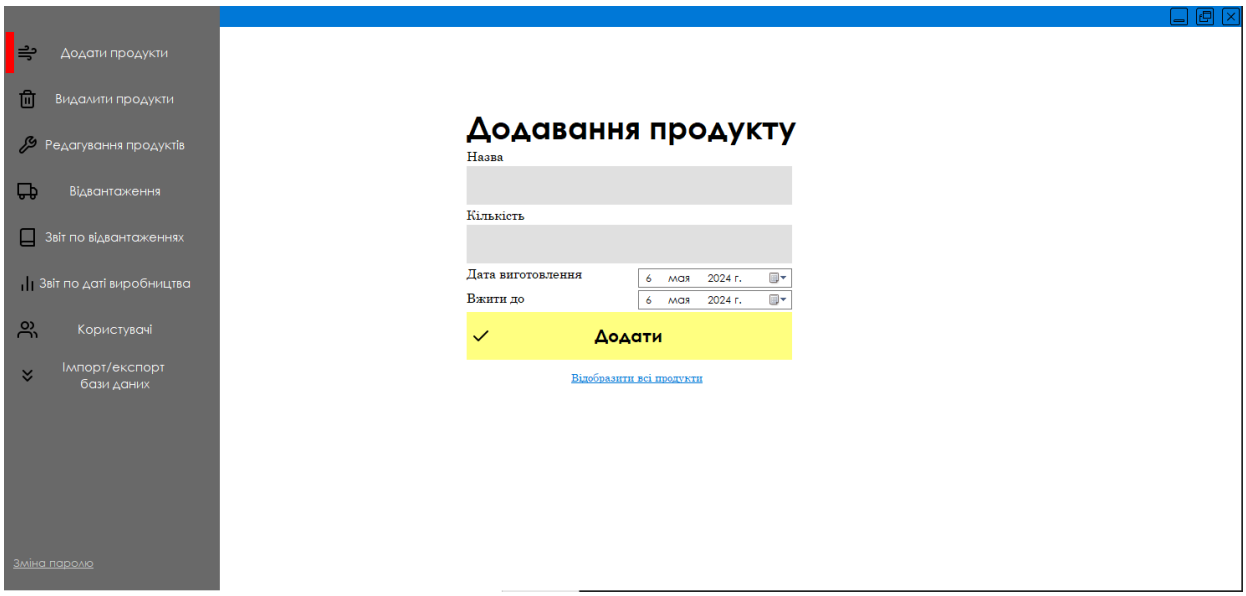

Рисунок 3.11 - Додавання продукту

На даній вкладці (див Рисунок 3.11) користувач може додавати готову продукцію, слід заповнити поля та натиснути «Додати».

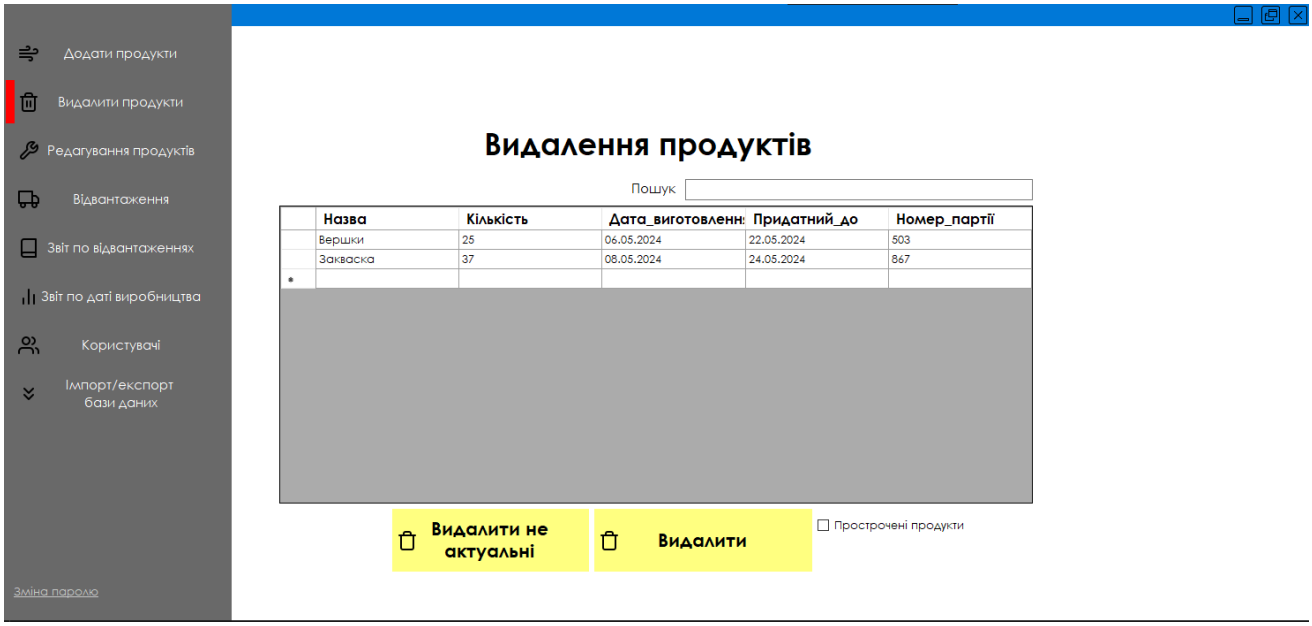

Рисунок 3.12 – Видалення продуктів

При видаленні продукції слід її виділити та натиснути «Видалити». Також на цій формі (див. Рисунок 3.12) користувач може видалити не актуальну продукцію (продукція, яка вже закінчилась) лише одною кнопкою «Видалити не актуальні». З додаткового, це пошук продукції по назві, в полі «Пошук» та виведення простроченої продукції.

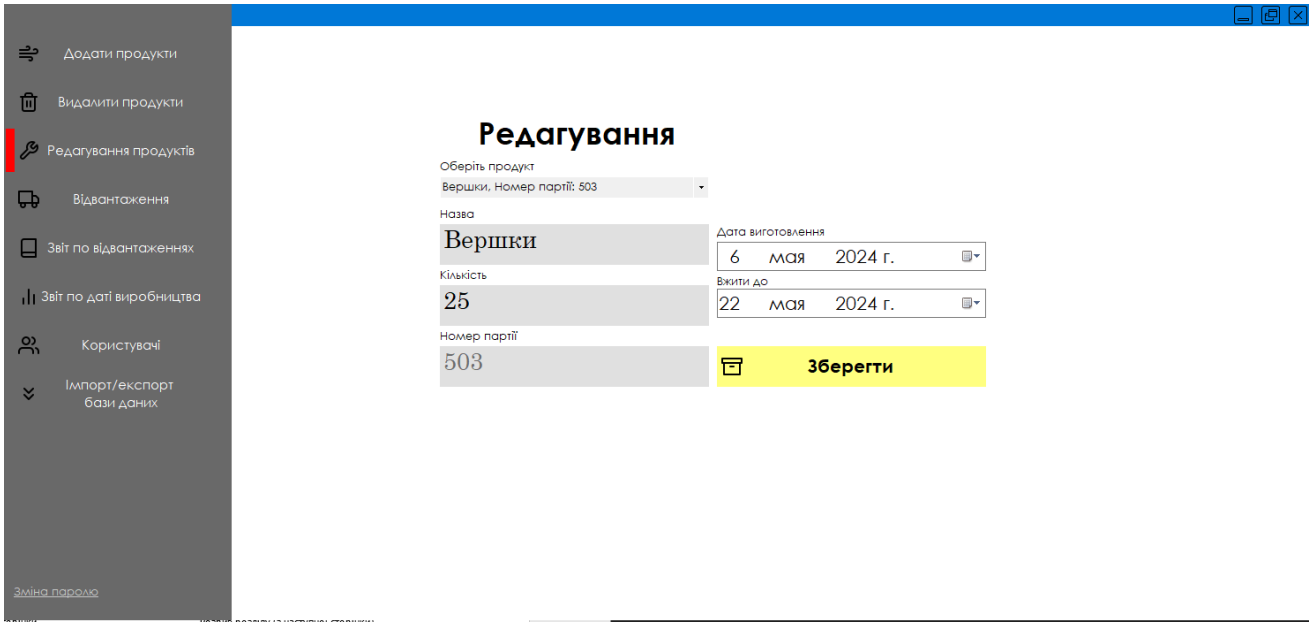

Рисунок 3.13 - Редагування продуктів

На вкладці «Редагування продуктів» (див. Рисунок 3.13) користувач може редагувати додані продукти. Слід лиш обрати зі списку продукцію, яку треба відредагувати, заповнити поля новими значеннями та натиснути «Зберегти».

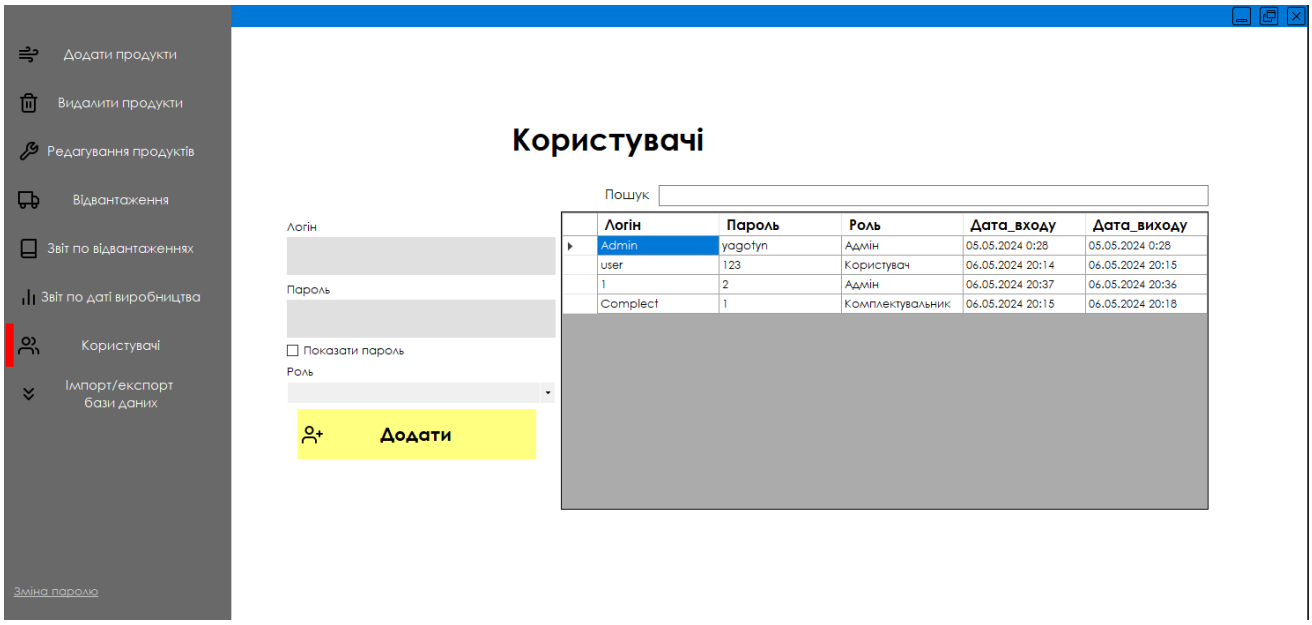

Рисунок 3.14 - Користувачі

На відкритій вкладці користувачів, є можливість створювати користувачів, видаляти їх, та надавати права доступу (див. Рисунок 3.14).

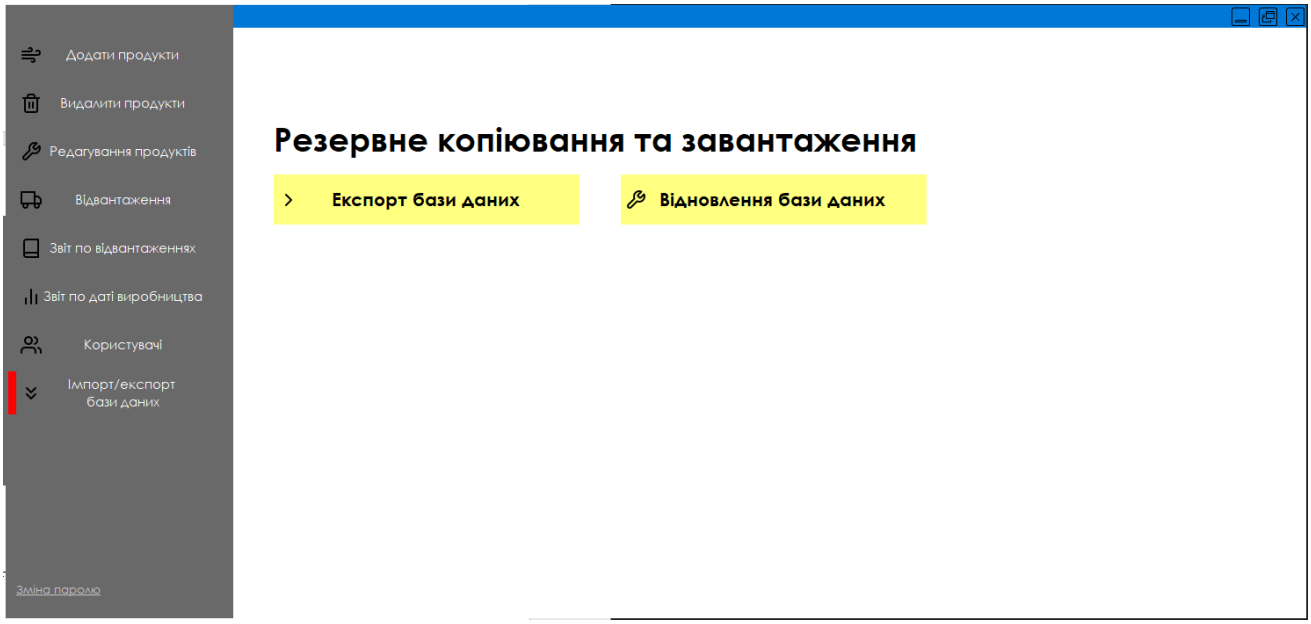

Рисунок 3.15 - Імпорт/експорт бази даних

На даній вкладці користувач може робити резервну копію бази даних (див. Рисунок 3.15). При натисканні на «Експорт бази даних» створюється резервний файл, а при «Відновлення бази даних» буде завантажено цей файл.

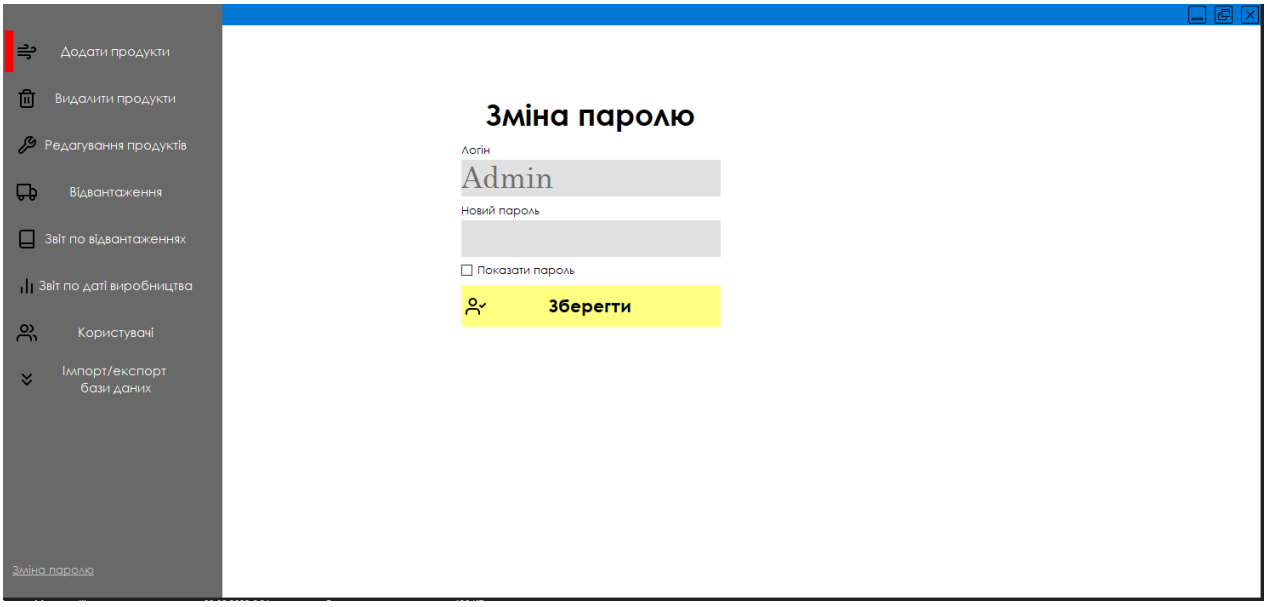

Рисунок 3.16 - Зміна паролю

При виборі даної вкладки (див. Рисунок 3.16) користувач може змінити пароль до свого облікового запису.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ

Такий аспект, як охорона праці є досить важливим в робочому середовищі. З метою забезпечення безпеки працівників на робочому місці, необхідно розробити певний перелік вимог та рекомендацій щодо охорони праці. Для уникнення негативних наслідків та зменшення ризиків виникнення аварійних ситуацій слід дотримуватись таких вимог:

Вимоги щодо обладнання:

- проведення очистки поверхонь від пилу та бруду;
- забезпечення притоку свіжого повітря.

Вимоги щодо забезпечення працівників:

- забезпечення достатньої кількості вбиралень;
- забезпечення дезинфікуючих засобів;
- забезпечення працівників питною водою.

Безпека для здоров’я працівників:

- надання наборів першої допомоги (аптечки);
- перед початком роботи перевіряти стан здоров’я працівників.

Вимоги щодо робочої організації:

- надання можливості відпочинку на протязі робочого дня;
- проведення інструктажів щодо роботи;
- контроль за станом робочих місць.

Дотримання даних вимог є обов’язковим до виконання всіма працівниками для запобігання надзвичайних ситуацій на робочих місцях.

Дотримання пожежної безпеки на робочу місці спрямовано на уникнення виникнення пожежних ситуацій. В разі виникнення пожежі, працівники повинні дотримуватись всіх інструкцій, для запобігання травм. Роботодавець повинен проводити інструктажі щодо пожежної безпеки та надати всі необхідні засоби щодо надзвичайних ситуацій (як то вогнегасники, тощо).

Правила пожежної безпеки затверджені згідно з регламентом ДСТУ, а саме:

- ДСТУ 8828:2019 – Пожежна безпека. Загальні положення;

- ДСТУ ISO 16069:2012 Графічні символи. Знаки безпеки. Системи позначення безпечного евакуювання;
- ДСТУ 7288:2012 Вогнегасні речовини;
- ДСТУ 3855-99 Визначення пожежної небезпеки матеріалів та конструкцій.

Наведені основні вимоги розроблені для мінімізації виникнення пожежних ситуацій на робочому місці. Роботодавець несе пряму відповідальність за відповідність робочих місць щодо пожежної безпеки та дотримання працівниками цих вимог.

Додатково, роботодавець відповідальний за проведення інструктажів щодо пожежної безпеки на робочих місцях.

ВИСНОВКИ

В даній кваліфікаційній роботі на тему “Створення інформаційної системи підтримки обліку складу готової продукції” була створена інформаційна система для поліпшення продуктивності та запобіганню помилок при обліку складського відділу з готової продукції підприємства.

Створена система забезпечує зручне та повне управління обліком від готової продукції, так і до її відвантажень. Слід зазначити, що дана система розроблялась саме для користувача, що також відіграє роль в швидкості користуванні нею.

Структура вихідного програмного забезпечення є досить простою у використанні та з інтуїтивно зрозумілим інтерфейсом, що зменшить час на опанування її співробітниками підприємства.

Користувачами даної системи є співробітники складського відділу підприємства, а саме начальник складу, старший комірник, комірник та комплектувальник. Для користування системою користувач має мати свій обліковий запис в системі, після чого він зможе авторизуватись та користуватись нею.

Інтеграція даної інформаційної системи надасть високу швидкодію як складському відділу, так і підприємству загалом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Проектування інформаційних систем [Електронний ресурс] [Текст] : конспект лекцій для студ. освіт. ступ. "Бакалавр" спец. 122 "Комп'ютерні науки" ден. та заоч. форм навч. / О. М. М'якшило, О. В. Харкянен ; Нац. ун-т харч. технол. — Київ : НУХТ, 2018. — 47 с. — каф. інформаційних систем.
2. Проектування та розробка програмного забезпечення [Електронний ресурс] [Текст] : лабораторний практикум для здобувачів освіт. ступ. "Бакалавр" спец. 122 "Комп'ютерні науки" освіт.-проф. програм "Комп'ютерні науки", "Інформаційні системи та штучний інтелект" ден. та заоч. форм навч. / уклад. : О. М. М'якшило, О. В. Харкянен ; Нац. ун-т харч. технол. — Київ : НУХТ, 2022. — 102 с. — каф. інформаційних технологій, штучного інтелекту і кібербезпеки.
3. Проектування та розробка програмного забезпечення [Електронний ресурс] [Текст] : метод. рекомендації до викон. курсового проекту для здобувачів освіт. ступ. "Бакалавр" спец. 122 "Комп'ютерні науки" освіт.-проф. програм "Комп'ютерні науки", "Інформаційні системи та штучний інтелект" ден. та заоч. форм навч. / уклад. : О. М. М'якшило, О. В. Харкянен ; Нац. ун-т харч. технол. — Київ : НУХТ, 2023. — 27 с. — каф. інформаційних технологій, штучного інтелекту і кібербезпеки.
4. Офіційний сайт phpMyAdmin [Електронний ресурс] – Режим доступу до ресурсу: https://www.phpmyadmin.net/
5. Фреймворк для роботи з базою даних MySQL в середовищі Visual Studio. [Електронний ресурс] – Режим доступу до ресурсу: https://forums.mysql.com/read.php?38,709633,709633
6. Інструкція по підключенню бази даних MySQL до проекту Visual Studio. [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@constanza.dc.999/connect-to-your-mysql-database-using-c-windows-forms-55aaa3c6e7c4

7. Створення програми з використанням MySQL.[Електронний ресурс] – Режим доступу до ресурсу: https://dizzpy.medium.com/rbuilding-a-login-form-application-with-c-net-and-mysql-database-647a55b3f1bd
8. Офіційна сторінка досліджуваного підприємства [Електронний ресурс] – Режим доступу до ресурсу: https://milkalliance.com.ua/company/enterprises/yagotinskij-maslozavod/ -
9. Приклад реалізації інтерфейсу користувача. [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@riazraza0/designing-a-beautiful-flat-dashboard-ui-ux-for-desktop-application-in-c-winforms-using-visual-408f4dcb4d81
10. Звіт про управління ТДВ ”Яготинський маслозавод” [Електронний ресурс] – Режим доступу до ресурсу: http://1.https//milkalliance.com.ua/tools/cms/site/download.php?url=/uploads/site_factory_docs/file/0008/11.pdf&name=zvit-pro-upravlinnya-za-2019-rik
11. Офіційний сайт підприємства “Яготинський маслозавод” [Електронний ресурс] – Режим доступу до ресурсу: https://milkalliance.com.ua/company/enterprises/yagotinskij-maslozavod/
12. Офіційний сайт з інформацією про Oracle Warehouse Management [Електронний ресурс] – Режим доступу до ресурсу: https://www.oracle.com/scm/logistics/warehouse-management/
13. Офіційний сайт IBS Виробництво [Електронний ресурс] – Режим доступу до ресурсу: https://www.ibsystems.com.ua/ua/ibs-%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B0-%D0%B4%D0%BB%D1%8F-%D0%B0%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D1%96%D1%97-%D0%BE%D0%B1%D0%BB%D1%96%D0%BA%D1%83-%D0%B2%D0%B8%D1%80%D0%BE%D0%B1%D0%BD%D0%B8%D1%86%D1%82%D0%B2%D0%B0/%D0%BE%D0%BF%D0%B8%D1%81

14. ДСТУ 3008:2015 — Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. – К.: ДП «УкрНДНЦ», 2015. – 32 с
15. ДСТУ 3918:1999 (ISO/IEC 12207:2008). Інформаційні технології. Процеси життєвого циклу програмного забезпечення. – 57 с.
16. ДСТУ ISO/IEC TR 15504. Інформаційні технології. Оцінювання процесів життєвого циклу програмних засобів. – 315 с.
17. ДСТУ 2226:1993. Автоматизовані системи.
18. ДСТУ ISO/IEC 27000:2015. Інформаційні технології. Методи захисту. Система управління інформаційною безпекою. Огляд і словник.
19. ДСТУ ISO/IEC 12207:2016 (ISO/IEC 12207:2008, IDT). Інженерія систем і програмного забезпечення. Процеси життєвого циклу програмного забезпечення.
20. ДСТУ ISO/IEC/IEEE 29119-1:2017. Інженерія систем і програмних засобів. Тестування програмних засобів. Частина 1. Поняття та визначення (ISO/IEC/IEEE 29119-1:2013, IDT).
21. ДСТУ 2941:1994. Системи оброблення інформації. Розроблення систем. Терміни та визначення.
22. ДСТУ 1.0:2003. СТУ 1.0:2003. Національна стандартизація. Основні положення.
23. ДСТУ 3321:2003. Системи конструкторської документації. Терміни та визначення основних понять.
24. ДСТУ ISO 6309:2007. Пожежна безпека. Загальні вимоги.
25. ДСТУ ISO/IEC 29155-1:2015. Розроблення систем і програмного забезпечення. Платформи для тестування проектів з розроблення інформаційних систем. Частина 1. Концепції та визначення.
26.. ДСТУ ISO/IEC 12207:2014. Інженерія систем і програмного забезпечення. Процеси життєвого циклу програмного забезпечення.
27. ДСТУ ISO/IEC 15910:2012. Інформаційні технології. Документування програм. Документація користувача.

28. Гайна Г. А. Основи проектування баз даних : навч. посіб. – К. : КНУБА, 2005. – 204 с.
29. Пасічник В. В., Резніченко В. А. Організація баз даних та знань. – К. : BHW, 2006. – 384 с.
30. Управління ІТ проектами [Електронний ресурс]: методичні рекомендації до виконання курсової роботи для здобувачів освітнього ступеня «Бакалавр» спеціальності 122 «Комп'ютерні науки» освітньо – професійних програм «Комп'ютерні науки» та «Інформаційні системи та штучний інтелект» денної та заочної форм навчання. / Уклад.: С. В. Грибков, О. Л. Сєдих – К.: НУХТ, 2023. – 76 с.
31. Яготинський маслозавод підприємство харчової промисловості у місті Яготин Яготинського району Київської області України [Електронний ресурс] – Режим доступу до ресурсу: https://www.wikidata.uk-ua.nina.az/%D0%AF%D0%B3%D0%BE%D1%82%D0%B8%D0%BD%D1%81%D1%8C%D0%BA%D0%B8%D0%B9_%D0%BC%D0%B0%D1%81%D0%BB%D0%BE%D0%B7%D0%B0%D0%B2%D0%BE%D0%B4.html

ДОДАТКИ

ДОДАТОК А. Функціональна модель

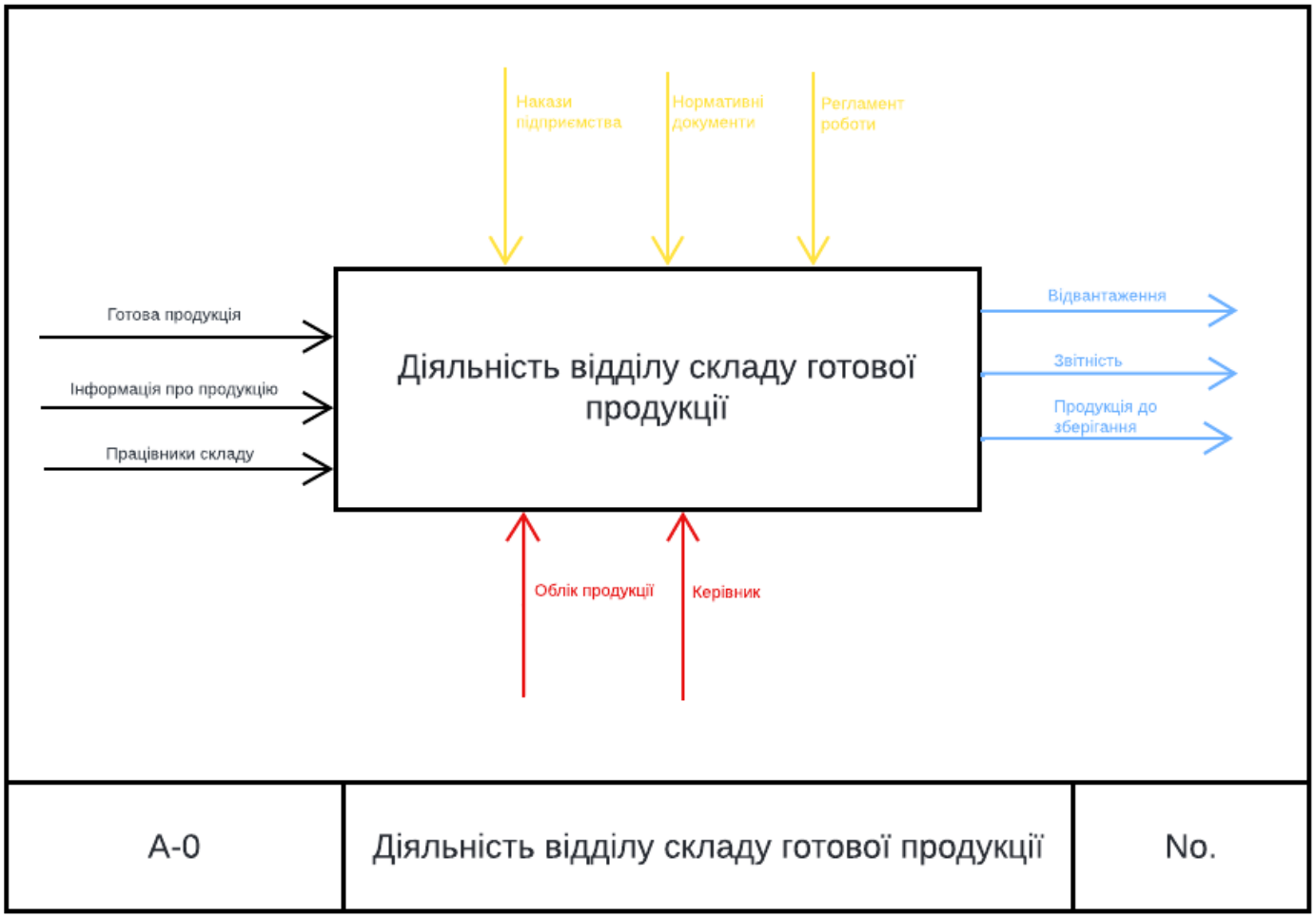

Рисунок А.1 - Функціональна модель

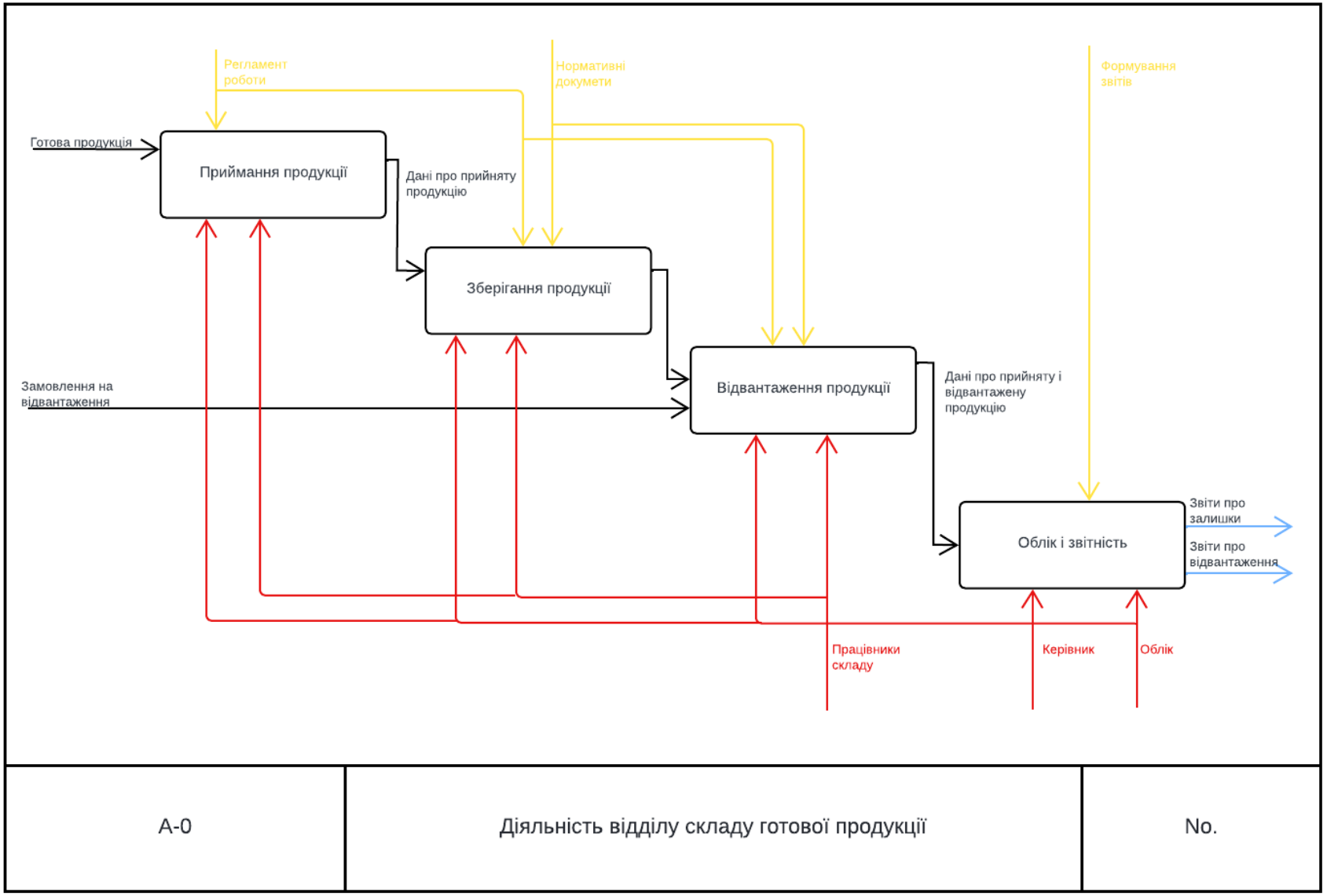

Рисунок А.2 - Декомпозиція першого рівня

ДОДАТОК Б. Опис сутностей бази даних

Таблиця Б.1. Опис сутностей

Назва таблиці та представлень	**Стовпець**	**Тип**
1	**2**	**3**
Таблиця «users»	UserID	Int
	Логін	Varchar
	Пароль	Varchar
	Роль	Varchar
	Дата_входу	Datetime
	Дата_виходу	Datetime
Таблиця «product»	id	Int
	Назва	Varchar
	Кількість	Int
	Дата_виготовлення	Date
	Придатний_до	Date
	Номер_партії	Int
Таблиця «drivers»	Id	Int

Продовження таблиці Б.1. Опис сутностей

1	2	3
	Імя	Varchar
	Прізвище	Varchar
	По_батькові	Varchar
	Номер_авто	Varchar
	Номер_телефону	Varchar
Таблиця «shipment»	ShipmentID	Int
	DriverID	Int
	Назва_відправки	Varchar
	Дата	Date
	Імя	Varchar
	Прізвище	Varchar
	Номер_авто	Varchar
	Деталі	text
Таблиця «productstoshipment»	ShipmentID	Int
	ProductId	Int
	Кількість	Int

Продовження таблиці Б.1. Опис сутностей

1	2	3
	Назва_продукту	Varchar
	Номер_партії	Int
Представлення «shipmentdetails»	ID_Відправки	Int
	Ім_я_водія	Varchar
	Прізвище_водія	Varchar
	Номер_авто_водія	Varchar
	Назва_відправки	Varchar
	Дата_відправки	Date
	Назва_продукту	Varchar
	Номер_партії_продукту	Int
	Кількість_продукту	int

ДОДАТОК В. Схема бази даних

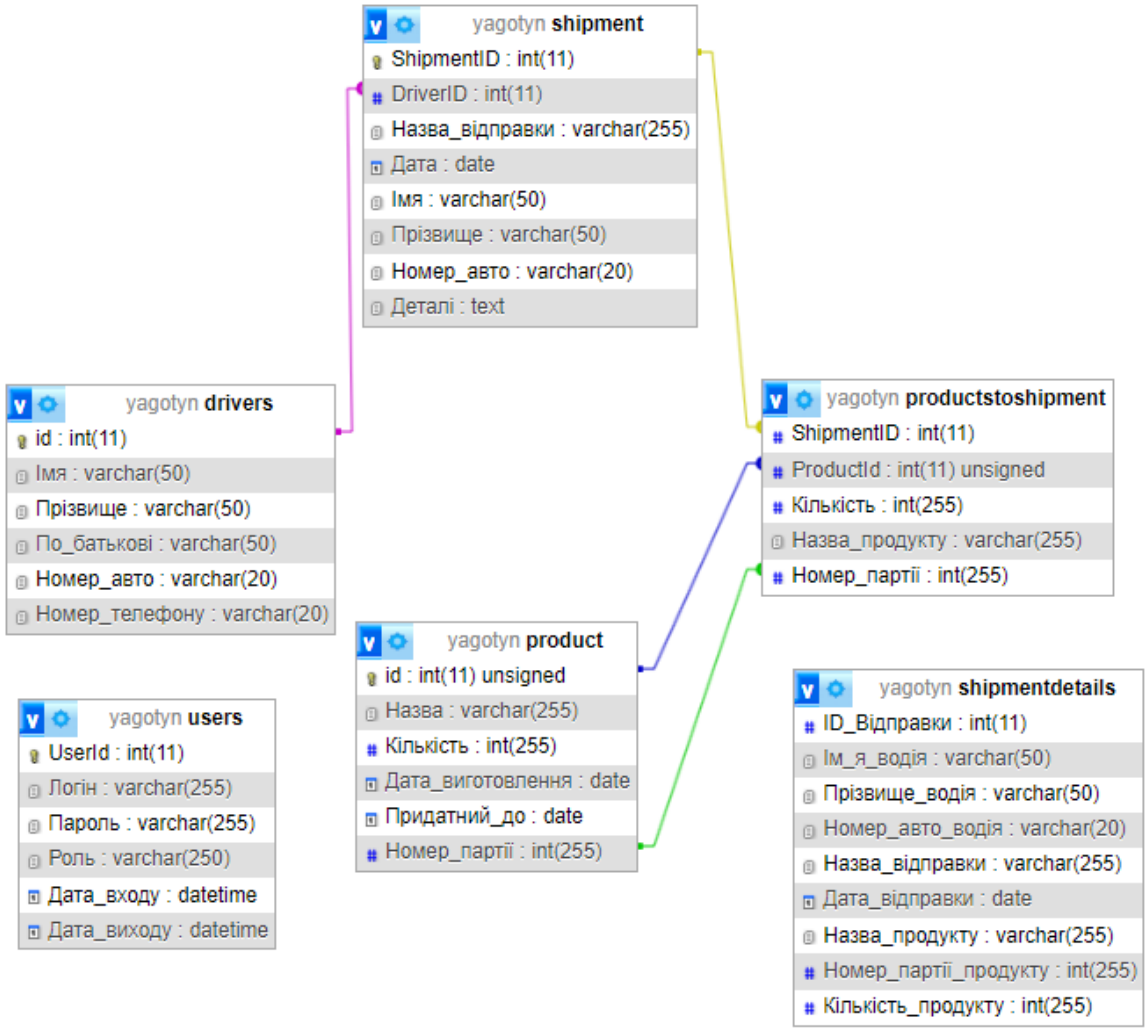

Рисунок В.1 - Побудована схема БД

ДОДАТОК Г. Код програми

Код вікна авторизації:

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Threading;
namespace DIPLOMFINAL
{
   public partial class LoginForm : Form
   {
      bool dragging = false;
      Point dragCursorPoint;
      Point dragFormPoint;
      public LoginForm()
      {
         InitializeComponent();
      }
      public string login, password,role;
      Thread th;
      private void panel2_MouseUp(object sender, MouseEventArgs e)
      {
         dragging = false;
      }

      private void panel2_MouseMove(object sender, MouseEventArgs e)
      {
         if (dragging)
         {
            Point dif = Point.Subtract(Cursor.Position, new Size(dragCursorPoint));
            this.Location = Point.Add(dragFormPoint, new Size(dif));
         }
      }
      private void panel2_MouseDown(object sender, MouseEventArgs e)
      {
         dragging = true;
         dragCursorPoint = Cursor.Position;
         dragFormPoint = this.Location;
      }

      private void button2_Click(object sender, EventArgs e)
      {
         Close();
```

```
    }

    private void button6_Click(object sender, EventArgs e)
    {

        this.WindowState = FormWindowState.Minimized;
    }

    private void LoginForm_Load(object sender, EventArgs e)
    {

        FormAnimation.Slide(this, 100, FormAnimation.SlideDirection.Down);
    }


    private void LoginForm_Click(object sender, EventArgs e)
    {

    }

    private void button1_Click(object sender, EventArgs e)
    {

        login = LoginBox.Text;
        password = PassBox.Text;
        DB db = new DB();
        DataTable table = new DataTable();
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        MySqlCommand cmd = new MySqlCommand("SELECT Логін, Пароль, Роль FROM `users` WHERE Логін = @log AND Пароль = @pass", db.getConnection());

        cmd.Parameters.Add("@log", MySqlDbType.VarChar).Value = login;
        cmd.Parameters.Add("@pass", MySqlDbType.VarChar).Value = password;

        adapter.SelectCommand = cmd;
        adapter.Fill(table);


        if (table.Rows.Count > 0)
        {
            MySqlCommand cmd2 = new MySqlCommand("UPDATE `users` SET `Дата_входу` = @date WHERE Логін = @log AND Пароль = @pass", db.getConnection());
            cmd2.Parameters.Add("@log", MySqlDbType.VarChar).Value = login;
            cmd2.Parameters.Add("@pass", MySqlDbType.VarChar).Value = password;
            cmd2.Parameters.Add("@date", MySqlDbType.DateTime).Value = DateTime.Now;

            db.openConnection();
            cmd2.ExecuteNonQuery();
            db.closeConnection();

            if (table.Rows[0]["Роль"] != DBNull.Value)
```

```
            {
                role = table.Rows[0]["Роль"].ToString();
                this.Close();
                th = new Thread(openForm);
                th.SetApartmentState(ApartmentState.STA);
                th.Start();
            }
            else
            {
                MessageBox.Show("Помилка входу");
            }
        }
        else
        {
            MessageBox.Show("Невірні дані для входу");
        }



    }
    public void openForm()
    {
        if(login != null || password != null || role !=null)
        {
            Application.Run(new MainForm(login, password, role));
        }
        else
        {
            MessageBox.Show("Помилка входу");
        }

    }
  }
}
```

Код вікна головного меню:

```
using Microsoft.ReportingServices.ReportProcessing.OnDemandReportObjectModel;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics.Eventing.Reader;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using WinFormsApp3;
```

```
namespace DIPLOMFINAL
{
    public partial class MainForm: Form
    {
        AddDelete add = new AddDelete();
        private string login;
        private string password;
        private string role;
        public MainForm(string login, string password, string role)
        {
            InitializeComponent();
            this.login = login;
            this.password = password;
            this.role = role;
            passwordChange1.oldlogin= login;
        }
        bool dragging = false;
        Point dragCursorPoint;
        Point dragFormPoint;
        private void button1_Click(object sender, EventArgs e)
        {
            BackPanel.BringToFront();
            addItemControl1.BringToFront();
            SidePanel.Height = AddItem.Height;
            SidePanel.Top = AddItem.Top;
        }

        private void DeleteItem_Click(object sender, EventArgs e)
        {
            BackPanel.BringToFront();
            deleteProduct1.Update();
            deleteProduct1.RefreshData();
            deleteProduct1.BringToFront();

            SidePanel.Height = DeleteItem.Height;
            SidePanel.Top = DeleteItem.Top;
        }

        private void ShipmentButton_Click(object sender, EventArgs e)
        {


            ShipmentForm shipment = new ShipmentForm();
            shipment.ShowDialog();
        }

        private void button4_Click(object sender, EventArgs e)
        {

            SidePanel.Height = button4.Height;
            SidePanel.Top = button4.Top;
```

```
}


private void MainForm_Load(object sender, EventArgs e)
{

   switch (role)
   {
      case "Адмін":
         {
            UsersButton.Visible = true;
            BackuButton.Visible = true;
            users1.Visible = true;
            BackPanel.BringToFront();
            addItemControl1.BringToFront();
            SidePanel.Height = AddItem.Height;
            SidePanel.Top = AddItem.Top;
            break;
         }
      case "Комплектувальник":
         {
            BackPanel.BringToFront();
            button2.Top=AddItem.Top;
            button3.Top = DeleteItem.Top;
            button4.Top = EditProductButton.Top;
            AddItem.Visible = false;
            DeleteItem.Visible = false;
            EditProductButton.Visible = false;
            break;
         }
      case "Користувач":
         {
            BackPanel.BringToFront();
            addItemControl1.BringToFront();
            SidePanel.Height = AddItem.Height;
            SidePanel.Top = AddItem.Top;
            break;
         }
   }


}

private void button1_Click_1(object sender, EventArgs e)
{

  Application.Exit();
}

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

```
    BackPanel.BringToFront();
    editProducts1.BringToFront();
    editProducts1.Parent.Update();
    editProducts1.RefreshData();

    SidePanel.Height = EditProductButton.Height;
    SidePanel.Top = EditProductButton.Top;
}

private void button2_Click(object sender, EventArgs e)
{
    ShipmentReport shipmentReport = new ShipmentReport();
    shipmentReport.ShowDialog();
    BackPanel.BringToFront();


}

private void button3_Click(object sender, EventArgs e)
{
    BackPanel.BringToFront();

    DateReport dateReport = new DateReport();
    dateReport.ShowDialog();

}

private void button5_Click(object sender, EventArgs e)
{
    if(this.WindowState == FormWindowState.Maximized)
    {
        this.WindowState = FormWindowState.Normal;
        this.StartPosition = FormStartPosition.CenterScreen;
    }
    else
    {
        this.WindowState = FormWindowState.Maximized;
    }
}

private void button6_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}

private void panel2_MouseDoubleClick(object sender, MouseEventArgs e)
{
    if(this.WindowState == FormWindowState.Maximized)
    {
        this.WindowState = FormWindowState.Normal;
        this.StartPosition = FormStartPosition.CenterScreen;
    }
```

```
    else
    {
        this.WindowState = FormWindowState.Maximized;
    }
}

private void panel2_MouseDown(object sender, MouseEventArgs e)
{
    dragging = true;
    dragCursorPoint = Cursor.Position;
    dragFormPoint = this.Location;
}

private void panel2_MouseUp(object sender, MouseEventArgs e)
{
    dragging = false;
}

private void panel2_MouseMove(object sender, MouseEventArgs e)
{
    if (dragging)
    {
        Point dif = Point.Subtract(Cursor.Position, new Size(dragCursorPoint));
        this.Location = Point.Add(dragFormPoint, new Size(dif));
    }
}

private void button1_MouseEnter(object sender, EventArgs e)
{
    button1.BackColor = Color.Red;
}

private void button1_MouseLeave(object sender, EventArgs e)
{
    button1.BackColor = Color.Transparent;
}

private void panel2_Paint(object sender, PaintEventArgs e)
{

}

private void UsersButton_Click(object sender, EventArgs e)
{
    BackPanel.BringToFront();
    users1.BringToFront();
    SidePanel.Height = UsersButton.Height;
    SidePanel.Top = UsersButton.Top;
}

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

```
        BackPanel.BringToFront();
        backupDataBase1.BringToFront();
        SidePanel.Height = BackuButton.Height;
        SidePanel.Top = BackuButton.Top;
    }

    private void MainForm_FormClosing(object sender, FormClosingEventArgs e)
    {
        DB db = new DB();
        MySqlCommand cmd2 = new MySqlCommand("UPDATE `users` SET `Дата_виходу` =
@date WHERE Логін = @log AND Пароль = @pass", db.getConnection());
        cmd2.Parameters.Add("@log", MySqlDbType.VarChar).Value = this.login;
        cmd2.Parameters.Add("@pass", MySqlDbType.VarChar).Value = this.password;
        cmd2.Parameters.Add("@date", MySqlDbType.DateTime).Value = DateTime.Now;

        db.openConnection();
        cmd2.ExecuteNonQuery();
        db.closeConnection();
        Application.Exit();
    }

    private void label1_Click(object sender, EventArgs e)
    {
        BackPanel.BringToFront();
        passwordChange1.BringToFront();

    }

    private void timer1_Tick(object sender, EventArgs e)
    {

        try
        {
            string backupFile = "backup.sql";

            DB db = new DB();

            MySqlCommand cmd = new MySqlCommand();
            cmd.Connection = db.getConnection();
            MySqlBackup mb = new MySqlBackup(cmd);
            db.openConnection();
            mb.ExportToFile(backupFile);
            db.closeConnection();


        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка під час резервного копіювання: " + ex.Message);
        }
    }
}
```

```
}
```

Код меню відвантажень:

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

namespace DIPLOMFINAL
{
   public partial class ShipmentForm : Form
   {
      public ShipmentForm()
      {
         InitializeComponent();
      }
      bool dragging = false;
      Point dragCursorPoint;
      Point dragFormPoint;
      private void button1_Click(object sender, EventArgs e)
      {
         Close();
      }

      private void AddItem_Click(object sender, EventArgs e)
      {
         createShipment1.RefreshData();
         createShipment1.BringToFront();

         SidePanel.Height = AddItem.Height;
         SidePanel.Top = AddItem.Top;
      }



      private void AddDriver_Click(object sender, EventArgs e)
      {
         driversControl1.RefreshData();
         driversControl1.BringToFront();
         SidePanel.Height = AddDriver.Height;
         SidePanel.Top = AddDriver.Top;
      }
```

```
private void button2_Click(object sender, EventArgs e)
{
   productsToShipmentControl1.RefreshData();
   productsToShipmentControl1.BringToFront();
   SidePanel.Height = button2.Height;
   SidePanel.Top = button2.Top;
}

private void ShipmentForm_Load(object sender, EventArgs e)
{
   FormAnimation.Slide(this, 100, FormAnimation.SlideDirection.Down);
   createShipment1.BringToFront();
}

private void panel2_MouseDoubleClick(object sender, MouseEventArgs e)
{
   if (this.WindowState == FormWindowState.Maximized)
   {
      this.WindowState = FormWindowState.Normal;
      this.StartPosition = FormStartPosition.CenterScreen;
   }
   else
   {
      this.WindowState = FormWindowState.Maximized;
   }
}

private void panel2_MouseDown(object sender, MouseEventArgs e)
{
   dragging = true;
   dragCursorPoint = Cursor.Position;
   dragFormPoint = this.Location;
}

private void panel2_MouseUp(object sender, MouseEventArgs e)
{
   dragging = false;
}

private void panel2_MouseMove(object sender, MouseEventArgs e)
{
   if (dragging)
   {
      Point dif = Point.Subtract(Cursor.Position, new Size(dragCursorPoint));
      this.Location = Point.Add(dragFormPoint, new Size(dif));
   }
}

private void button1_MouseEnter(object sender, EventArgs e)
{
   button1.BackColor = Color.Red;
}
```

```
        private void button1_MouseLeave(object sender, EventArgs e)
        {
            button1.BackColor = Color.Transparent;
        }
    }
}
```

Код вікна відображень всієї продукції:

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

namespace DIPLOMFINAL
{
    public partial class AllProductsForm : Form
    {
        ShowTable show = new ShowTable();
        public AllProductsForm()
        {
            InitializeComponent();
        }
        public void itemSearch(DataGridView DataTable, TextBox searchbox)
        {
            DB db = new DB();
            DataTable table = new DataTable();
            MySqlCommand command = new MySqlCommand("SELECT * FROM product WHERE
Назва LIKE @SearchText;", db.getConnection());
            command.Parameters.AddWithValue("@SearchText", "%" + searchbox.Text + "%");
            MySqlDataAdapter adapter = new MySqlDataAdapter();
            adapter.SelectCommand = command;
            adapter.Fill(table);
            DataTable.DataSource = table;
            db.closeConnection();
        }
        private void AllProductsForm_Load(object sender, EventArgs e)
        {
            FormAnimation.Slide(this, 100, FormAnimation.SlideDirection.Down);
            dataTable.DataSource = show.ShowInfo("product", "Назва, Кількість,
Дата_виготовлення, Придатний_до, Номер_партії");
        }
```

```
        private void button1_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void textBox1_TextChanged(object sender, EventArgs e)
        {
            itemSearch(dataTable, SearchBox);
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Код вікна створення звіту по відвантаження:

```
using Microsoft.Reporting.WinForms;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DIPLOMFINAL
{
    public partial class ShipmentReport : Form
    {
        private IList<ShipmentList> _shipmentList;
        public ShipmentReport()
        {
            InitializeComponent();
            fillCombo();



        }
        DB db = new DB();
        bool dragging = false;
        Point dragCursorPoint;
        Point dragFormPoint;
        DataTable table = new DataTable();
        public void FillGridView()
```

```
    {
        DB db = new DB();
        table.Clear();
        table.Rows.Clear();
        MySqlCommand command = new MySqlCommand("SELECT ID_Відправки, Ім_я_водія, Прізвище_водія, Номер_авто_водія, Назва_Відправки, Дата_відправки, Назва_продукту, Номер_партії_продукту, Кількість_продукту FROM shipmentdetails WHERE ID_Відправки = @ID", db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        command.Parameters.AddWithValue("@ID", comboBox1.SelectedValue);
        adapter.SelectCommand = command;
        adapter.Fill(table);
       // dataGridView1.DataSource = table;
       // dg.DataSource = table;
      db.closeConnection();

    }

    public void fillCombo()
    {
        string query = "SELECT ID_Відправки, CONCAT(Назва_відправки, ', ', Номер_авто_водія) AS DisplayText FROM shipmentdetails GROUP BY ID_Відправки";


        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();

        try
        {
            db.openConnection();
            adapter.Fill(table);

            comboBox1.DisplayMember = "DisplayText";
            comboBox1.ValueMember = "ID_Відправки";
            comboBox1.DataSource = table;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }




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

```
    FormAnimation.Slide(this, 100, FormAnimation.SlideDirection.Down);


}



private void button1_Click(object sender, EventArgs e)
{
    FillGridView();
    reportViewer1.LocalReport.ReportEmbeddedResource = "DIPLOMFINAL.Report1.rdlc";
    var ds = new ReportDataSource("DataSetReport", table);
    reportViewer1.LocalReport.DataSources.Add(ds);
    reportViewer1.RefreshReport();
}

private void comboBox1_SelectedValueChanged(object sender, EventArgs e)
{

}

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

private void button5_Click(object sender, EventArgs e)
{
    if (this.WindowState == FormWindowState.Maximized)
    {
        this.WindowState = FormWindowState.Normal;
        this.StartPosition = FormStartPosition.CenterScreen;
    }
    else
    {
        this.WindowState = FormWindowState.Maximized;
    }
}

private void button6_Click(object sender, EventArgs e)
{
    this.WindowState = FormWindowState.Minimized;
}

private void panel1_MouseDoubleClick(object sender, MouseEventArgs e)
{
    if (this.WindowState == FormWindowState.Maximized)
    {
        this.WindowState = FormWindowState.Normal;
        this.StartPosition = FormStartPosition.CenterScreen;
    }
    else
```

```
        {
            this.WindowState = FormWindowState.Maximized;
        }
    }

    private void panel1_MouseDown(object sender, MouseEventArgs e)
    {
        dragging = true;
        dragCursorPoint = Cursor.Position;
        dragFormPoint = this.Location;
    }

    private void panel1_MouseUp(object sender, MouseEventArgs e)
    {
        dragging = false;
    }

    private void panel1_MouseMove(object sender, MouseEventArgs e)
    {
        if (dragging)
        {
            Point dif = Point.Subtract(Cursor.Position, new Size(dragCursorPoint));
            this.Location = Point.Add(dragFormPoint, new Size(dif));
        }
    }

    private void button2_MouseEnter(object sender, EventArgs e)
    {
        button2.BackColor = Color.Red;
    }

    private void button2_MouseLeave(object sender, EventArgs e)
    {
        button2.BackColor = Color.Transparent;
    }
  }
}
```

Код вікна створення звіту по даті виробництва:

```
using Microsoft.Reporting.WinForms;
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace DIPLOMFINAL
{
    public partial class DateReport : Form
    {
        DataTable table = new DataTable();
        DB db = new DB();
        bool dragging = false;
        Point dragCursorPoint;
        Point dragFormPoint;
        public DateReport()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Close();
        }

        private void DateReport_Load(object sender, EventArgs e)
        {
            FormAnimation.Slide(this, 100, FormAnimation.SlideDirection.Down);
        }

        private void button2_Click(object sender, EventArgs e)
        {
            FillGridView();
            reportViewer1.LocalReport.ReportEmbeddedResource = "DIPLOMFINAL.Report2.rdlc";
            var ds = new ReportDataSource("ReportProductSet1", table);
            reportViewer1.LocalReport.DataSources.Add(ds);
            reportViewer1.RefreshReport();
        }

        public void FillGridView()
```

```
    {
        DB db = new DB();
        table.Clear();
        table.Rows.Clear();
        string formattedDate = dateTimePicker1.Value.ToString("yyyy-MM-dd");
        MySqlCommand command = new MySqlCommand("SELECT * FROM product WHERE
Дата_виготовлення = @Date", db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        command.Parameters.AddWithValue("@Date", formattedDate);
        adapter.SelectCommand = command;
        adapter.Fill(table);
        // dataGridView1.DataSource = table;
        // dg.DataSource = table;
        db.closeConnection();

    }

    private void button5_Click(object sender, EventArgs e)
    {
        if (this.WindowState == FormWindowState.Maximized)
        {
            this.WindowState = FormWindowState.Normal;
            this.StartPosition = FormStartPosition.CenterScreen;
        }
        else
        {
            this.WindowState = FormWindowState.Maximized;
        }
    }

    private void button6_Click(object sender, EventArgs e)
    {
        this.WindowState = FormWindowState.Minimized;
    }

    private void panel2_MouseDoubleClick(object sender, MouseEventArgs e)
    {
        if (this.WindowState == FormWindowState.Maximized)
        {
            this.WindowState = FormWindowState.Normal;
            this.StartPosition = FormStartPosition.CenterScreen;
        }
        else
        {
            this.WindowState = FormWindowState.Maximized;
        }
    }

    private void panel2_MouseDown(object sender, MouseEventArgs e)
    {
        dragging = true;
        dragCursorPoint = Cursor.Position;
```

```
            dragFormPoint = this.Location;
        }

        private void panel2_MouseUp(object sender, MouseEventArgs e)
        {
            dragging = false;
        }

        private void panel2_MouseMove(object sender, MouseEventArgs e)
        {
            if (dragging)
            {
                Point dif = Point.Subtract(Cursor.Position, new Size(dragCursorPoint));
                this.Location = Point.Add(dragFormPoint, new Size(dif));
            }
        }

        private void button1_MouseEnter(object sender, EventArgs e)
        {
            button1.BackColor = Color.Red;
        }

        private void button1_MouseLeave(object sender, EventArgs e)
        {
            button1.BackColor = Color.Transparent;
        }
    }
}
```

Код вкладки додавання продукції:

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using WinFormsApp3;

namespace DIPLOMFINAL
{
    public partial class AddItemControl : UserControl
    {
```

```
public AddItemControl()
{
    InitializeComponent();

}
DB connection = new DB();
public void addItem(string newName, string NewCount, DateTime cdate, DateTime bbd)
{
    try
    {
        Random rnd = new Random();
        int minValue = 100; // Мінімальне значення діапазону
        int maxValue = 1000; // Максимальне значення діапазону (виключно)
        int randomNumber = rnd.Next(minValue, maxValue);

        using (MySqlCommand command = new MySqlCommand("INSERT INTO product (Назва, Кількість, Дата_виготовлення, Придатний_до, Номер_партії) VALUES (@NewName, @NewCount, @Newcdate,@Newbbd, @NewNumpart)", connection.getConnection()))
        {
            command.Parameters.AddWithValue("@NewName", newName);
            command.Parameters.AddWithValue("@NewCount", NewCount);
            command.Parameters.AddWithValue("@Newcdate", cdate);
            command.Parameters.AddWithValue("@Newbbd", bbd);
            command.Parameters.AddWithValue("@NewNumpart", randomNumber);
            connection.openConnection();
            command.ExecuteNonQuery();
        }

        // LoadData(mainForm.DataGridView1); // Assuming DataGridView in MainForm is called DataGridView1
    }
    catch (Exception ex)
    {
        MessageBox.Show("Помилка при додаванні" + ex.Message);
    }
    finally
    {
        connection.closeConnection();
    }
}



private void QuantityItem_KeyPress(object sender, KeyPressEventArgs e)
{

    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
    {
        e.Handled = true;
    }
}
```

```
        private void button1_Click(object sender, EventArgs e)
        {

            string Name = Nameitem.Text;
            string Quant = QuantityItem.Text;
            DateTime fdate = cdate.Value;
            DateTime bdate = bbd.Value;
            if (!string.IsNullOrEmpty(Name) && !string.IsNullOrEmpty(Quant))
            {
                addItem(Name, Quant, fdate, bdate);
                Nameitem.Clear();
                QuantityItem.Clear();
            }
        }

    }

}
```

Код вкладки видалення продукції:

```
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections;
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 DIPLOMFINAL
{
    public partial class DeleteProduct : UserControl
    {



        public DeleteProduct()
        {
            InitializeComponent();
        }
        private BindingSource bindingSource2 = new BindingSource();
        DB db = new DB();
        DataTable table = new DataTable();
        public void itemSearch(DataGridView DataTable, TextBox searchbox)
```

```
    {
        DB db = new DB();
        DataTable table = new DataTable();
        MySqlCommand command = new MySqlCommand("SELECT * FROM product WHERE
Назва LIKE @SearchText;", db.getConnection());
        command.Parameters.AddWithValue("@SearchText", "%" + searchbox.Text + "%");
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        DataTable.DataSource = table;
        db.closeConnection();
    }
    private void SearchBox_TextChanged(object sender, EventArgs e)
    {
        itemSearch(dataGridView1, SearchBox);
    }

    private void DeleteProduct_Load(object sender, EventArgs e)
    {
        DB db = new DB();

        MySqlCommand command = new MySqlCommand("SELECT * FROM product",
db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        dataGridView1.DataSource = table;
        dataGridView1.Columns[0].Visible = false;
        db.closeConnection();
    }

    private void DeleteRecordFromDatabase(int id)
    {
        db.openConnection();
        try
        {
            string query = "DELETE FROM `product` WHERE id = @id";
            using (MySqlCommand command = new MySqlCommand(query, db.getConnection()))
            {
                command.Parameters.AddWithValue("@id", id);
                command.ExecuteNonQuery();
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка при видаленні запису: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }
```

```
public void LoadData(DataGridView DeleteView)
{
    try
    {
        db.openConnection();

        using (MySqlCommand cmd = new MySqlCommand("SELECT * FROM product", db.getConnection()))
        {
            DataTable dt = new DataTable();
            using (MySqlDataAdapter adapter = new MySqlDataAdapter(cmd))
            {
                adapter.Fill(dt);
                DeleteView.DataSource = dt;
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Помилка при завантаженні даних: " + ex.Message);
    }
    finally
    {
        db.closeConnection();
    }
}

public void DeleteItem(DataGridView DeleteView)
{
    if (DeleteView.SelectedRows.Count > 0)
    {
        int id = Convert.ToInt32(DeleteView.SelectedRows[0].Cells["id"].Value);

        DeleteRecordFromDatabase(id);
        LoadData(DeleteView);
    }
    else
    {
        MessageBox.Show("Будь ласка, виберіть рядок для видалення.");
    }
}

private void button1_Click(object sender, EventArgs e)
{
    //DeleteItem(dataGridView1);
}

public void sortBbd()
{
```

```
        bindingSource2.Filter = "Придатний_до <= #" + DateTime.Now.ToString("yyyy-MM-dd") +
"#";
    }

    public void fillTable()
    {
        string query = "SELECT * FROM product ";

        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();
        table.Clear();
        table.Rows.Clear();
        try
        {
            db.openConnection();
            adapter.Fill(table);

            DataView dataView = new DataView(table);
            dataGridView1.DataSource = dataView;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }


    private void checkBox1_CheckedChanged(object sender, EventArgs e)
    {


        if (checkBox1.Checked)
        {
            string query = "SELECT * FROM product WHERE Придатний_до < CURDATE()";

            MySqlCommand command = new MySqlCommand(query, db.getConnection());
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
            DataTable table = new DataTable();
            table.Clear();
            table.Rows.Clear();
            try
            {
                db.openConnection();
                adapter.Fill(table);

                DataView dataView = new DataView(table);
                dataGridView1.DataSource = dataView;
```

```
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка: " + ex.Message);
            }
            finally
            {
                db.closeConnection();
            }


        }
        else
        {
            fillTable();
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {

        string query = "DELETE FROM product " +
                       "WHERE Кількість = 0;";

        db.openConnection();
        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        int deletedRowsCount = command.ExecuteNonQuery();
        db.closeConnection();

        MessageBox.Show("Кількість затронутих записів: " + deletedRowsCount);
        fillTable();


    }


    public void RefreshData()
    {

        dataGridView1.DataSource = null;
        dataGridView1.Rows.Clear();
        dataGridView1.Columns.Clear();
        DataTable table2 = new DataTable();
        table2.Clear();
        table2.Rows.Clear();
        DB db = new DB();

        MySqlCommand command = new MySqlCommand("SELECT * FROM product",
db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table2);
```

```
            dataGridView1.DataSource = table2;
            dataGridView1.Columns[0].Visible = false;
            db.closeConnection();


        }

        private void dataGridView1_UserDeletingRow(object sender,
DataGridViewRowCancelEventArgs e)
        {

            // DialogResult result = MessageBox.Show("Ви дійсно бажаєте видалити цей запис?",
"Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
            // if (result == DialogResult.Yes)
             //{
                DeleteItem(dataGridView1);
             //}
            // else
            // {
               // e.Cancel = true;
            // }


        }

        private void button1_Click_1(object sender, EventArgs e)
        {
           DeleteItem(dataGridView1);
        }
    }
}
```

Код вкладки редагування продукції:

```
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace DIPLOMFINAL
{
    public partial class EditProducts : UserControl
```

```
{
    DB db = new DB();
    public EditProducts()
    {
        InitializeComponent();
    }
    public void fillCombo()
    {
        string query = "SELECT id, CONCAT(Назва, ', Номер партії: ', Номер_партії) AS DisplayText FROM product";


        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();

        try
        {
            db.openConnection();
            adapter.Fill(table);

            comboBox1.DisplayMember = "DisplayText";
            comboBox1.ValueMember = "id";
            comboBox1.DataSource = table;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }
    private void EditProducts_Load(object sender, EventArgs e)
    {

        fillCombo();

    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        if (comboBox1.SelectedValue != null)
        {
            string selectedId = comboBox1.SelectedValue.ToString();
            string query = "SELECT Назва, Кількість, Дата_виготовлення, Придатний_до, Номер_партії " +
                           "FROM product WHERE id = @id";


            MySqlCommand command = new MySqlCommand(query, db.getConnection());
```

```
            command.Parameters.AddWithValue("@id", selectedId);
            MySqlDataAdapter adapter = new MySqlDataAdapter(command);
            DataTable table = new DataTable();

            try
            {
                db.openConnection();
                adapter.Fill(table);

                if (table.Rows.Count > 0)
                {
                    Nameitem.Text = table.Rows[0]["Назва"].ToString();
                    QuantityBox.Text = table.Rows[0]["Кількість"].ToString();
                    if (DateTime.TryParse(table.Rows[0]["Дата_виготовлення"].ToString(), out
DateTime Дата_виготовлення))
                    {
                        dateTimePicker1.Value = Дата_виготовлення;
                    }
                    if (DateTime.TryParse(table.Rows[0]["Придатний_до"].ToString(), out DateTime
Придатний_до))
                    {
                        dateTimePicker2.Value = Придатний_до;
                    }
                    NumBox.Text = table.Rows[0]["Номер_партії"].ToString();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка: " + ex.Message);
            }
            finally
            {
                db.closeConnection();
            }
        }
    }




    private void button1_Click(object sender, EventArgs e)
    {
        string query = "UPDATE product " +
            "SET Назва = @Назва, " +
            "Кількість = @Кількість, " +
            "Дата_виготовлення = @Дата_виготовлення, " +
            "Придатний_до = @Придатний, " +
            "Номер_партії = @Номер_партії " +
            "WHERE id = @id";
```

```
    MySqlCommand command = new MySqlCommand(query, db.getConnection());

    try
    {
        db.openConnection();

        command.Parameters.AddWithValue("@Назва", Nameitem.Text);
        command.Parameters.AddWithValue("@Кількість", QuantityBox.Text);
        command.Parameters.AddWithValue("@Дата_виготовлення", dateTimePicker1.Value);
        command.Parameters.AddWithValue("@Придатний", dateTimePicker2.Value);
        command.Parameters.AddWithValue("@Номер_партії", NumBox.Text);
        command.Parameters.AddWithValue("@id", comboBox1.SelectedValue);

        int rowsAffected = command.ExecuteNonQuery();

        if (rowsAffected > 0)
        {
            MessageBox.Show("Запис успішно оновлено");
        }
        else
        {
            MessageBox.Show("Запис не було оновлено");
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Помилка: " + ex.Message);
    }
    finally
    {
        db.closeConnection();
    }
}

private void NumBox_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
    {
        e.Handled = true;
    }
}

private void QuantityBox_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
    {
        e.Handled = true;
    }
}
public void RefreshData()
```

```
        {
            Nameitem.Text = null;
            QuantityBox.Text = null;
            NumBox.Text = null;
            comboBox1.DisplayMember =null;
            comboBox1.ValueMember = null;
            comboBox1.DataSource = null;
            fillCombo();
        }
    }
    }
```

Код вкладинки користувачі:

```
using Microsoft.ReportingServices.RdlExpressions.ExpressionHostObjectModel;
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DIPLOMFINAL
{
    public partial class Users : UserControl
    {
        public string role;
        public Users()
        {
            InitializeComponent();
        }
        DB db = new DB();
        private void FillUserTable()
        {
            DataTable table = new DataTable();
            table.Rows.Clear();
            table.Clear();
            MySqlCommand command = new MySqlCommand("SELECT Логін, Пароль, Роль,
Дата_входу, Дата_виходу FROM users", db.getConnection());
            MySqlDataAdapter adapter = new MySqlDataAdapter();
            adapter.SelectCommand = command;
            adapter.Fill(table);
            table.Columns["Дата_входу"].DataType = typeof(DateTime);
            table.Columns["Дата_виходу"].DataType = typeof(DateTime);
```

```
        dataGridView1.DataSource = table;
        db.closeConnection();
    }
    private void Users_Load(object sender, EventArgs e)
    {
        FillUserTable();
    }

    private void SearchBox_TextChanged(object sender, EventArgs e)
    {
        DataTable table = new DataTable();
        MySqlCommand command = new MySqlCommand("SELECT Логін, Пароль, Роль,
Дата_входу, Дата_виходу FROM users WHERE Логін LIKE @SearchText;", db.getConnection());
        command.Parameters.AddWithValue("@SearchText", "%" + SearchBox.Text + "%");
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConnection();
    }
    private string GetUserLogin(DataGridViewRow row)
    {

        string login = string.Empty;
        if (row.Cells["Логін"].Value != null && row.Cells["Роль"].Value != null)
        {
            login = row.Cells["Логін"].Value.ToString();
            role = row.Cells["Роль"].Value.ToString();
        }
        return login;
    }

    private void DeleteUserByLogin(string Login)
    {

        string query = "DELETE FROM users WHERE Логін = @log";


        {
            MySqlCommand command = new MySqlCommand(query, db.getConnection());
            command.Parameters.AddWithValue("@log", Login);

            try
            {
                db.openConnection();
                int rowsAffected = command.ExecuteNonQuery();
                if (rowsAffected > 0)
                {
                    MessageBox.Show("Користувач був видалений.");
                }
                else
                {
```

```
                MessageBox.Show("Помилка видалення.");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
    }
}


    private void dataGridView1_UserDeletingRow(object sender, DataGridViewRowCancelEventArgs
e)
    {
        DialogResult result = MessageBox.Show("Ви дійсно бажаєте видалити цей запис?",
"Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            int rowIndex = e.Row.Index;
            DataGridViewRow row = dataGridView1.Rows[rowIndex];

            string login = GetUserLogin(row);
            if(role == "Адмін")
            {
                MessageBox.Show("Неможливо видалити цей запис");

                return;
            }
            else
            {
                if (!string.IsNullOrEmpty(login))
                {
                    DeleteUserByLogin(login);
                }
                else
                {
                    MessageBox.Show("Не вдалося отримати номер телефону водія.");
                }
            }
            FillUserTable();
        }
        else
        {
            e.Cancel = true;
        }

    }

    private void checkBox1_CheckedChanged(object sender, EventArgs e)
    {
        if(checkBox1.Checked)
```

```
        {
            PassBox.UseSystemPasswordChar = false;
        }
        else
        {
            PassBox.UseSystemPasswordChar = true;
        }
    }

    private void button2_Click(object sender, EventArgs e)
    {
        string Login = LoginBox.Text;
        string Password = PassBox.Text;
        string role = RoleCombo.Text;

        string query = "INSERT INTO users (Логін, Пароль, Роль) VALUES (@Логін, @Пароль, @Роль)";
        string checkQuery = "SELECT * FROM users WHERE Логін = @login";
        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        command.Parameters.AddWithValue("@Логін", Login);
        command.Parameters.AddWithValue("@Пароль", Password);
        command.Parameters.AddWithValue("@Роль", role);
        if (string.IsNullOrWhiteSpace(LoginBox.Text) || string.IsNullOrWhiteSpace(PassBox.Text) || string.IsNullOrWhiteSpace(RoleCombo.Text))
        {
            MessageBox.Show("Заповніть всі поля");
        }
        else
        {
            try
            {
                db.openConnection();
                DataTable table2 = new DataTable();
                MySqlCommand checkCommand = new MySqlCommand(checkQuery, db.getConnection());

                MySqlDataAdapter adapterCheck = new MySqlDataAdapter();
                checkCommand.Parameters.AddWithValue("@login", Login);

                adapterCheck.SelectCommand = checkCommand;
                adapterCheck.Fill(table2);
                if (table2.Rows.Count > 0)
                {
                    MessageBox.Show("Такий користувач існує");
                }
                else
                {

                    db.openConnection();
                    command.ExecuteNonQuery();
                    MessageBox.Show("Додано успішно");
                    LoginBox.Clear();
```

```
                PassBox.Clear();
                dataGridView1.DataSource = null;
                dataGridView1.Rows.Clear();
                FillUserTable();
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка при додаванні " + ex.Message);
        }
    }
    }


    private void button1_Click(object sender, EventArgs e)
    {

    }

  }
}
```

Код вкладки імпорту та експорту бд:

```
using MySql.Data.MySqlClient;

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

namespace DIPLOMFINAL
{
  public partial class BackupDataBase : UserControl
  {
    public BackupDataBase()
    {
      InitializeComponent();
    }
    DB db = new DB();
    private void BackupDataBase_Load(object sender, EventArgs e)
    {

    }
```

```
        private void btnBackup_Click(object sender, EventArgs e)
        {
            try
            {
                string backupFile = "backup.sql";

                DB db = new DB();

                MySqlCommand cmd = new MySqlCommand();
                cmd.Connection = db.getConnection();
                MySqlBackup mb = new MySqlBackup(cmd);
                db.openConnection();
                mb.ExportToFile(backupFile);
                db.closeConnection();

                MessageBox.Show("Успіх.");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка: " + ex.Message);
            }
        }

        private void btnRestore_Click(object sender, EventArgs e)
        {
            try
            {
                string backupFile = "backup.sql";

                DB db = new DB();
                MySqlCommand cmd = new MySqlCommand();
                cmd.Connection = db.getConnection();
                MySqlBackup mb = new MySqlBackup(cmd);

                db.openConnection();
                mb.ImportFromFile(backupFile);
                db.closeConnection();

                MessageBox.Show("Відновлення пройшло успішно.");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка: " + ex.Message);
            }
        }

    }
}
```

Код вкладки створення відвантаження:

```
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement.ListView;

namespace DIPLOMFINAL
{
    public partial class CreateShipment : UserControl
    {
        public CreateShipment()
        {
            InitializeComponent();
        }
        DB db = new DB();
        int driverID;
        private void CreateShipment_Load(object sender, EventArgs e)
        {
            comboBox1.DisplayMember = null;
            comboBox1.ValueMember = null;
            comboBox1.DataSource = null;
            dataGridView1.DataSource = null;
           RefreshData();

        }
        public void RefreshData()
        {
            FillDriverCombo();
            dataGridView1.DataSource = ShowTable();
        }


        public void itemSearch()
        {

            DB db = new DB();
            DataTable table = new DataTable();
            MySqlCommand command = new MySqlCommand("SELECT Назва_відправки, Дата, Ім'я, Прізвище, Номер_авто, Деталі FROM shipment WHERE Назва_відправки LIKE @SearchText;", db.getConnection());
            command.Parameters.AddWithValue("@SearchText", "%" + SearchBox.Text + "%");
```

```
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConnection();
    }


    private void SearchBox_TextChanged(object sender, EventArgs e)
    {
        itemSearch();
    }
    public DataTable ShowTable()
    {

        DataTable table = new DataTable();
        table.Clear();
        table.Columns.Clear();
        db.openConnection();
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        MySqlCommand command = new MySqlCommand("SELECT Назва_відправки, Дата, Ім'я,
Прізвище, Номер_авто, Деталі FROM `shipment`", db.getConnection());

        adapter.SelectCommand = command;
        adapter.Fill(table);
        db.closeConnection();
        return table;
    }


    public void FillDriverCombo()
    {
        string query = "SELECT id, CONCAT(Ім'я, ', ', Прізвище, ', ', Номер_авто) AS DisplayText
FROM drivers"; ;


        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();

        try
        {
            db.openConnection();
            adapter.Fill(table);

            comboBox1.DisplayMember = "DisplayText";
            comboBox1.ValueMember = "id";
            comboBox1.DataSource = table;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
```

```
        }
        finally
        {
            db.closeConnection();
        }

    }


    private void button1_Click(object sender, EventArgs e)
    {

        string NameShip = ShipmentNameBox.Text;
        string details = DetailsBox.Text;

        if (driverID == 0 || string.IsNullOrEmpty(NameShip) || dateTimePicker1.Value == DateTime.MinValue)
        {
            MessageBox.Show("Заповніть всі поля");
        }
        else
        {
            string checkQuery = "SELECT * FROM shipment WHERE DriverId = @driverId";
            string query = "INSERT INTO shipment (DriverId, Назва_відправки, Дата, Ім'я, Прізвище, Номер_авто, Деталі) " +
                           "SELECT @driverId, @назва_відправки, @дата, d.ім'я, d.прізвище, d.номер_авто, @деталі " +
                           "FROM drivers d " +
                           "WHERE d.id = @driverId";

            try
            {
                db.openConnection();
                DataTable table2 = new DataTable();
                MySqlCommand checkCommand = new MySqlCommand(checkQuery, db.getConnection());

                MySqlDataAdapter adapterCheck= new MySqlDataAdapter();
                checkCommand.Parameters.AddWithValue("@driverId", driverID);

                adapterCheck.SelectCommand = checkCommand;
                adapterCheck.Fill(table2);

                if (table2.Rows.Count > 0)
                {
                    MessageBox.Show("Відправлення для цього водія вже існує");

                }
                else
                {
                    MySqlCommand command = new MySqlCommand(query, db.getConnection());
                    command.Parameters.AddWithValue("@driverId", driverID);
```

```
                command.Parameters.AddWithValue("@назва_відправки", NameShip);
                command.Parameters.AddWithValue("@дата", dateTimePicker1.Value);
                command.Parameters.AddWithValue("@деталі", details);

                int rowsAffected = command.ExecuteNonQuery();

                if (rowsAffected > 0)
                {
                    MessageBox.Show("Запис успішно додано");
                    dataGridView1.DataSource = null;
                    dataGridView1.DataSource = ShowTable();
                    ShipmentNameBox.Clear();
                    DetailsBox.Clear();
                }
                else
                {
                    MessageBox.Show("Помилка додавання запису");
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }



}



    private void dataGridView1_UserDeletingRow(object sender,
DataGridViewRowCancelEventArgs e)
    {
        DialogResult result = MessageBox.Show("Ви дійсно бажаєте видалити цей запис?",
"Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            int rowIndex = e.Row.Index;
            string shipmentName =
dataGridView1.Rows[rowIndex].Cells["Назва_відправки"].Value.ToString();
            string carNumber =
dataGridView1.Rows[rowIndex].Cells["Номер_авто"].Value.ToString();

            DeleteShipment(shipmentName, carNumber);
        }
        else
```

```
        {
            e.Cancel = true;
        }
    }


    private void DeleteShipment(string shipmentName, string carNumber)
    {
        string query = "DELETE FROM shipment WHERE Назва_відправки = @shipmentName AND
Номер_авто = @carNumber";
        MySqlCommand command = new MySqlCommand(query, db.getConnection());

        try
        {
            db.openConnection();

            command.Parameters.AddWithValue("@shipmentName", shipmentName);
            command.Parameters.AddWithValue("@carNumber", carNumber);

            int rowsAffected = command.ExecuteNonQuery();

            if (rowsAffected > 0)
            {
                MessageBox.Show("Запис успішно видалено");
            }
            else
            {
                MessageBox.Show("Помилка видалення запису");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        driverID = Convert.ToInt32(comboBox1.SelectedValue);
    }
  }
}
```

Код вкладки створення водія:

```
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace DIPLOMFINAL
{
   public partial class DriversControl : UserControl
   {
      public DriversControl()
      {
         InitializeComponent();
      }
      DB db = new DB();
      private void PhoneBox_KeyPress(object sender, KeyPressEventArgs e)
      {
         if(PhoneBox.TextLength>12)
         {
            PhoneBox.Clear();
            PhoneBox.Text = "+380";
            PhoneBox.SelectionStart = PhoneBox.Text.Length;
         }
         if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
         {
            e.Handled = true;
         }
         PhoneBox.SelectionStart = PhoneBox.Text.Length;



      }

      private void PhoneBox_Click(object sender, EventArgs e)
      {
         PhoneBox.Text = "+380";
         PhoneBox.SelectionStart = PhoneBox.Text.Length;
      }

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

```
        if (PhoneBox.Text.Length < "+380".Length)
        {
            PhoneBox.Text = "+380";
            PhoneBox.SelectionStart = PhoneBox.Text.Length;

        }
    }

    public void fillDataDriver()
    {

        DataTable table = new DataTable();
        MySqlCommand command = new MySqlCommand("SELECT Імя, Прізвище, По_батькові,
Номер_авто, Номер_телефону FROM drivers", db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConnection();
    }

    private void DriversControl_Load(object sender, EventArgs e)
    {
        fillDataDriver();
    }

    public void RefreshData()
    {
        dataGridView1.DataSource = null;
        fillDataDriver();
    }



    public void itemSearch()
    {

        DataTable table = new DataTable();
        MySqlCommand command = new MySqlCommand("SELECT Імя, Прізвище, По_батькові,
Номер_авто, Номер_телефону FROM drivers WHERE Номер_авто LIKE @SearchText;",
db.getConnection());
        command.Parameters.AddWithValue("@SearchText", "%" + SearchBox.Text + "%");
        MySqlDataAdapter adapter = new MySqlDataAdapter();
        adapter.SelectCommand = command;
        adapter.Fill(table);
        dataGridView1.DataSource = table;
        db.closeConnection();
    }

    private void SearchBox_TextChanged(object sender, EventArgs e)
    {
        itemSearch();
```

```
    }

    private void button1_Click(object sender, EventArgs e)
    {
        string name = NameBox.Text;
        string surname = SurnameBox.Text;
        string lastname = lastNameBox.Text;
        string carnumber = CarnumberBox.Text;
        string phone = PhoneBox.Text;
        string query = "INSERT INTO drivers (Імя, Прізвище, По_батькові, Номер_авто, Номер_телефону) VALUES (@Імя, @Прізвище, @По_батькові, @Номер_авто, @Номер_телефону)";

        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        command.Parameters.AddWithValue("@Імя", name);
        command.Parameters.AddWithValue("@Прізвище", surname);
        command.Parameters.AddWithValue("@По_батькові", lastname);
        command.Parameters.AddWithValue("@Номер_авто", carnumber);
        command.Parameters.AddWithValue("@Номер_телефону", phone);
        if (string.IsNullOrWhiteSpace(NameBox.Text) || string.IsNullOrWhiteSpace(SurnameBox.Text) || string.IsNullOrWhiteSpace(lastNameBox.Text) || string.IsNullOrWhiteSpace(CarnumberBox.Text) || string.IsNullOrWhiteSpace(PhoneBox.Text))
        {
            MessageBox.Show("Заповніть всі поля");
        }
        else
        {
            try
            {
                db.openConnection();
                command.ExecuteNonQuery();
                MessageBox.Show("Додано успішно");
                NameBox.Clear();
                lastNameBox.Clear();
                SurnameBox.Clear();
                CarnumberBox.Clear();
                PhoneBox.Clear();
                dataGridView1.DataSource = null;
                fillDataDriver();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка при додаванні " + ex.Message);
            }
        }


    }

    private void dataGridView1_UserDeletingRow(object sender, DataGridViewRowCancelEventArgs e)
    {
```

```
        DialogResult result = MessageBox.Show("Ви дійсно бажаєте видалити цей запис?", "Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            int rowIndex = e.Row.Index;
            DataGridViewRow row = dataGridView1.Rows[rowIndex];

            string phoneNumber = GetDriverPhoneNumberFromRow(row);

            if (!string.IsNullOrEmpty(phoneNumber))
            {
                DeleteDriverByPhoneNumber(phoneNumber);
            }
            else
            {
                MessageBox.Show("Не вдалося отримати номер телефону водія.");
            }
        }
        else
        {
            e.Cancel = true;
        }
    }



    private string GetDriverPhoneNumberFromRow(DataGridViewRow row)
    {

        string phoneNumber = string.Empty;
        if (row.Cells["Номер_телефону"].Value != null)
        {
            phoneNumber = row.Cells["Номер_телефону"].Value.ToString();
        }
        return phoneNumber;
    }

    private void DeleteDriverByPhoneNumber(string phoneNumber)
    {

        string query = "DELETE FROM drivers WHERE Номер_телефону = @phoneNumber";


        {
            MySqlCommand command = new MySqlCommand(query, db.getConnection());
                command.Parameters.AddWithValue("@phoneNumber", phoneNumber);

                try
                {
                    db.openConnection();
                    int rowsAffected = command.ExecuteNonQuery();
                    if (rowsAffected > 0)
```

```
                {
                    MessageBox.Show("Водій успішно видалений.");
                }
                else
                {
                    MessageBox.Show("Помилка видалення водія.");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Помилка: " + ex.Message);
            }
        }
    }
}
```

Код вкладки додавання продукції до відвантаження:

```
using MySql.Data.MySqlClient;
using MySqlX.XDevAPI.Relational;
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DIPLOMFINAL
{
   public partial class ProductsToShipmentControl : UserControl
   {
      public ProductsToShipmentControl()
      {
         InitializeComponent();
      }
      DB db = new DB();
      public void FillShipmentCombo()
      {
         string query = "SELECT ShipmentID, CONCAT(Назва_відправки, ', ', Номер_авто) AS
DisplayText FROM shipment";


         MySqlCommand command = new MySqlCommand(query, db.getConnection());
         MySqlDataAdapter adapter = new MySqlDataAdapter(command);
         DataTable table = new DataTable();

         try
         {
            db.openConnection();
            adapter.Fill(table);

            comboBox1.DisplayMember = "DisplayText";
            comboBox1.ValueMember = "ShipmentID";
            comboBox1.DataSource = table;
         }
         catch (Exception ex)
         {
            MessageBox.Show("Помилка: " + ex.Message);
         }
         finally
         {
            db.closeConnection();
         }
```

```
    }

    public void FillProductCombo()
    {
        string query = "SELECT id, CONCAT(Назва, ', Ном. партії: ', Номер_партії, ', Кількість ', Кількість) AS DisplayText FROM product";


        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        MySqlDataAdapter adapter = new MySqlDataAdapter(command);
        DataTable table = new DataTable();

        try
        {
            db.openConnection();
            adapter.Fill(table);

            comboBox2.DisplayMember = "DisplayText";
            comboBox2.ValueMember = "id";
            comboBox2.DataSource = table;
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }

    public DataTable ShowTable()
    {
        if(comboBox1.SelectedValue !=null)
        {
            string id = comboBox1.SelectedValue.ToString();
            DataTable table = new DataTable();
            table.Clear();
            table.Columns.Clear();
            db.openConnection();
            MySqlDataAdapter adapter = new MySqlDataAdapter();
            MySqlCommand command = new MySqlCommand("SELECT Кількість, Назва_продукту, Номер_партії FROM `productstoshipment` WHERE ShipmentID = @shipmentID GROUP BY Назва_продукту, Номер_партії", db.getConnection());
            command.Parameters.AddWithValue("@shipmentID", id);
            adapter.SelectCommand = command;
            adapter.Fill(table);
            db.closeConnection();
            return table;
        }
        else { return null; }
```

```
    }

    private int GetAvailableQuantity(string productId)
    {
        string query = "SELECT Кількість FROM product WHERE id = @productId";
        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        command.Parameters.AddWithValue("@productId", productId);

        int availableQuantity = 0;

        try
        {
            db.openConnection();
            var result = command.ExecuteScalar();
            if (result != null)
            {
                availableQuantity = Convert.ToInt32(result);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }

        return availableQuantity;
    }


    private void AddProductToShipment(string shipmentId, string productId, string Quantity)
    {

        int availableQuantity = GetAvailableQuantity(productId);

        if (availableQuantity < Convert.ToInt32(Quantity))
        {
            MessageBox.Show("Недостатня кількість продукту для відвантаження");
            return;
        }
        string query = "INSERT INTO productstoshipment (ShipmentID, ProductId, Кількість,
Назва_продукту, Номер_партії) " +
            "SELECT @shipmentId, id, @кількість, Назва, Номер_партії " +
            "FROM product " +
            "WHERE id = @productId";

        MySqlCommand command = new MySqlCommand(query, db.getConnection());

        try
        {
```

```
            db.openConnection();

            command.Parameters.AddWithValue("@shipmentId", shipmentId);
            command.Parameters.AddWithValue("@productId", productId);
            command.Parameters.AddWithValue("@кількість", Quantity);

            int rowsAffected = command.ExecuteNonQuery();


            if (rowsAffected > 0)
            {
                string queryUpdateProductQuantity = "UPDATE product SET Кількість = Кількість -
@quantity WHERE id = @productId";
                MySqlCommand commandUpdateProductQuantity = new
MySqlCommand(queryUpdateProductQuantity, db.getConnection());

                commandUpdateProductQuantity.Parameters.AddWithValue("@productId", productId);
                commandUpdateProductQuantity.Parameters.AddWithValue("@quantity", Quantity);

                int rowsAffectedUpdateProductQuantity =
commandUpdateProductQuantity.ExecuteNonQuery();

                if (rowsAffectedUpdateProductQuantity > 0)
                {
                    MessageBox.Show("Продукт успішно додано до заявки на відвантаження і
кількість продукту оновлено");
                }
                else
                {
                    MessageBox.Show("Помилка оновлення кількості продукту");
                }
            }
            else
            {
                MessageBox.Show("Помилка додавання продукту до заявки на відвантаження");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }

    public void RefreshData()
    {
        comboBox1.DisplayMember = null;
        comboBox1.ValueMember = null;
        comboBox1.DataSource = null;
```

```
    dataGridView1.DataSource = null;
    FillShipmentCombo();
    FillProductCombo();
    dataGridView1.DataSource = ShowTable();
}
private void ProductsToShipmentControl_Load(object sender, EventArgs e)
{
    FillShipmentCombo();
    FillProductCombo();
    dataGridView1.DataSource = ShowTable();


}

private void Nameitem_KeyPress(object sender, KeyPressEventArgs e)
{
    if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
    {
        e.Handled = true;
    }
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    dataGridView1.DataSource = ShowTable();
}

private void AddButton_Click(object sender, EventArgs e)
{
    if(comboBox1.SelectedValue != null)
    {
        if (comboBox2.SelectedValue != null)
        {
            if (Nameitem.Text != null)
            {
                string shipmentid = comboBox1.SelectedValue.ToString();
                string productid = comboBox2.SelectedValue.ToString();
                string Quantity = Nameitem.Text;
                AddProductToShipment(shipmentid, productid, Quantity);
                dataGridView1.DataSource = ShowTable();
            }
        }
    }
    else
    {
        MessageBox.Show("Заповніть всі необхідні значення");
    }
}


private int GetProductId(string shipmentId, string productName, string batchNumber)
{
```

```
        string query = "SELECT ProductId FROM productstoshipment WHERE ShipmentID = @shipmentId AND Назва_продукту = @productName AND Номер_партії = @batchNumber";
        MySqlCommand command = new MySqlCommand(query, db.getConnection());
        command.Parameters.AddWithValue("@shipmentId", shipmentId);
        command.Parameters.AddWithValue("@productName", productName);
        command.Parameters.AddWithValue("@batchNumber", batchNumber);

        int productId = -1;
        try
        {
            db.openConnection();
            var result = command.ExecuteScalar();
            if (result != null)
            {
                productId = Convert.ToInt32(result);
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
        return productId;
    }



    private void dataGridView1_UserDeletingRow(object sender, DataGridViewRowCancelEventArgs e)
    {
        DialogResult result = MessageBox.Show("Ви дійсно бажаєте видалити цей запис?", "Підтвердження видалення", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
        if (result == DialogResult.Yes)
        {
            int rowIndex = e.Row.Index;
            string shipmentId = comboBox1.SelectedValue.ToString();
            string productName = dataGridView1.Rows[rowIndex].Cells["Назва_продукту"].Value.ToString();
            string batchNumber = dataGridView1.Rows[rowIndex].Cells["Номер_партії"].Value.ToString();

            int productId = GetProductId(shipmentId, productName, batchNumber);
            if (productId != -1)
            {
                DeleteProductFromShipment(productId);
            }
            else
            {
```

```
            MessageBox.Show("Не вдалося знайти productId для видалення");
        }
    }
    else
    {
        e.Cancel = true;
    }
}



private void DeleteProductFromShipment(int productId)
{
    int quantityDeleted = 0;
    string querySelectQuantityDeleted = "SELECT Кількість FROM productstoshipment
WHERE ProductId = @productId";
    MySqlCommand commandSelectQuantityDeleted = new
MySqlCommand(querySelectQuantityDeleted, db.getConnection());
    commandSelectQuantityDeleted.Parameters.AddWithValue("@productId", productId);

    try
    {
        db.openConnection();
        object result = commandSelectQuantityDeleted.ExecuteScalar();
        if (result != null)
        {
            quantityDeleted = Convert.ToInt32(result);
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Помилка при отриманні кількості видалених продуктів: " +
ex.Message);
        return;
    }
    finally
    {
        db.closeConnection();
    }
    string queryDeleteFromProductToShipment = "DELETE FROM productstoshipment WHERE
ProductId = @productId";
    MySqlCommand commandDeleteFromProductToShipment = new
MySqlCommand(queryDeleteFromProductToShipment, db.getConnection());
    commandDeleteFromProductToShipment.Parameters.AddWithValue("@productId",
productId);
    string queryUpdateProductQuantity = "UPDATE product SET Кількість = Кількість +
@quantityDeleted WHERE id = @productId";
    MySqlCommand commandUpdateProductQuantity = new
MySqlCommand(queryUpdateProductQuantity, db.getConnection());
    commandUpdateProductQuantity.Parameters.AddWithValue("@productId", productId);
    commandUpdateProductQuantity.Parameters.AddWithValue("@quantityDeleted",
quantityDeleted);
```

```
        try
        {
            db.openConnection();
            int rowsAffectedDeleteFromProductToShipment = commandDeleteFromProductToShipment.ExecuteNonQuery();
            int rowsAffectedUpdateProductQuantity = commandUpdateProductQuantity.ExecuteNonQuery();
            if (rowsAffectedDeleteFromProductToShipment > 0 && rowsAffectedUpdateProductQuantity > 0) {
                MessageBox.Show("Запис успішно видалено з таблиці productstoshipment та кількість продукту оновлено");
            }
            else
            {
                MessageBox.Show("Помилка видалення запису з таблиці productstoshipment або оновлення кількості продукту");
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show("Помилка: " + ex.Message);
        }
        finally
        {
            db.closeConnection();
        }
    }
  }
}
```

Код вкладки зміни паролю користувача:

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

namespace DIPLOMFINAL
{
  public partial class PasswordChange : UserControl
  {
    private string _LastLogin;
    public PasswordChange()
    {
```

```
    InitializeComponent();
}


public string oldlogin
{
    get { return _LastLogin; }
    set
    {
        _LastLogin = value;
                }
}


private void checkBox1_CheckedChanged(object sender, EventArgs e)
{
    if (checkBox1.Checked)
    {
        PassBox.UseSystemPasswordChar = false;

    }
    else
    {
        PassBox.UseSystemPasswordChar = true;

    }
}

private void button1_Click(object sender, EventArgs e)
{
    string newPass = PassBox.Text;
    string login = _LastLogin;
    DB db = new DB();
    string query = "UPDATE users SET Пароль = @Пароль WHERE Логін = @Логін";

    MySqlCommand command = new MySqlCommand(query, db.getConnection());
    command.Parameters.AddWithValue("@Пароль", newPass);
    command.Parameters.AddWithValue("@Логін", login);

    try
    {
        db.openConnection();
        int rowsAffected = command.ExecuteNonQuery();
        if (rowsAffected > 0)
        {
            MessageBox.Show("Пароль користувача " + login + " був успішно змінений.");
        }
        else
        {
            MessageBox.Show("Не вдалося змінити пароль користувача " + login + ".");
        }
    }
```

```
            catch (Exception ex)
            {
                MessageBox.Show("Помилка при оновленні паролю користувача: " + ex.Message);
            }
            finally
            {
                db.closeConnection();
            }
        }

        private void PasswordChange_Load(object sender, EventArgs e)
        {
            if (_LastLogin != null)
            {
                OldPass.Text = _LastLogin;
            }
        }
    }
}
```

Код створеного класу для підключення до БД

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Xml.Linq;
using MySql.Data.MySqlClient;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace DIPLOMFINAL
{
    class DB
    {
        public MySqlConnection connection = new MySqlConnection("server=localhost;port=3306;username=Admin;password=Admin;database=Yagotyn");

        public void openConnection()
        {

            if (connection.State == System.Data.ConnectionState.Closed)
            {
                try
                {
                    connection.Open();

                }
                catch (Exception ex)
```

```
            {
                MessageBox.Show("Помилка з підключенням до серверу " + ex.Message);
            }

        }
    }
    public void closeConnection()
    {

        if (connection.State == System.Data.ConnectionState.Open)
        {
            connection.Close();
        }
    }
    public MySqlConnection getConnection()
    {

        return connection;
    }
  }
}
```