Хмельницький національний університет
Факультет інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА

Вітвіцький Юрій Євгенович
Прізвище, ім'я, по батькові студента(ки)

на здобуття ступеня вищої освіти Бакалавра

Програмне забезпечення для управління складом
Назва теми

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

Шифр КвРІПЗ. 200176.01.25.ПЗ

Виконав студент IV курсу, група ІПЗ-20-1 _____ Юрій ВІТВІЦЬКИЙ
Підпис Ім'я. ПРІЗВИЩЕ

Керівник асистент _____ Анастасія ДЬОМІНА
Науковий ступінь. вчене звання Підпис Ім'я. ПРІЗВИЩЕ

Нормоконтролер канд. техн. Наук, доцент _____ Оксана ЯШИНА
Посада Підпис Ім'я. ПРІЗВИЩЕ

До захисту допускаю:
Завідувач кафедри інженерії
програмного забезпечення _____ Леонід БЕДРАТЮК
Підпис Ім'я. ПРІЗВИЩЕ

13 червня 2024 р.

Хмельницький 2024

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Інженерії програмного забезпечення

Рівень вищої освіти Перший (бакалаврський)

Галузь знань 12 «Інформаційні технології»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня програма Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Я. П. Бедратюк
2.01 2024 2024 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

Вітвіцькому Юрію Євгеновичу

Прізвище. ім`я. по батькові студента

1. Тема кваліфікаційної роботи Програмне забезпечення для управління складом

Керівник проекту (роботи) Дьоміна Анастасія Іванівна, асистент

Прізвище. ім`я. по батькові. науковий ступінь. вчене звання

Затверджена наказом ректора університету від 08.01.2024 р. № 6

2. Строк подання студентом роботи на кафедру _____ р.

3. Вихідні дані до роботи Матеріали переддипломної практики

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити): Дослідження предметної області та постановка задачі, проектування програмного забезпечення, програмна реалізація, тестування програмного забезпечення

5. Перелік графічного матеріалу (із зазначенням обов`язкових креслень:Презентаційні матеріали (слайди. шт.) 15

6. Консультанти розділів дипломного проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	к.т.н.доцент Яшина О.М	9.06.24	12.06.24
Антиплагіат	к.т.н.доцент Фартушко В	30.05.24	06.06.24

7. Дата видачі завдання « » 2024р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1 Ознайомлення з тематикою дипломного проектування (ДП). визначення та узгодження індивідуальних тем кваліфікаційних робіт (КвР)	01.12 – 31.12.2023	
2 Збір матеріалу за темою КвР: дослідження предметної області, в якій планується використання програмного забезпечення (ПЗ). визначення задач та вимог. розробка технічного завдання	01.01 – 20.02.2024	
3 Проектування програмного забезпечення	21.02 – 20.03.2024	
4 Програмна реалізація з використанням відповідних засобів розробки. Тестування ПЗ	21.03 – 30.04.2024	
5 Написання вступу. загальних висновків. оформлення джерел посилання та додатків. Оформлення пояснювальної записки КвР згідно вимог стандартів	01.05 – 25.05.2024	
6 Попередній захист КвР	Травень 2024	Згідно графіка
7 Перевірка КвР на плагіат. нормоконтроль. отримання відгуків. рецензій та інших супровідних документів. Брошурування (зшиття) пояснювальної записки.	26.05 – 30.05.2024	
8 Здача КвР на кафедру: підготовка КвР для розміщення у репозитарії ХНУ; підготовка до захисту та захист КвР	з 01.06.2024	

Студент _____ Ю. Є. Вітвіцький
Підпис Ініціали. прізвище

Керівник роботи _____ А. І. Дьоміна
Підпис Ініціали. прізвище

ВІДОМІСТЬ ДОКУМЕНТІВ

№ рядка	Формат	Позначення документа	Найменування документа	К-сть аркушів	№ екз.	Примітка
			Текстові документи			
1	А4	КвРІПЗ.200176.01.25.ПЗ	Пояснювальна записка	82		
2	А4		Завдання на кваліфіка-	1		
			ційну роботу			
3	А4		Анотація	1		
			Графічні документи			
4	А3	КвРІПЗ.200176.01.25.Е8	Еталонна архітектура	1		
5	А3	КвРІПЗ.200176.01.25.Е8	Логічна схема даних	1		
			бази для складу			
6	А3	КвРІПЗ.200176.01.25.Е8	Алгоритм роботи	1		
			програми			
7	А3	КвРІПЗ.200176.01.25.Е8	Діаграма прецедентів	1		

Змн.	Арк.	№ докум.	Підпис	Дата	КвРІПЗ. 200176.01.25.ВД		
Виконав		Вітвіцький Ю.Є			Програмне забезпечення для управління складом	Літ.	Арк. 1 / Аркушів 1
Керівник		Дьоміна А.І.					
Н. контр.		Яшина О.М.			Відомість документів	ХНУ, ІПЗ-20-1	
Зав. каф.		Бедратюк Л.П.					

ЗМІСТ

ВСТУП

WMS, або система управління складом, — це програмне забезпечення, яке допомагає компаніям керувати щоденними складськими операціями та контролювати їх, починаючи з моменту надходження товарів і матеріалів у центр розподілу або виконання до моменту їх виходу. Програмні системи WMS є ключовим компонентом управління ланцюгом постачань і забезпечують видимість у реальному часі всіх запасів компанії, на складах і в дорозі. Окрім керування запасами, WMS пропонує інструменти для процесів комплектування та пакування, використання ресурсів, аналітики тощо.

Надійна цифрова система управління складом необхідна для будь-якого бізнесу з наявними товарними запасами – вона може допомогти заощадити гроші та підвищити ефективність у багатьох сферах. П’ять основних переваг системи WMS:

Підвищення операційної ефективності: системи WMS автоматизують і оптимізують складські процеси від вхідних надходжень до вихідних поставок – для підвищення ефективності, більш плавної роботи та здатності обробляти великі обсяги. Вони зменшують помилки під час комплектування та відвантаження товарів і усувають повторну та непотрібну роботу. WMS також обмінюється даними з ERP і системами управління транспортуванням, надаючи вам цілісне уявлення, яке виходить за межі вашого складу та допомагає прискорити рух товарів.

Зменшення відходів і витрат: якщо у вас є запаси з обмеженим терміном використання або швидкопсувні запаси, програмне забезпечення WMS може визначити, які позиції потрібно відібрати першими або які можуть потребувати стимулювання продажу, щоб мінімізувати відходи. Це також може допомогти вам визначити найефективніше використання складського простору, від розміщення запасів до оптимальних маршрутів руху. Деякі системи пропонують вдосконалене моделювання для створення планів

поверхів і розміщення піддонів, полиць і обладнання в найкращих місцях, щоб працювати з максимальною ефективністю та економити час і гроші.

Видимість запасів у режимі реального часу: використовуючи штрих-кодування, RFID-мітки, датчики чи інші методи відстеження місцезнаходження, система WMS дає вам уявлення про запаси в реальному часі, коли вони переміщуються на ваш склад, навколо нього та до наступного місця. Завдяки цій видимості ви можете створювати точніші прогнози попиту, виконувати стратегію своєчасної інвентаризації та покращувати відстеження, що особливо важливо у випадку відкликання

Покращене управління робочою силою: WMS може допомогти вам прогнозувати потреби в робочій силі, створювати графіки, оптимізувати час у дорозі в межах складу та призначати правильне завдання потрібному працівнику на основі рівня кваліфікації, близькості та інших факторів. Хороша система WMS також може допомогти підвищити моральний дух працівників, створивши більш спокійне, організоване та безпечне середовище, де працівники відчувають, що їхній час цінується та використовується з розумом.

Кращі відносини з клієнтами та постачальниками: за допомогою WMS клієнти насолоджуються покращеним виконанням замовлень, швидшою доставкою та меншою кількістю неточностей, що підвищує їхнє задоволення та лояльність і покращує репутацію вашого бренду. Постачальники також можуть відчути скорочення часу очікування на вантажних майданчиках і доках для покращення відносин.

Об’єкт дослідження-діяльність підприємств що використовують програми для автоматизації складської діяльності.

Предмет-розробка програмного забезпечення для автоматизації складської діяльності.

Завдання :

- дослідити матеріал пов’язаний з складською діяльністю;
- дослідити процес розробки програмного забезпечення;
- розробити програмний застосунок;

РОЗДІЛ 1.ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Змістовний аналіз предметної області, її структурних та функціональних особливостей

Системи управління запасами відіграють важливу роль у забезпеченні ефективного та ефективного управління запасами компанії. Ці системи надають підприємствам інструменти та функції для відстеження, контролю та оптимізації рівня запасів. Хоча вони відрізняються за функціональністю, більшість з них дозволяють користувачам обліковувати всі вхідні та вихідні запаси, що впливає на все, від обліку та виробництва до продажів і задоволеності клієнтів.

Система управління запасами — це програмне рішення, яке дозволяє підприємствам контролювати та контролювати свої запаси від закупівлі до виконання замовлення. Він діє як централізований центр, забезпечуючи видимість у реальному часі та точні дані щодо запасів щодо рівня запасів, руху та наявності. Завдяки вдосконаленому відстеженню запасів компанії можуть оптимізувати продуктивність бізнесу, зменшити витрати та краще відповідати очікуванням клієнтів.

Важливі характеристики системи управління запасами

Певні специфікації завжди будуть вирішальними для оптимальної продуктивності під час створення системи інвентаризації. Загальні вимоги до системи управління запасами включають:

Масштабованість – це дозволяє системі обробляти зростаючі обсяги запасів у міру зростання бізнесу.

Автоматизація – це допомагає зменшити ручні процеси, пов’язані з керуванням складом, запасами та іншими бізнес-функціями.

Безпека. Щоразу, коли ви використовуєте систему, яка зберігатиме записи про запаси, замовлення на продаж і замовлення на купівлю ви хочете переконатися, що вона має найновіші функції безпеки.

Мобільна доступність – найкращі системи управління запасами дозволяють користувачам планувати запаси на ходу. Це забезпечує кращу швидкість реагування, особливо якщо компанія має кілька філій.

Можливості інтеграції – якщо нова IMS може легко підключатися до інших бізнес-систем (бухгалтерського обліку, електронної комерції тощо), це допоможе оптимізувати бізнес-процеси.

Простий інтерфейс – серед провідних вимог до управління запасами є простий у використанні інтерфейс. Це допомагає скоротити час навчання та потребу в підтримці.

Основні функції системи управління запасами за категоріями

Нижче ми включили деякі ключові функції для розробки системи управління запасами. Хоча складські функції відрізняються від компанії до компанії, майже кожному підприємству знадобляться певні функції в наведених нижче категоріях

Контроль запасів – ця функція обробляє продукти, які вже є на складі компанії. Зазвичай це включає в себе інструменти для класифікації продуктів, визначення їх місцезнаходження за серійним номером або SKU, а також створення звітів у реальному часі та сповіщень про низькі запаси.

Управління запасами. Система управління запасами повинна підтримувати різні базові складські функції, такі як обробка замовлень, включаючи створення, відстеження та виконання замовлень

Штрих-кодування – додавання програмного забезпечення для штрих-кодування може допомогти зменшити проблеми із введенням даних шляхом максимальної автоматизації процесу. Це також полегшує збір і впорядкування як елементів, так і даних.

Прогнозування та планування попиту. Однією з розширених функцій управління запасами є прогнозування та планування попиту. По суті, ці програми використовують алгоритми для прогнозування моделей попиту та відповідно керують рівнями запасів.

Оптимізація запасів – оптимізація запасів вважається наступним рівнем управління запасами. Це передбачає використання інструментів, які можуть автоматизувати етапи процесу, створювати звіти на основі минулих і поточних даних і загалом покращувати зусилля з управління ланцюгом постачання в усіх напрямках.

Особливості управління запасами, які слід враховувати

Тепер, коли ми визначили деякі з основних категорій функцій керування запасами, давайте розглянемо деякі менш відомі або нові категорії, які можуть вплинути на можливості IMS.

Управління постачальниками. Ефективне керування постачальниками дозволяє підприємствам підтримувати міцні відносини з постачальниками, керувати замовленнями на закупівлю, відстежувати ефективність постачальників та запаси.

Звітування та аналітика – інтуїтивно зрозумілі можливості звітування та аналітики надають підприємствам цінну інформацію про продуктивність своїх запасів. Він також може допомогти в управлінні замовленнями на купівлю, передачею та звітами про оборот.

Хмарна інфраструктура. Хмарне програмне забезпечення є масштабованим і економічно ефективним рішенням. Хоча це не обов’язково, воно стало однією з найпопулярніших вимог до системи відстеження запасів, особливо коли задіяно кілька місць.

Інтеграція Інтернету речей – Інтернет речей пропонує незліченну кількість рішень для компаній будь-якого розміру та типу, від відстеження RFID та цифрового маркування до відстеження GPS тощо.

Мобільні системи – Мобільна система може бути найбільш логічним вибором, залежно від конкретних інструкцій із впровадження системи управління запасами. Багато з цих рішень IMS працюють на планшетах, iOS або Android.

Машинне навчання та штучний інтелект. Хоча це аж ніяк не одна з основних функцій систем управління запасами, машинне навчання може аналізувати дані, щоб покращити керування постачальниками, зменшити витрати на доставку та створити рекомендований запас.

Основні вимоги до системи управління запасами

Команда InventorSoft проаналізувала широкий спектр критеріїв оцінки системи управління запасами. Розглядаючи різні функції, категорії та специфікації вище, ми склали список основних вимог IMS. Вважайте це списком найкращих практик щодо вибору програмного забезпечення для керування запасами.

Управління запасами

Важливою вимогою до системи управління запасами є ефективне керування всіма аспектами, пов’язаними з контролем запасів. Це включає точне відстеження рівня запасів, моніторинг переміщень товарів і впорядкування даних про запаси. Система повинна дозволити підприємствам легко переглядати, керувати та оновлювати інформацію про свої запаси, щоб усунути розбіжності даних, оптимізувати операції та приймати обґрунтовані рішення щодо поповнення запасів. Нарешті, користувачі можуть розглянути концепцію, відому як Vendor Managed Inventory, яка є унікальним рішенням, придатним для довгострокових ділових відносин. Особливості включають:

- категоризація продукту;
- історія продукту;
- спільна інвентаризація;
- автоматизовані звіти про відсутність;
- інвентаризація, керована постачальником;

– електронна комерція;

– фондові запити;

– ефективне управління переказами.

Надійна IMS має полегшувати переміщення запасів між різними місцями або складами, відстеження замовлень на передачу та відповідне оновлення рівня запасів. Така функція є важливою, оскільки вона дозволяє компаніям, які обслуговують кілька сайтів, переміщувати свій інвентар туди, де це принесе найбільшу цінність. Важливим є те, що система забезпечує чітку видимість статусів переказів, полегшує розширений контроль запасів і мінімізує брак або надмірні запаси.

Особливості включають:

– вибір замовлення;

– підбір голосу;

– відстеження кількох місць;

– передача запасів;

– Pick-To-Light;

– звітність і аналітика.

Комплексні можливості звітності та аналітики є одними з найкращих вимог до системи оптимізації запасів у будь-якій IMS. Метою тут є створення звітів щодо таких речей, як рівень запасів, тенденції продажів, історія замовлень і оборотність запасів. Це допомагає компаніям оцінювати стан своїх запасів, надаючи їм цінну інформацію про продуктивність запасів. Зрештою, ці дані допомагають встановити інвентаризаційні показники та KPI, які можуть допомогти у прийнятті керованих даними рішень, які зменшують резервний запас і скорочують транспортні витрати. Зрештою, можливості програмного забезпечення для прогнозування запасів є величезним бонусом для будь-якого бізнесу, незалежно від галузі.

Особливості включають:

– налаштування звіту;

– інтеграція звітів;
– прості у використанні інформаційні панелі;
– KPI та метрична довідка;
– доставка та логістика.

Ще однією з найбільш затребуваних функцій системи управління запасами є можливість бездоганної інтеграції цього рішення з транспортуванням і логістичними операціями. Це може значно полегшити такі завдання, як маркування та доставка декількома перевізниками, а також гарантувати, що клієнти отримають свої товари якомога швидше та ефективніше. Залежно від вибраного програмного забезпечення IMS може дозволити користувачам друкувати транспортні етикетки, відстежувати відправлення та керувати процесами виконання замовлень з одного місця. Нарешті, координація з API транспортних перевізників може надати інформацію про відстеження в реальному часі та автоматизувати оновлення доставки. Особливості включають:

– відправка по ID;
– інтеграція API;
– маркування;
– доставка кількома перевізниками;
– кілька замовлень на доставку;
– розгортання.

IMS із гнучкими параметрами розгортання ідеально підходить, оскільки як локальні, так і хмарні варіанти мають свої переваги та недоліки. Зрештою, найкраще дозволити компаніям вибрати варіант, який найкраще відповідає їхнім вимогам до інфраструктури та масштабованості. Звичайно, хмарні продукти інвентаризації пропонують легкий доступ, масштабованість і автоматичне оновлення програмного забезпечення. Однак локальне розгортання забезпечує кращий контроль і безпеку для підприємств із особливими потребами відповідності. Який би вибір вони не зробили, система

має забезпечувати плавний та ефективний процес розгортання, щоб мінімізувати збої та дозволити підприємствам якомога швидше використовувати управління запасами.

Оптимізація операцій Dropshipping

Все більше і більше компаній використовують бізнес-модель дропшиппінгу для отримання пасивного доходу. Якщо це схоже на вас, важливо, щоб ваша система управління запасами ефективно полегшувала ваші операції дропшиппінгу. В ідеалі ваша IMS могла б легко інтегруватися з постачальниками дропшиппінгу, автоматично пересилати замовлення постачальникам для виконання та оновлювати інформацію про відстеження для клієнтів. Як і в будь-якій іншій операції, видимість у реальному часі тут є величезним бонусом, оскільки вона може допомогти компаніям краще керувати відносинами з постачальниками, розширюючи пропозицію продуктів і знижуючи витрати на зберігання запасів. Особливості включають:

- автоматизоване замовлення та виконання;
- управління джерелами;
- управління постачальником;
- видимість у реальному часі ;
- оптимізація прогнозування попиту;
- зв'яжіться з нами.

Комплексні можливості прогнозування попиту можуть допомогти компаніям точно передбачити, коли їхні клієнти будуть готові купувати. Ці програми можуть аналізувати історичні дані, ринкові тенденції та сезонність, щоб створювати прогнози попиту на різні продукти та артикули. Оптимізувавши прогнозування попиту, підприємства можуть покращити планування запасів, зменшити брак і надмірні запаси, а також створити точки повторного замовлення, які запускаються автоматично протягом року. Особливості включають:

- захист від запасів;

– аналіз тенденцій ринку;
– контроль продажів;
– автоматичне повторне замовлення;
– ефективна практика закупівель.

Гарне рішення для закупівель дозволить підприємствам автоматизувати створення замовлень на купівлю, керувати відносинами з постачальниками та відстежувати статуси замовлень. Багато найкращих прикладів починаються з шаблонів, а потім заповнюються наявними даними, отриманими з інших частин IMS. Це може забезпечити видимість часу виконання постачальником і ціноутворення, таким чином полегшуючи прийняття обґрунтованих рішень про закупівлю.

Особливості включають:

– замовлення на придбання;
– автоматизоване формування замовлення на купівлю;
– отримання;
– управління постачальником;
– відстеження статусу замовлення;
– мобільна підтримка.

Мобільна підтримка дозволяє компаніям отримувати доступ до даних про запаси та керувати ними на ходу. Для цього багато рішень пропонують мобільні веб-сайти або інформаційні панелі, доступ до яких можна отримати з телефону, планшета чи інших мобільних пристроїв. Звідси користувачі можуть виконувати завдання, пов’язані з інвентаризацією, наприклад переглядати рівень запасів, створювати замовлення та відстежувати відправлення, будь-де, де вони можуть знайти підключення до Інтернету. Особливості включають:

– мобільний сайт;
– ОС для різних пристроїв ;
– Just in Time (JIT) Управління запасами;

– Just in Time Inventory Management.

Управління запасами Just in Time (JIT) стосується ситуації, коли на складі є мінімальна кількість запасів, необхідна для задоволення попиту. Це знижує витрати на запаси, одночасно усуваючи ризик втрати продукції та мертвих запасів. Щоб це зробити, система повинна забезпечувати точне прогнозування попиту, підтримувати автоматичне повторне замовлення або поповнення запасів і сприяти ефективному управлінню постачальниками.

Особливості включають:

– сповіщення про запаси;

– прогнозування;

– автоматичне перезамовлення та поповнення;

– відстеження пакетів.

Найкращі рішення IMS дозволяють користувачам відстежувати та керувати запасами за партіями. Це особливо важливо для підприємств, які працюють зі швидкопсувними товарами, фармацевтичними препаратами або продуктами з певним терміном виготовлення чи терміном придатності. Система повинна дозволити підприємствам призначати номери партій продуктам, відстежувати рух партій і забезпечувати належну ротацію для мінімізації відходів і дотримання нормативних вимог. Це також може бути корисним у разі відкликання виробника.

Вирішальні вимоги до інтеграції

Хороший IMS повинен добре працювати сам по собі. Однак у міру того, як ланцюг постачання стає все більш цифровим, стає важливішим, щоб усі різноманітні програми, API та типи програмного забезпечення могли спілкуватися один з одним. Розглядаючи інвестиції в систему управління запасами, переконайтеся, що вона може інтегруватися з наступним:

Програми електронної комерції

Інтеграція з програмами електронної комерції є великою перевагою для IMS. Це дозволяє користувачам автоматично синхронізувати дані про запаси

між IMS і однією або декількома платформами електронної комерції, забезпечуючи при цьому точність і актуальність інформації в онлайн-магазині. Зрештою, це спрощує робочий процес між двома рішеннями, зменшуючи введення даних вручну та мінімізуючи ризик помилок або перепродажу.

Бухгалтерські програми

Ще одна важлива вимога до інтеграції IMS стосується програм бухгалтерського обліку. Мета тут полягає в тому, щоб полегшити безперебійний потік фінансових даних між системою управління запасами та бухгалтерським програмним забезпеченням. Це має вирішальне значення для синхронізації продажів, закупівель і транзакцій, пов'язаних із запасами, одночасно забезпечуючи точні та послідовні фінансові записи.

Інтеграції API

Інтеграції API (інтерфейс програмування прикладних програм) дозволяють системі управління запасами спілкуватися та обмінюватися даними з іншими програмними додатками. Мати IMS, яка не може «спілкуватися» з транспортними компаніями, платформами дропшиппінгу, системами POS-терміналів тощо, мало цінності. API — це те, що допомагає полегшити мікросервіси в програмному забезпеченні ланцюга постачання З часом така інтеграція ставатиме все більш важливою для прибутковості.

1.2 Аналіз наявного програмно-технічного забезпечення предметної області

Система управління складом(рис 1.1) — це потужний програмний інструмент, спеціально розроблений для оптимізації складських операцій, оптимізації управління запасами та підвищення загальної ефективності.

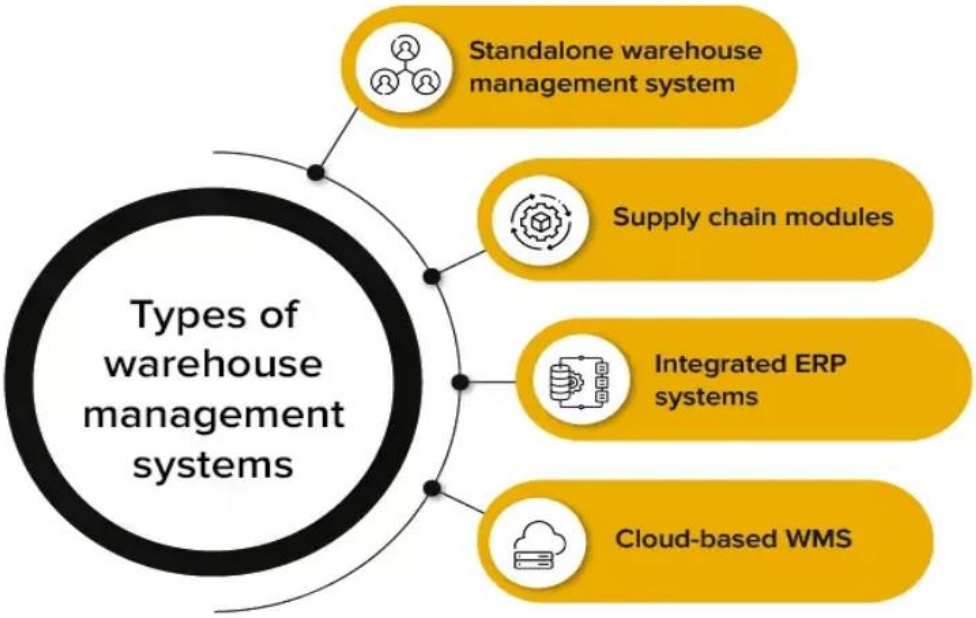


Рисунок 1.1- Типи систем для діяльності складу

Автономна система управління складом — це потужне програмне рішення для оптимізації та оптимізації кожного аспекту ваших складських операцій що забезпечує повний контроль та ефективність, як ніколи раніше.

Це вдосконалене програмне забезпечення оцифровує та автоматизує ключові завдання щоб усунути людські помилки та підвищити точність. Він аналізує дані, визначає шаблони та розумно організовує макет вашого складу.

У порівнянні з іншими типами WMS, впровадження автономної WMS часто вимагає меншої складності та меншої кількості залежностей від зовнішніх систем. Вони працюють незалежно, не потребуючи широкої інтеграції з іншим програмним забезпеченням або складного обміну даними.

Оскільки він функціонує як самодостатня система, процес впровадження оптимізований і зосереджений головним чином на налаштуванні WMS для узгодження з вашими складськими операціями.

Тим не менш, впровадження автономної WMS вимагає ретельного планування та співпраці. Це командна робота, у якій беруть участь люди з ваших складських операцій, IT-відділу та постачальника WMS.

Проаналізуйте свої поточні складські процеси та інфраструктуру а також перевірте, як управляються ваші запаси, як виконуються замовлення та будь-які існуючі системи. Після цього програмне забезпечення буде налаштовано відповідно до плану вашого складу структури запасів і робочих процесів. Під час трансляції почніть фазу ретельного тестування, щоб гарантувати, що все працює бездоганно [35].

Спрощене виконання замовлень: автономний WMS(рис 1.2) керує операціями комплектування, пакування та доставки та направляє вашу команду найоптимальнішими шляхами.

Удосконалена інтеграція штрих-кодів і RFID: бездоганна інтеграція зі сканерами штрих-кодів і зчитувачами RFID забезпечує швидке та безпомилкове сканування та відстеження запасів.

Майстерність управління запасами: з автономною WMS ви отримуєте неперевершений контроль над своїми запасами. Він відстежує кожен предмет, записуючи такі деталі, як місце розташування, кількість і історія переміщення.

Відстеження партій і партій: для галузей, які мають справу зі швидкопсувними або регульованими товарами, автономна WMS пропонує розширені можливості для відстеження та відстеження продуктів на основі номерів партій або партій [21].

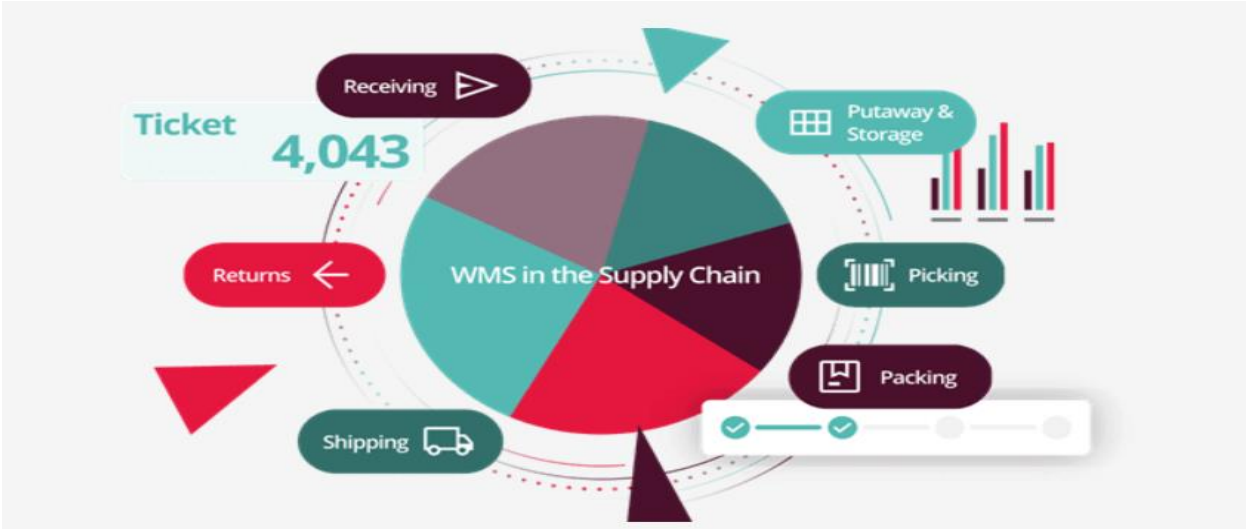


Рисунок 1.2 – Елементи WMS

Управління поверненнями та зворотною логістикою: автономна WMS надає спеціальні функції для керування поверненнями, відстеження потоків продуктів назад на склад і ефективної обробки зворотної логістики.

Управління працею та відстеження продуктивності: програмне забезпечення фіксує дані про виконані завдання, витрачений час і показники продуктивності, щоб ви могли визначити області для вдосконалення та оптимізувати розподіл ресурсів [12].

Інтелектуальна оптимізація складу: вона аналізує дані та стратегічно організовує ваші продукти на основі попиту та частоти доступу. Створення логічних зон, призначення відповідних місць зберігання та оптимізація шляхів комплектування допомагають мінімізувати непотрібний час у дорозі для вашої робочої сили.

Системи управління складом модуля ланцюга постачання призначені для інтеграції з існуючою системою або модулем управління ланцюгом постачання Ця інтеграція забезпечує плавний потік даних і видимість у реальному часі по всьому ланцюжку постачання, від закупівлі до розподілу.

Основна функція модуля ланцюга постачання WMS полягає в ефективному управлінні та контролі потоку товарів на складі.

Одним із ключових аспектів цього типу системи є її здатність керувати складними процесами ланцюжка поставок. Це гарантує оптимальний розподіл запасів, швидку обробку замовлень і оперативну доставку продуктів клієнтам.

Крім того, системи керування складськими складами в модулях ланцюга постачання дозволяють підприємствам відстежувати та відстежувати продукти протягом усього шляху в ланцюжку поставок.

Порівняно з іншими типами [28], впровадження системи управління складським складом із модулем ланцюга постачання представляє певний набір проблем. Рівень складності може змінюватися залежно від кількох факторів, зокрема:

Міграція даних: рівень складності зростає з великими наборами даних, складними структурами даних і потребою в очищенні чи перетворенні даних.

Складність існуючого програмного забезпечення для керування ланцюгом поставок: для розрізнених систем або кількох застарілих систем встановлення вимагає більше зусиль і досвіду.

Вимоги до налаштувань: якщо систему потрібно суттєво адаптувати, для відповідного налаштування та тестування системи знадобиться додатковий час і досвід.

Обсяг інтеграції: якщо інтеграція вимагає складного відображення даних, складного узгодження процесу та обміну інформацією в режимі реального часу, процес інсталяції може бути складнішим.

Ось деякі основні характеристики модульних систем управління складом ланцюга постачання.

Ефективне виконання замовлень: ця система забезпечує інтелектуальний розподіл замовлень, оптимізацію комплектування та ефективність пакування, забезпечуючи швидку та точну обробку замовлень.

Автоматизоване керування завданнями: система керування складським модулем ланцюга постачання дозволяє вам автоматизувати такі завдання, як приймання, поповнення, пакування, маркування та відвантаження.

Видимість у режимі реального часу: це дає змогу бачити в реальному часі ваш інвентар, замовлення та загальну продуктивність складу. Слідкуйте за рівнем запасів, відстежуйте стан замовлень і легко виявляйте вузькі місця.

Централізоване управління запасами: це дає вам повне уявлення про ваші запаси в різних місцях і дозволяє відстежувати рівень запасів, стежити за рухом продукції та ефективно оптимізувати простір для зберігання.

Розширена аналітика та звітність [10]: він збирає важливі дані про KPI, як-от точність замовлень, оборотність запасів і продуктивність праці. Перетворюючи ці дані в глибокі звіти й аналітику, ви отримуєте корисну інформацію для оптимізації процесів.

Модуль ланцюга поставок WMS пропонує перевагу спільної роботи над аналогами. Завдяки можливості інтегрувати та обмінюватися даними в реальному часі з постачальниками, продавцями та іншими зацікавленими сторонами ці системи сприяють співпраці та забезпечують більш плавну координацію в усій мережі постачання.

Модуль ланцюга постачань WMS дозволяє знайти ідеальний баланс між ручними та автоматизованими процесами пристосовуючи систему до ваших унікальних операційних вимог. Незалежно від того, чи вирішите ви автоматизувати конкретні завдання, запровадити робототехніку чи застосувати передові технології, як-от IoT або AI, ці системи можуть легко інтегруватися та адаптуватися, підвищуючи операційну ефективність і продуктивність.

Інтеграція з іншими системами

Модуль ланцюга постачань WMS може легко інтегруватися з різними системами у вашій екосистемі ланцюга постачання, зокрема:

платформи електронної комерції [20].

- системи управління виробництвом (MES);
- системи управління транспортом (TMS);
- системи планування ресурсів підприємства (ERP);
- Business Intelligence (BI) і аналітичні платформи;
- системи управління взаємовідносинами з клієнтами (CRM).

Інтегрована в ERP система управління складом(рис 1.3) працює шляхом інтеграції 2 основних компонентів : системи ERP і WMS. Інтеграція починається з синхронізації даних між системами ERP і WMS.

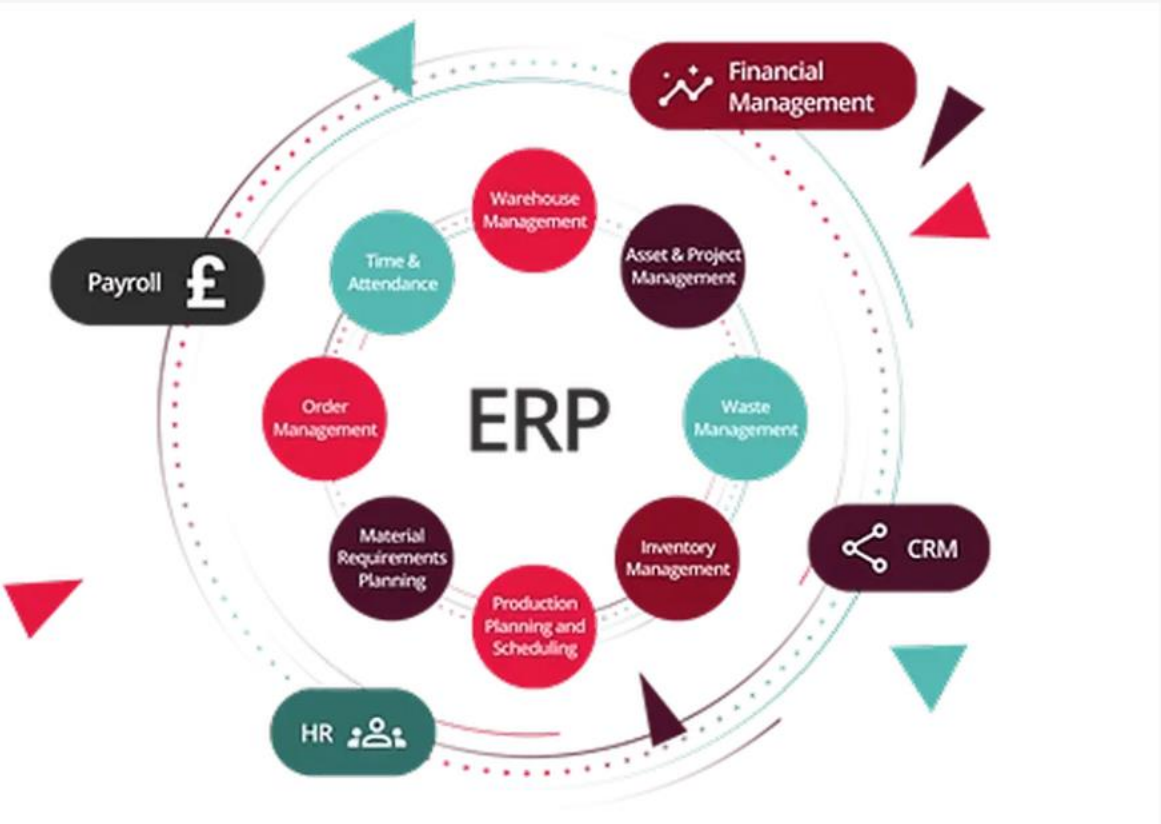


Рисунок 1.3 – Інтегрована в ERP система

Ця інтеграція часто досягається через системні інтерфейси або API Це забезпечує обмін даними та зв’язок між системами ERP і WMS у режимі реального часу. Основні дані, такі як коди позицій, описи, одиниці вимірювання та розташування складів, повинні бути узгоджені між системами ERP і WMS, щоб уникнути розбіжностей і забезпечити точне управління запасами [33].

Оскільки інтегрована в ERP система управління складом інтегрована з більш широкою системою ERP, вона використовує існуючу інфраструктуру та дані [25] Це означає, що вам не потрібно починати з нуля або інвестувати в окреме програмне забезпечення для управління складом.

Як і будь-яке важливе впровадження, правильне планування є життєво важливим. Це включає проведення ретельного аналізу ваших існуючих процесів, визначення цілей управління складом і визначення точок інтеграції з системою ERP.

Щоб забезпечити точність і узгодженість даних під час процесу інтеграції та передачі даних вам потрібно зіставити поля даних, очистити дані та встановити безперебійні з'єднання даних між системою ERP і системою управління складом.

Ось функціональні можливості та функції, завдяки яким система управління складом, інтегрована в ERP, змінює правила гри.

Спрощена обробка замовлень: система автоматизує робочі процеси керування замовленнями, починаючи від введення замовлення до комплектування, пакування та відправлення.

Комплексне управління запасами: ви можете точно відстежувати рівень запасів, відстежувати рух продукції та легко керувати запасами в кількох місцях.

Точне прогнозування попиту: ця система використовує історичні дані та аналізує тенденції покупців, щоб ви могли передбачити майбутній попит, відповідно скоригувати рівень запасів і оптимізувати процеси ланцюжка поставок.

Переваги ERP: інтеграція з ERP-системою відкриває цілий світ можливостей. Ви можете легко обмінюватися даними між відділами. Ваша команда продажів може отримати доступ до наявності запасів у реальному часі, ваш фінансовий відділ може відстежувати складські витрати, а ваша команда закупівель може оптимізувати замовлення на закупівлю на основі фактичного попиту.

Багатоканальне виконання: у сучасному багатоканальному бізнес-ландшафті WMS, інтегрована в ERP, надає вам інструменти для виконання замовлень різними каналами. Система забезпечує ефективну обробку замовлень і розподіл запасів відповідно до потреб кожного каналу, незалежно від того, чи йдеться про поставки безпосередньо споживачам, оптову дистрибуцію чи онлайн-ринки.

WMS, інтегрована в ERP, використовує інфраструктуру існуючої системи ERP. Це усуває потребу в окремих сховищах даних або дублюючих процесах Спираючись на основу вашої ERP-системи, WMS може безперешкодно синхронізувати дані між відділами, підвищуючи ефективність і зменшуючи складність.

Однак інтеграція WMS із системою ERP означає, що масштабованість і гнучкість системи управління складом залежать від можливостей і обмежень самої системи ERP.

Будь-які обмеження або проблеми в системі ERP впливатимуть на функціональність WMS. Оцініть можливості та обмеження основної системи ERP перед впровадженням інтегрованої в ERP WMS.

Інтеграція з іншими системами

Ось лише кілька прикладів систем, з якими може інтегруватися система управління складом, інтегрована в ERP.

- системи управління активами;
- системи управління якістю (СУЯ);
- системи управління виробництвом (MES);
- системи керованої інвентаризації постачальника (VMI);
- системи управління транспортом (TMS);
- фінансові системи (ERP, бухгалтерське програмне забезпечення);
- системи планування потреб у матеріалах (MRP);
- системи управління взаємовідносинами з клієнтами (CRM).

1.3 Визначення вимог до програмного забезпечення та технічне завдання

Системи управління запасами відіграють важливу роль у забезпеченні ефективного та ефективного управління запасами компанії. Ці системи надають підприємствам інструменти та функції для відстеження, контролю та

оптимізації рівня запасів. Хоча вони відрізняються за функціональністю, більшість з них дозволяють користувачам обліковувати всі вхідні та вихідні запаси, що впливає на все, від обліку та виробництва до продажів і задоволеності клієнтів.

Система управління запасами — це програмне рішення, яке дозволяє підприємствам контролювати та контролювати свої запаси від закупівлі до виконання замовлення. Він діє як централізований центр, забезпечуючи видимість у реальному часі та точні дані щодо запасів щодо рівня запасів, руху та наявності. Завдяки вдосконаленому відстеженню запасів компанії можуть оптимізувати продуктивність бізнесу, зменшити витрати та краще відповідати очікуванням клієнтів.

Основна функція модуля ланцюга постачання WMS полягає в ефективному управлінні та контролі потоку товарів на складі.

Одним із ключових аспектів цього типу системи є її здатність керувати складними процесами ланцюжка поставок. Це гарантує оптимальний розподіл запасів, швидку обробку замовлень і оперативну доставку продуктів клієнтам.

Крім того, системи керування складськими складами в модулях ланцюга постачання дозволяють підприємствам відстежувати та відстежувати продукти протягом усього шляху в ланцюжку поставок.

Порівняно з іншими типами, впровадження системи управління складським складом із модулем ланцюга постачання представляє певний набір проблем. Рівень складності може змінюватися залежно від кількох факторів, зокрема:

Міграція даних: рівень складності зростає з великими наборами даних, складними структурами даних і потребою в очищенні чи перетворенні даних.

Складність існуючого програмного забезпечення для керування ланцюгом поставок: для розрізнених систем або кількох застарілих систем встановлення вимагає більше зусиль і досвіду.

Вимоги до налаштувань: якщо систему потрібно суттєво адаптувати, для відповідного налаштування та тестування системи знадобиться додатковий час і досвід [22].

Обсяг інтеграції: якщо інтеграція вимагає складного відображення даних, складного узгодження процесу та обміну інформацією в режимі реального часу, процес інсталяції може бути складнішим.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Проєктування архітектури та структури системи

Операції на складі контролюються системою управління та контролю складу (WMCS). Щоб досягти високої продуктивності складу, WMCS повинна ефективно використовувати обмежені ресурси складу. WMCS зазвичай розглядається як багаторівнева система. Наприклад, Тен Хомпель і Шмідт [8] виділяють кілька рівнів управління WMCS. До них належать стратегічний контроль з боку системи планування(табл 2.1) ресурсів підприємства (ERP), функціональний контроль з боку керівництва складу.

Таблиця 2.1 – Рівна системи керування складом

Планування ресурсів підприємства (ERP)
Планування (запаси, замовлення та ресурси)
Планування (час виконання завдань і балансування навантаження на ресурси)
Управління матеріальними потоками (MFC)
Система обробки матеріалів (MHS)

(WMS), управління дільницями - системою управління складом (WCS), а управління процесами - контролером матеріального потоку (MFC).

Верхній рівень - це ERP-система, яка відповідає за управління замовленнями та запасами на високому рівні. Другий рівень, рівень планування, займається розподілом замовлень між запасами та ресурсами обладнання. Рівень планування також відповідає за підтримання достатнього рівня запасів шляхом поповнення низького рівня запасів і переміщення надлишкових запасів. Рівень планування відповідає за збалансування системи для досягнення оптимальної продуктивності системи. До обов'язків рівня планування входить визначення пріоритетності завдань, які були призначені ресурсам. Планування відбирає завдання і пересилає їх на рівень MFC, який

контролює складське обладнання і надає інформацію про стан обладнання на рівень планування. Рівень MFC забезпечує інтерфейс до нижнього рівня - системи обробки вантажів (MHS), яка містить складське обладнання, що забезпечує фактичне виконання складських завдань.

Оскільки модульність і вільний зв'язок є важливими характеристиками систем, побудованих з компонентів багаторазового використання, ми дозволили собі надихнутися нещодавніми дослідженнями систем управління на основі агентів. Такі системи складаються з набору автономних агентів, кожен з яких має обмежену сферу застосування. Прикладами зі сфери складування є роботи Кіма та ін. [4] і Грейвза та ін. [3]. Вони досліджують агентну WMCS для складів з концепцією комплектування «людина-товар» та «товар-людина» відповідно. Вони пропонують ієрархічну агентну WMCS, в якій агенти високого рівня обчислюють глобальний графік. Агенти нижчого рівня постійно переплановують свою роботу відповідно до умов реального часу. Якщо вони знаходять кращий розклад, вони ведуть переговори з агентами високого рівня про зміну розкладу. Агенти високого рівня дозволять запропоновану зміну розкладу, якщо новий розклад не погіршить глобальний розклад. Результати Кіма та ін. [4] і Грейвса та ін. [3] показують, що постійне перепланування розкладу агентами низького рівня позитивно впливає на продуктивність системи. Хоча агенти WMCS не володіють повними знаннями про систему, вони мають таку ж продуктивність, як і оригінальна WMCS за нормальних обставин, і кращу продуктивність у випадку винятків.

Децентралізоване управління також застосовується в галузях, схожих на сферу складування. Інші приклади децентралізованого контролю в основному стосуються виробничої сфери. Існує багато публікацій, що описують переваги децентралізованих виробничих систем. Прикладом може слугувати лінія з виробництва двигунів, описана Флітвудом та ін. [2], які представляють децентралізовану систему управління, що має кращу продуктивність і гнучкість, ніж її централізований аналог.

Багато застосувань децентралізованих систем управління є дуже специфічними для конкретного застосування. Проте, для децентралізованих систем управління було запропоновано кілька еталонних архітектур. Прикладами є PROSA [9] та ADACOR [5]. Застосовність цих еталонних архітектур була продемонстрована на прикладі різноманітних застосувань у виробничій сфері. Ці еталонні архітектури покращують можливість повторного використання завдяки визначенню загальних ролей і відповідних протоколів взаємодії. В рамках проекту Falcon Монева та ін. [7] визначили еталонну архітектуру WMCS, яка стандартизує функціональність WMCS за допомогою ролей і протоколів взаємодії, подібних до PROSA. Ці загальні ролі та протоколи взаємодії дозволяють виконувати складські операції за допомогою набору агентів, організованих в ієрархію, що відповідає базовій системі обробки матеріалів.

Ми представимо еталонну архітектуру WMCS, яка стандартизує не лише компоненти та їхні інтерфейси, але й функціональність компонентів. Це досягається завдяки використанню загальних моделей поведінки з локальними плагінами бізнес-правил, які можуть бути використані для того, щоб зробити поведінку специфічною для конкретного додатку.

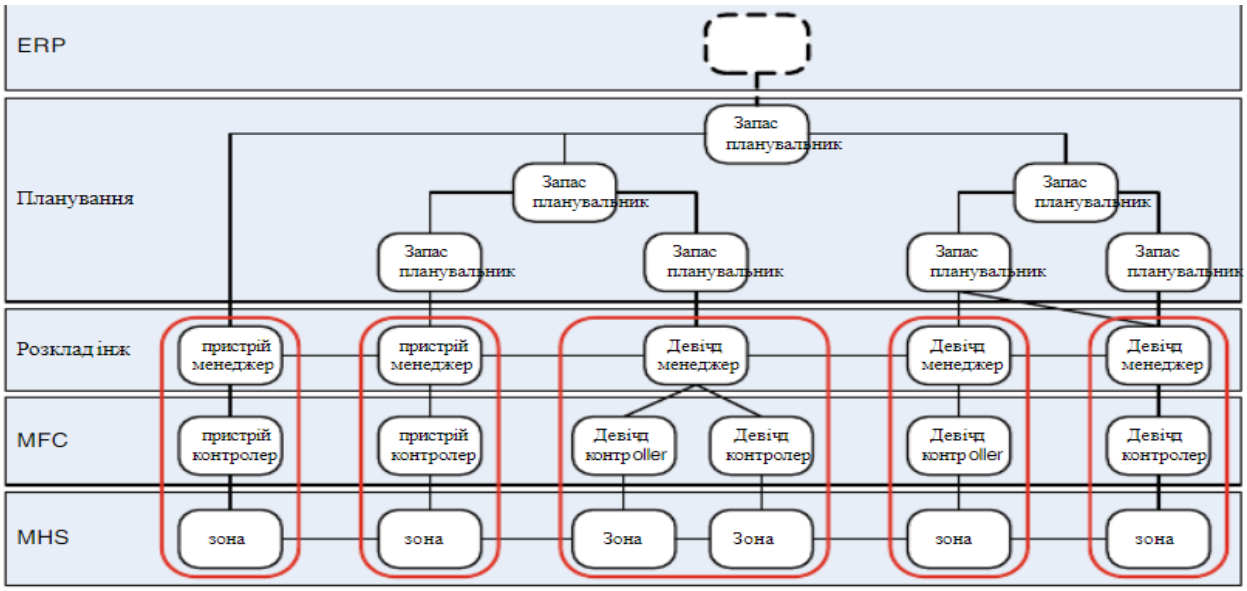


Рисунок 2.1 - Еталонна архітектура WMCS

Еталонна архітектура WMCS розрізняє чотири типи компонентів, кожен з яких відповідає своєму рівню ієрархії WMCS: планувальники запасів, менеджери пристроїв, контролери пристроїв і зони (обробки матеріалів).

Планувальники запасів знаходяться на рівні планування. Основною відповідальністю планувальників запасів є доставка товарів своїм (місцевим) клієнтам. Ця служба доставки визначається підключеними до неї менеджерами пристроїв, які визначають, які типи товарів можуть бути доставлені з локального запасу планувальника запасів. На рисунку 2.1 показано, що планувальники запасів утворюють дерево. Ця деревоподібна структура дозволяє логічно групувати планувальників запасів, наприклад, на основі типу товарів, які вони можуть постачати.

Менеджери пристроїв на рівні планування пов'язують абстрактні планувальники запасів на рівні планування з конкретними контролерами пристроїв на рівні MFC. Вони відповідають за послідовність виконання завдань, призначених планувальниками запасів. Після призначення або завершення завдання, диспетчер пристроїв вибирає наступне завдання для виконання і пересилає його відповідному контролеру пристроїв.

Ця мережа є абстракцією топології складу на рівні MHS. Ця абстракція використовується для визначення того, як передавати роботу на рівні планування, щоб отримати бажані потоки товарів через склад. Зв'язки в цій мережі можна розглядати як відносини виробник-споживач між диспетчерами пристроїв.

Контролери пристроїв на рівні MFC відповідають за координацію виконання завдань зонами обробки матеріалів. На рисунку 2.1 показано, що між зонами і контролерами пристроїв існує однозначна відповідність. Також показано, що кожен контролер пристрою підключений до одного диспетчера пристроїв, але диспетчер пристроїв може бути підключений до декількох контролерів пристроїв. Тоді диспетчер пристроїв відповідає за розподіл роботи між відповідними контролерами пристроїв.

Зони на рівні MHS відповідають за фактичне виконання складських завдань. Прикладами зон обробки матеріалів є міні-навантаження, робочі місця для комплектування замовлень і транспортні контури.

Як зони, так і контролери пристроїв є специфічними для обладнання, а отже, компонентами багаторазового використання. У нашій еталонній архітектурі WMCS планувальник запасів і диспетчери пристроїв є загальними компонентами. Обидва типи компонентів мають параметри, з яких можна побудувати структуру, як показано на рис. 2.1. Параметри планувальників запасів включають їхніх батьків і нащадків у дереві планувальників запасів і підключених до них диспетчерів пристроїв. Параметри диспетчерів пристроїв включають сусідів у мережі диспетчерів пристроїв і можливості базових контролерів пристроїв.

2.2. Проєктування логічної моделі бази даних

Логічна модель даних встановлює структуру елементів даних і зв’язки між ними. Це не залежить від фізичної бази даних, яка деталізує, як дані будуть реалізовані. Логічна модель даних служить планом для використаних даних. Логічна модель даних робить елементи концептуального моделювання даних на крок далі, додаючи до них більше інформації.

Логічна модель даних включає в себе всі елементи інформації, які є життєво важливими для ведення повсякденного бізнесу.

Компоненти логічної моделі даних

Логічна модель даних складається з трьох основних компонентів:

Сутності: кожна сутність представляє набір речей, осіб або концепцій, пов’язаних із бізнесом

Відносини: кожен зв’язок являє собою асоціацію між двома з перерахованих вище об’єктів

Атрибути: кожен атрибут є описовою частиною, характеристикою або будь-якою іншою інформацією, яка є корисною для подальшого опису сутності

Кожному з цих компонентів логічної моделі даних присвоєно назву та текстове визначення. Вони служать для постійного документування бізнес-правил і окреслення вимог до інформації. Однак наведені вище компоненти обмежуються лише описом бізнес-вимог. Їх не хвилює те, як ці бізнес-вимоги обробляються, реалізуються або зберігаються.

Необхідність логічної моделі даних

З огляду на те, що дані втілюють найважливіший аспект будь-якої програми, програми чи системи, якісні системи обробки та зберігання даних мають бути побудовані на надійній і точний базовій структурі даних. Надійна структура даних дає розробникам додатків свободу створювати найкращий інтерфейс користувача, систему обробки або налаштування статистичного аналізу та звітності.

Незалежно від того, наскільки елегантною чи технічною є ваша система, вона має відповідати вимогам, дотримуватися правил і служити цілям бізнесу чи підприємства, для якого вона створена, інакше вона не матиме практичної користі. Тому логічне моделювання даних об’єднує дві найважливіші основи розробки додатків:

– бізнес вимоги;

– якість структури даних.

Характеристика логічної моделі даних

Ось найважливіші характеристики логічної моделі даних:

Логічна модель даних може описати потреби в даних для кожного окремого проекту. Тим не менш, він розроблений для бездоганної інтеграції з іншими логічними моделями даних, якщо цього вимагає проект.

Логічна модель даних може бути розроблена та спроектована незалежно від системи керування базами даних. Тип системи керування базами даних не так сильно впливає.

Атрибути даних містять типи даних із точною довжиною та точністю.

У логічному моделюванні даних не визначено первинний або вторинний ключ. На цьому рівні моделювання даних потрібно перевірити та налаштувати деталі з'єднувача, які було встановлено до визначення зв'язків.

Логічна модель даних схожа на графічне представлення інформаційних вимог бізнес-сфери. Це сама по собі не база даних або система керування базами даних.

Логічна модель даних не залежить від будь-якого фізичного пристрою зберігання даних, наприклад файлової системи.

Логічна модель даних повинна бути розроблена так, щоб вона була незалежною від технології, щоб на неї не впливали швидкі зміни в технології.

Деталі логічного моделювання даних

Модель даних, у двох словах, — це набір специфікацій даних і діаграм, пов'язаних з поясненням вимог до даних і відповідних дизайнів. Загалом існує три типи типів моделювання даних і дій:

Концептуальна модель даних

Ця модель даних в основному визначає, що містить система. Бізнес-стейкхолдери та архітектори даних, як правило, створюють концептуальні моделі даних з наміром організувати та визначити різні бізнес-концепції та правила, а також встановити їх параметри або обсяг.

Логічна модель даних

Логічна модель даних служить для визначення того, як система повинна бути реалізована незалежно від використовуваної системи керування базою даних. Розробниками логічної моделі даних зазвичай є архітектори даних і бізнес-аналітики. Метою створення логічної моделі даних є розробка високотехнічної карти основних правил і структур даних.

Фізична модель даних

Фізична модель даних стосується того, як буде реалізована система, і факторів конкретної системи керування базами даних. Ця модель зазвичай створюється розробниками. Ідея полягає більше в тому, щоб визначити, як фактична база даних буде використовуватися або реалізована для бізнес-цілей.

Загалом кажучи, як концептуальне моделювання даних, так і логічне моделювання даних є видами діяльності «аналізу вимог», тоді як фізичне моделювання даних вважається діяльністю з розробки.

Логічна модель даних служить основою для фізичної моделі даних, яка включає бізнес-вимоги та збирає метадані. Логічне моделювання даних можна виконати за допомогою стандартних методів і нотацій моделювання даних.

Моделювання даних — це діяльність, спрямована на організацію семантики даних, опис даних і вирішення обмежень узгодженості даних. Це можна порівняти з кресленням архітектора або схемою будівлі, яка є основою для концептуального моделювання та встановлює зв’язки між різними компонентами даних.

Методи моделювання даних поділяються на одну з двох категорій:

Модель зв'язку сутностей (ER).

UML (уніфікована мова моделювання)

Логічне моделювання даних належить до моделі зв’язків сутностей, створеної за допомогою діаграми зв’язків сутностей (відомої як ERD), стандартної техніки моделювання, яка використовується як засіб зв’язку розробниками моделей даних у всьому світі. У ньому міститься повний набір бізнес-вимог, але не технічні компоненти.

Переваги логічної моделі даних

Оскільки дані залишаються стабільними з часом, логічна модель даних також є стабільною та дуже сприятливою для повторного використання даних і фізичного обміну даними, що в кінцевому підсумку призводить до зменшення обсягу зберігання надлишкових даних.

Компоненти логічної моделі даних можна переробляти, повторно використовувати та адаптувати, коли більше команд зважують свої (часто мінливі) потреби.

Витрати, пов’язані зі створенням і підтримкою логічної моделі даних, у довгостроковій перспективі компенсуються перевагами, які вона надає, не в останню чергу шляхом визначення та інтеграції всіх бізнес-потреб і правил на початку.

Компоненти процесу створення, а саме проектування, кодування, тестування та розгортання, відбуваються швидше, як прямий результат інтеграції та уточнення бізнес-правил.

Наявність логічної моделі даних полегшує і, отже, економічно вигідно вносити зміни, виправляти помилки або вводити відсутні дані під час самого життєвого циклу розробки до впровадження.

Запити користувачів на внесення змін можна мінімізувати, якщо бути проактивним.

Логічні моделі даних можна використовувати для аналізу впливу, оскільки кожен бізнес-процес і правило пов’язані між собою.

Оскільки об’єкти в логічній моделі даних містять текстові визначення діловою мовою, це полегшує підтримку системної документації та доступ до неї.

Простіше кажучи, проблеми можуть бути. Користувачі можуть захопитися процесами та діяльністю, якщо їм не нагадувати про необхідність виділяти дані, а не технології, як ключовий інгредієнт під час розробки нової системи. Розробка моделі даних виключно на основі фізичного робочого процесу втрачає представлення важливих бізнес-вимог.

Таблиці та файли, які створюють дизайнери, не маючи елементів даних, окреслених відповідно до бізнес-вимог, як правило, погано організовані та не мають надійної базової структури. Виявлення та спроба врахувати додаткові елементи даних з макетів екрана чи звіту під час процесу кодування,

тестування або навіть під час розгортання змушує розробників реагувати, а не діяти на попередження. Результатом є комбінована сутність, якою важко керувати або підтримувати, вона повна помилок або зайвого тексту, без системної документації, забирає час і, можливо, непридатна для використання.

Оскільки логічна модель даних визначає структуру елементів даних на основі основних бізнес-вимог, а також зв’язків між ними, відсутність логічної моделі даних означає багато втрачених можливостей для вдосконалення бізнес-процесів. Розробники просто автоматизують існуючі процедури або відтворюють застарілі системи на новішій технологічній платформі, яка з часом може застаріти.

Застосування логічного моделювання даних дозволяє аналітикам даних мислити незалежно від новітніх технологій і зосереджуватися на вдосконаленні бізнес-процесів.

Тому логічна модель даних повинна стати життєво важливим і незмінним компонентом кожного проекту розробки додатків. Це важливий крок, який в ідеалі має передувати розробці бази даних.

WMS забезпечує роботу цілого складу, де управління запасами є однією з основних складових. Простіше кажучи, управління складом — це широкий термін, що стосується всіх процесів усередині складу, тоді як управління запасами — це вузька спеціалізація в процесі ланцюжка поставок

Оскільки WMS та IMS можуть бути двома різними та автономними програмними рішеннями, давайте розглянемо ключові функції, які вони можуть містити.

WMS включає:

Проектування складу: Оптимізація розміщення запасів і забезпечення індивідуального робочого процесу, а також точний вибір способів комплектування.

Відстеження запасів: використання спеціальних методів, таких як пристрої з підтримкою Інтернету речей, RFID, штрих-коди, які дозволяють автоматично визначати поточне місцезнаходження предметів.

Керування завданнями: відстеження ефективності працівників за допомогою налаштованих показників ефективності, які називаються KPI.

Планування доків і керування майданчиком: дозволяйте водіям транспортних засобів швидко знаходити потрібний вантажний майданчик і допомагайте їм у крос-докінгу.

Отримання та розміщення товарів: контроль над складанням і пошуком інвентарю за допомогою підходу «підбирання до світла» (P2L) (вибір предметів зі складських полиць за допомогою освітлення для керування операторами). До речі, у звіті зазначено, що системи P2L можуть підвищити продуктивність підбору на 30–50%.

Процес комплектування та пакування: забезпечення більш ефективної операції комплектування та пакування товарів операторами складу.

Процедури відвантаження: Створення транспортних накладних, пакувальних листів і рахунків-фактур на доставку, відправка повідомлень про відвантаження.

Заходи з покращення: Створення звітів на основі ключових KPI, складських операцій для пошуку прогалин для вдосконалення.

Оскільки IMS є частиною WMS, деякі з їхніх функцій збігаються, включно з комплектуванням і упаковкою, доставкою, відстеженням запасів тощо. Але є також деякі більш специфічні особливості, про які варто згадати.

IMS також включає:

Оптимізація розташування товарів: забезпечення правильного розміщення товарів на складі, їх легкого пошуку та доступу, не займаючи зайвого місця.

Виконання замовлень: приймання вхідних замовлень і їх підготовка до виконання.

Циклічний підрахунок: використання спеціальної техніки аудиту запасів, яка дозволяє щодня підраховувати запаси невеликими порціями, щоб загальна сума завжди була актуальною.

Відстеження активів за допомогою штрих-кодів: Забезпечення сканування штрих-кодів товарів та інтеграція цих даних із відправленнями, фінансовими процедурами та іншими операціями.

Збір даних і створення звітів: збір і аналіз даних для створення інтелектуальних заходів для підвищення продуктивності та досягнення більш точного прийняття рішень.

Загалом, різниця між WMS та IMS залежить від компанії-співробітника. Системи управління складами характерні для великих підприємств, що володіють хоча б одним гігантським складом або декількома в різних містах і навіть країнах. Навпаки, SMB віддають перевагу системам управління запасами, оскільки їм не потрібні всі можливості багатофункціональної WMS.

Підприємства можуть працювати з багатьма типами запасів, головним чином залежно від продуктів, які продає певна компанія. Найпоширеніша класифікація включає:

Сировина: товари, які компанія використовує для виробництва готової продукції.

Незавершені товари: ці продукти знаходяться на стадії створення та стають готовими до наступного етапу.

Кінцева продукція: кінцева продукція (або готова продукція) – це постачання, готове до продажу кінцевим споживачам.

Матеріали для технічного обслуговування, ремонту та експлуатації: у виробничому процесі використовуються предмети, які в кінцевому підсумку не видно в самих готових виробах.

Інвентар безпеки: це додатковий запас, який зберігається компанією для вирішення проблеми з нестачею постачальників або коливаннями попиту.

Дослідження поточного ринку ланцюгів постачання показують, що окрім простого підвищення ефективності складу використання інтегрованого програмного забезпечення для управління запасами забезпечує додатковий прибуток для ланцюгів постачання. Наприклад, це може допомогти вам оптимізувати та збільшити простір для зберігання та звільнити місце для нових продуктів. Це також дозволяє ефективно використовувати запаси та краще узгоджувати попит і пропозицію.

Підприємства можуть підвищити ефективність на 25%, збільшити вільний простір на 20% і збільшити продуктивність використання запасів на 30%, якщо вони вдосконалять систему зберігання запасів шляхом переходу на більш автоматизовану форму обробки замовлень.

Більше третини експертів з ланцюгів поставок стверджують, що оптимізація управління запасами вкрай необхідна для збалансування попиту та пропозиції.

Ось деякі інші причини, чому програмні рішення для управління запасами є вирішальними для побудови потужного ланцюжка поставок у вашій компанії та сприяння її успіху в довгостроковій перспективі:

Задоволені та продуктивні працівники. Завдяки WMS та IMS кожен оператор на складі має певний набір обов’язків на кожен день. Це робить їхній робочий день насиченим і, як наслідок, працівники відчувають задоволення від виконаної роботи.

Вищі показники утримання працівників. Без точного управління складські операції приречені на безлад і хаос. Це змушує працівників звільнятися з роботи. Інтегрувавши спеціалізовані рішення WMS або IMS, ви можете зменшити оборот.

Краще відстеження запасів. Знову ж таки, неможливо впоратися з усім вручну. Це може призвести до втрати, недоречності або навіть пошкодження товарів. Управління запасами пропонує інноваційні методи точного відстеження, такі як QR-коди, RFID, пристрої, підключені до Інтернету, тощо.

Вони допомагають визначити точне місцезнаходження та відстежувати всі продукти та бути готовими до відправлення до клієнтів.

Покращена ефективність підбору та упаковки/відвантаження. Коли розподіл предметів правильний, кожен оператор знає, як і де знайти потрібний предмет, ефективність зростає. Процеси вибору, пакування та доставки стають легкими, безперебійними та не забирають багато часу.

Зменшені витрати. Неефективність на складі може призвести до значних грошових втрат. Чим більше часу ви витрачаєте на підготовку продукції до відправлення, тим дорожче обходиться обробка замовлення. Найкраще витратити більше на організацію запасів і повернути вигоди в довгостроковій перспективі.

Більше довіри з боку партнерів. Постачальники та сторонні організації частіше співпрацюватимуть із компаніями, які мають належним чином організований та ефективний ланцюг поставок. Вибір інтеграції системи управління складськими запасами може бути правильним рішенням, яке порадує ваших партнерів.

Підвищена лояльність клієнтів. Сучасні споживачі вимогливі і хочуть отримувати посилку максимум за пару днів. Тому швидкість виконання замовлення є життєво важливою для того, щоб вони залишалися задоволеними. IMS може допомогти в досягненні цієї мети.

На елементі коду нижче створимо базу даних для додатку управління складом та побудуємо логічно модель зв’язку(рис 2.2)

```
-- Створення бази даних
CREATE DATABASE WarehouseDB;
GO
-- Використання новоствореної бази даних
USE WarehouseDB;
GO
-- Створення таблиці Suppliers
```

```
CREATE TABLE Suppliers (
    SupplierID INT PRIMARY KEY IDENTITY(1,1),
    SupplierName NVARCHAR(100) NOT NULL,
    ContactName NVARCHAR(100),
    Address NVARCHAR(255),
    City NVARCHAR(100),
    PostalCode NVARCHAR(20),
    Country NVARCHAR(50),
    Phone NVARCHAR(50)
);
GO

-- Створення таблиці Customers
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY IDENTITY(1,1),
    CustomerName NVARCHAR(100) NOT NULL,
    ContactName NVARCHAR(100),
    Address NVARCHAR(255),
    City NVARCHAR(100),
    PostalCode NVARCHAR(20),
    Country NVARCHAR(50),
    Phone NVARCHAR(50)
);
GO

-- Створення таблиці Products
CREATE TABLE Products (
    ProductID INT PRIMARY KEY IDENTITY(1,1),
    ProductName NVARCHAR(100) NOT NULL,
```

```
    SupplierID INT,
    Category NVARCHAR(50),
    QuantityPerUnit NVARCHAR(50),
    UnitPrice DECIMAL(10, 2),
    UnitsInStock INT,
    UnitsOnOrder INT,
    ReorderLevel INT,
    Discontinued BIT,
    FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)
);
GO

-- Створення таблиці Inventory
CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY IDENTITY(1,1),
    ProductID INT,
    Quantity INT,
    Location NVARCHAR(100),
    LastUpdated DATETIME,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
GO

-- Створення таблиці Orders
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY IDENTITY(1,1),
    CustomerID INT,
    OrderDate DATETIME NOT NULL,
    ShipDate DATETIME,
```

```
    ShipAddress NVARCHAR(255),
    ShipCity NVARCHAR(100),
    ShipPostalCode NVARCHAR(20),
    ShipCountry NVARCHAR(50),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);
GO

-- Створення таблиці OrderDetails
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY IDENTITY(1,1),
    OrderID INT,
    ProductID INT,
    UnitPrice DECIMAL(10, 2),
    Quantity INT,
    Discount DECIMAL(5, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
GO
```

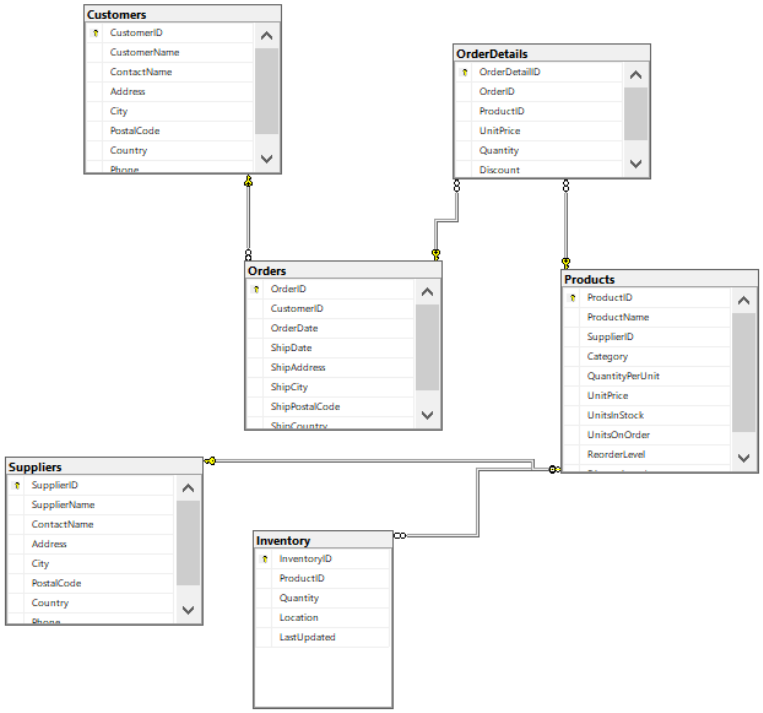


Рисунок 2.2- Логічна схема даних бази для складу

2.3. Проєктування інтерфейсу користувача

Визнання поточної господарської діяльності на складі має найбільш значний вплив на отримання прибутку та має вирішальне значення для продуктивності компанії. Компанія повинна знати, які функції програмного забезпечення Warehouse Management System (WMS) найбільше впливають на продуктивність. Складність бізнес-факторів слід розглядати з точки зору багатьох технічних аспектів, таких як обсяг повсякденної діяльності, необхідний ступінь залучення персоналу, витрати протягом життєвого циклу запасів і координація ланцюга постачання

Керівники складів розглянуть багато стратегічних питань щодо обсягу програмного забезпечення WMS. Вони стосуватимуться найбільш

прийнятного рішення, призначеного для складських операцій, незалежно від галузі чи інтеграції з розподільчим чи виробничим підприємством, і мають посилатися на конкретні функції WMS Функції WMS тісно пов'язані з внутрішніми процесами.

Технічні аспекти можливостей системи управління складом будуються на основі трьох основних компонентів:

Програмне забезпечення: це цифрова система, яка працює на основі обміну даними ERP, що дозволяє розширювати ERP додатковою транзакцією та забезпечувати дистанційне керування через спеціальні програми для мобільних і настільних пристроїв.

Пристрої радіозв’язку: для зв’язку між серверами та системою ERP для мобільних пристроїв у всьому складі потрібна система бездротової мережі.

Система штрих-кодів: програмне забезпечення WMS працює за допомогою сканерів RFID із функцією зчитування штрих-кодів/QR. Це фундаментальний елемент збору даних, і всі операції обробного пристрою покладаються на нього для безперервного надання даних системі.

Загальні критерії вибору програмного забезпечення Warehouse Management System (WMS) визначають кілька ключових елементів, які потребують оцифрування складських операцій. Функціональність системи управління складом включає ті області, які мають охоплювати програмне забезпечення разом із його функціями.

Система одиниць обробки

Отримання: реєстрація надходження товарів у документах за допомогою мобільних сканерів на базі додатку Warehouse Management System (WMS).

Прийом на зберігання: приписування вхідних товарів інвентарю по всьому складу за допомогою мобільних пристроїв на основі WMS.

Поповнення: переміщення товарів із вищих стелажів на нижні, переміщення запасів, що старіють, за допомогою мобільних додатків WMS на мобільних пристроях.

Підбір/пакування замовлень: підбір замовлень за допомогою мобільних пристроїв на основі додатків WMS для планування шляхів підбору та скорочення процесів.

Голосовий підбір: інтеграція голосової системи з WMS і ERP, щоб дозволити підбір замовлень без рук.

Система маркування: Інтеграція системи друкуючого обладнання з WMS і мобільними пристроями.

Послуга з доданою вартістю (VAS): оснащення пакувальних ліній VAS портативними сканерами та настільними програмами на основі WMS для відстеження введеного матеріалу.

Доставка: обробка вихідних відправлень за допомогою мобільних додатків на основі WMS для точної обробки відправлень і швидкого завантаження.

Управління запасами

Внутрішні переміщення: програмне забезпечення WMS надає через настільні та мобільні додатки можливість контролювати запаси за допомогою розподілу/переміщення та підтримує управління старінням запасів тощо.

Підрахунок циклу: персонал може проводити періодичний підрахунок запасів на основі мобільних додатків WMS, встановлених на мобільних пристроях.

Система зберігання : програмне забезпечення WMS забезпечує підтримку системи зберігання піддонів, системи маркування стелажів і класифікації товарів, що дозволяє негайно розподіляти товари за необхідними зонами.

Контроль інвентаризації: контролери інвентаризації підтримуються мобільними додатками на основі WMS для негайного вирішення розбіжностей інвентаризації.

Управління небезпечними вантажами: користувачі можуть бути попереджені WMS про те, що вони мають справу з вантажами DGR, оскільки

вони потребують належного зберігання, часто в холодильниках або ізольованих зонах з високим рівнем безпеки.

Товари широкого вжитку (FMCG): WMS може керувати товарами, чутливими до температури, і товарами з особливими вимогами до зберігання.

Управління персоналом

Планування змін персоналу: Система управління складом (WMS) дозволяє планувати зміни, оцінюючи необхідне обладнання для виконання конкретних завдань.

Управління робочим часом: менеджери та керівники за допомогою програмного забезпечення на основі WMS можуть відстежувати точні години, призначені для виконання певних завдань на складі.

Керування сертифікатами: програмне забезпечення WMS дозволяє відстежувати періодично оновлювані ліцензії та сертифікати персоналу для складського обладнання, такого як вантажівки, вилкові навантажувачі, комбіновані вантажівки тощо.

Управління робочими умовами: програмне забезпечення WMS може надати систему для розпізнавання умов праці в конкретних складських приміщеннях і сповіщення персоналу про будь-які небезпеки або вимоги.

Управління поверненнями

Управління поверненням упаковки: програмне забезпечення WMS підтримує систему маркування та дозволяє відстежувати цикл упаковки повернення упаковки, виключаючи з процесу пошкоджені коробки тощо.

Система повернення товарів (система RMA): WMS обробляє повернені товари, які потребують оцінки якості перед надходженням на склад. WMS має дозволити впровадження додаткових транзакцій контролю якості, окрім тих, які доступні в ERP.

Управління ключовими показниками ефективності: система управління складом (WMS) дозволяє отримувати всі дані з усіх відділів, забезпечуючи

прозорий і регульований огляд, який допомагає менеджерам контролювати загальні процеси.

Звітність: за допомогою вбудованих функцій WMS менеджери та офісні адміністратори можуть створювати щотижневі звіти для відстеження поточних процесів.

Прогнозування: на основі програмного забезпечення WMS персонал може створювати звіти, щоб передбачити певні події на складі та уникнути збоїв у процесі.

Управління ланцюгом поставок

Система доставки: WMS дозволяє відстежувати відправлення від дверей до складу та навпаки на основі функції підтвердження доставки (POD).

Зв’язок із перевізниками через систему EDI: програмне забезпечення WMS забезпечує зв’язок із системами управління транспортуванням (TMS) сторонніх перевізників, полегшуючи оновлення даних про відправлення тощо.

Фінансовий менеджмент

Система виставлення рахунків: на основі системи WMS усі відділи можуть збирати всі фінансові дані. Зовнішні треті сторони можуть надсилати рахунки до внутрішніх систем.

Глибока інтеграція ERP

Підключення існуючих систем: програмне забезпечення WMS може інтегрувати всі системи в організації, одержуючи необхідні дані з усіх відділів (таких як система управління транспортуванням (TMS), системи звітності та фінанси, операційні процеси тощо).

Інтеграція ERP: програмне забезпечення WMS має бути пов’язане з системою ERP як додатковий рівень для збільшення кількості транзакцій, надаючи персоналу всієї організації пряме уявлення про поточні операції.

Програма навчання автономності, гнучкості та експлуатації

Простий інтерфейс: мобільні програми WMS можуть забезпечити простоту використання та скоротити час навчання, спрощуючи інтерфейс користувача на мобільному пристрої для співробітника операцій.

Автономне сховище: програмне забезпечення WMS дозволяє внутрішньому персоналу створювати власні системні програми без зовнішньої підтримки.

Гнучкість: програмне забезпечення WMS може надавати функціональні можливості на основі BPM які дозволяють додавати нові системні транзакції та підтримувати операційні зміни на льоту.

Нуль інформаційних помилок: Усуньте інформаційні помилки та підвищте точність інвентаризації.

Зменшення часу отримання інформації: скорочення часу та зусиль, необхідних для збору та обробки даних про запаси та замовлення клієнтів.

Збільшення місткості складу: збільште місткість складу, дозволивши продуктам протікати через склад швидше.

Оптимізоване використання простору: оновлюйте інформацію для працівників складу про те, куди розміщувати продукти в кожному місці на основі відомих розмірів, оптимізуючи використання простору.

Підвищення продуктивності праці: підвищуйте продуктивність праці, керуючи завданнями складських працівників і відповідно плануючи робочий час.

Мінімізація складських втрат: зменшіть розбіжності в запасах і отримайте кращий контроль над складським попитом, мінімізуючи втрати.

Інтеграція з ERP: Novacura Flow полегшує інтеграцію всіх внутрішніх систем з ERP на основі API та з'єднувачів IoT для машин, щоб отримувати більше операційних даних із цеху після підключення складу до виробничого підприємства.

Програмний застосунок буде розрозроблений мовою C# у середовищі visual studio 2022 використовуючи технології WPF

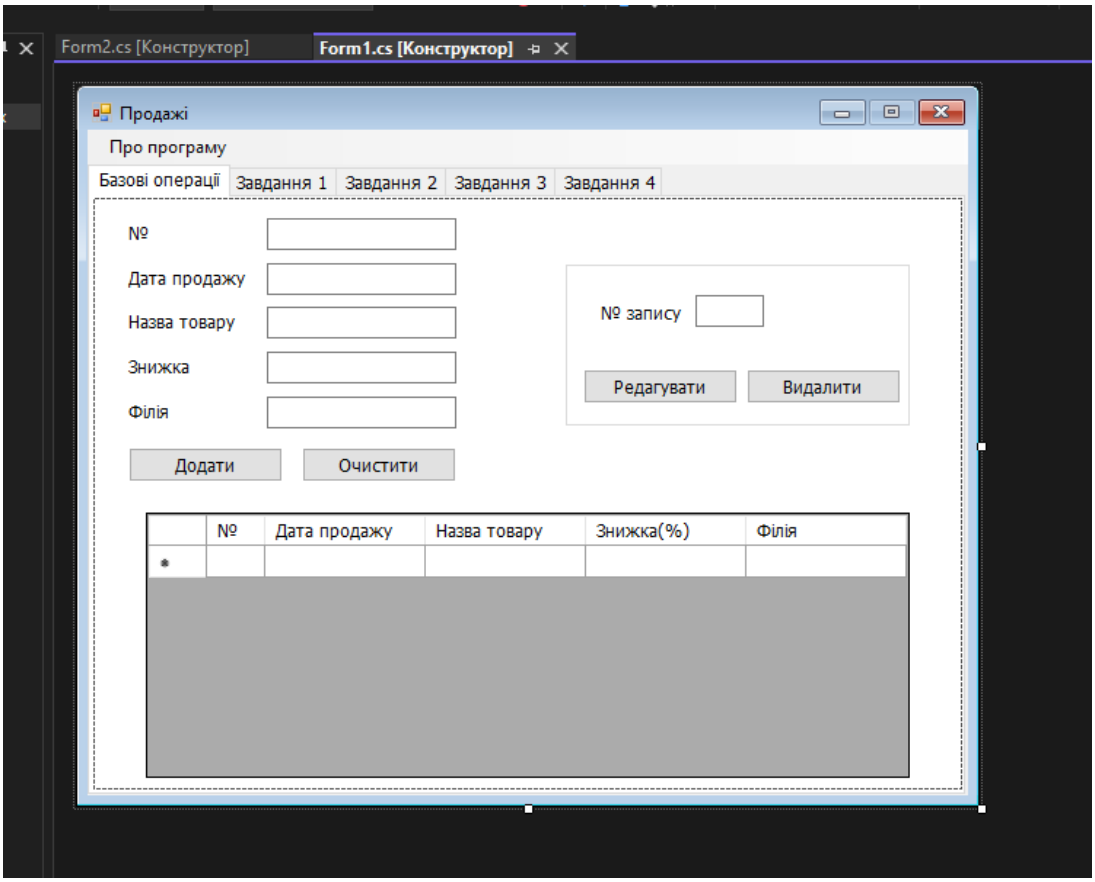


Рисунок 2 3-Головне меню програми

2.4. Аналіз та вибір технологій і методів реалізації

C# — це об’єктно-орієнтована мова програмування, створена корпорацією Майкрософт для розробки програм на платформіNET. Це дозволяє розробникам використовувати широкий спектр бібліотек і інструментівNET для створення потужних високоякісних програм. Коротше кажучи, C# — це потужна та універсальна мова програмування яка стала важливим інструментом для розробки програмного забезпечення на платформіNET

C# — це проста сучасна об’єктно-орієнтована мова, яка походить від C++ і Java.

Він покликаний поєднати високу продуктивність Visual Basic і сиру потужність C++.

Це частина Microsoft Visual Studio 7.0.

Visual Studio підтримує VB, VC++, C++, VBscript, JScript. Усі ці мови забезпечують доступ до платформи MicrosftNET

.NET включає механізм загального виконання та багату бібліотеку класів.

Microsoft JVM eqiv — це загальномовне середовище виконання (CLR).

CLR підтримує більше ніж одну мову, наприклад C#, VB.NET, Jscript, ASP.NET, C ++.

Вихідний код --->Проміжний код мови (IL) ---> (компілятор JIT) Власний код.

Класи та типи даних є спільними для всіх мовNET.

Ми можемо розробляти консольну програму, програму Windows і веб-програму на C#.

У C# Microsoft подбала про проблеми C++, такі як керування пам’яттю, покажчики тощо.

Він підтримує збирання сміття, автоматичне керування пам'яттю та багато іншого.

У C# відсутні покажчики.

Небезпечні операції, такі як прямі маніпуляції з пам’яттю, заборонені.

У C# не використовуються оператори «::» або «->».

Оскільки він знаходиться наNET, він успадковує функції автоматичного керування пам’яттю та збирання сміття.

Різні діапазони примітивних типів, таких як Integer, Floats тощо.

Цілі значення 0 і 1 більше не приймаються як логічні значення. Логічні значення є чистими значеннями true або false у C#, тому більше немає помилок "="operator і "=="operator.

"==" використовується для операції порівняння, а "=" використовується для операції присвоєння.

C# базується на поточних тенденціях і є дуже потужним і простим для створення сумісних, масштабованих, надійних програм.

C# містить вбудовану підтримку для перетворення будь-якого компонента на веб-службу, яку можна викликати через Інтернет із будь-якої програми, що працює на будь-якій платформі.

C# підтримує інкапсуляцію даних, успадкування, поліморфізм, інтерфейси.

(int, float, double) не є об'єктами в java, але C# містить структури (структури), які дозволяють примітивним типам стати об'єктами.

int i=1;

string a=i Tostring(); //conversion (or) Boxing

C#

У C# ми не можемо виконувати небезпечні приведення, як-от конвертувати подвійне в логічне значення.

Типи значень (примітивні типи) ініціалізуються нулями, а довідкові типи (об'єкти та класи) ініціалізуються компілятором автоматично нулем.

масиви мають нульовий базовий індекс і перевіряються зв'язки.

Переповнення типів можна перевірити.

C# містить власну підтримку програм на основі COM і Windows.

Дозвіл обмеженого використання власних покажчиків.

Користувачам більше не потрібно явно впроваджувати невідомі та інші COM-інтерфейси, ці функції є вбудованими.

C# дозволяє користувачам використовувати покажчики як небезпечні блоки коду для маніпулювання вашим старим кодом.

Компоненти з VB NET та інших мов керованого коду та безпосередньо використовуватися в C#.

.NET представив збірки, які самоописуються за допомогою свого маніфесту. маніфест визначає ідентифікатор збірки, версію, культуру та цифровий підпис тощо. Збірки не потрібно ніде реєструвати.

Щоб масштабувати нашу програму, ми видаляємо старі файли та оновлюємо їх новими. Немає реєстрації бібліотеки динамічного компонування.

Оновлення компонентів програмного забезпечення є завданням, яке загрожує помилками. Зміни, внесені до коду. може вплинути на версії існуючої програми підтримки C# у мові. Вбудована підтримка інтерфейсів і перевизначення методів дозволяють з часом розробляти та вдосконалювати складні структури.

C# — це сучасна об’єктно-орієнтована мова програмування, безпечна для типу, яка дозволяє програмістам швидко й легко створювати рішення для платформи MicrosoftNET саме тому для реалізації застосунку було обрано цю мову програмування.

Windows Presentation Foundation (WPF) — це платформа розробки, яка використовується для створення настільних програм. Це частинаNET framework WPF має незалежний від роздільної здатності та векторний механізм візуалізації, який корисний для роботи з сучасним графічним обладнанням. Остання версія WPF – 4.6 У цьому фреймворку інтерфейс програми розроблено мовою XAML, а логіка програми написана мовою програмування C#

Особливості WPF такі:

- розширювана мова розмітки програм (XAML);
- елементи управління;
- прив'язка даних;
- макет;
- 2D і 3D графіка;
- Анімація;
- стилі;
- шаблони;

– документи;

– текст.

Архітектура WPF. Основними компонентами WPF є PresentationFramework, PresentationCore, Milcore, Common Language Runtime (CLR) User32, Kernel. Milcore написаний у некерованому коді, щоб забезпечити тісну інтеграцію з DirectX, який відповідає за відображення. WPF має точний контроль над пам’яттю та виконанням. Механізм композиції в milcore надзвичайно чутливий до продуктивності, тому для підвищення продуктивності потрібно відмовитися від багатьох переваг Common Language Runtime.

Він використовує поточний стандарт, оскільки він новіший

розробники елементів керування, швидше за все, будуть більше зосереджені на WPF, оскільки його XAML дозволяє легко створювати та редагувати ваш інтерфейс і дозволяє розділити роботу над розробкою між дизайнером (XAML) і програмістом (C#).

Прив’язка даних використовується для чіткого відокремлення даних від макета.

Ефективно використовує апаратне забезпечення для малювання інтерфейсу користувача для кращої продуктивності.

Він використовується для створення інтерфейсів користувача як для додатків Windows, так і для веб-додатків.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ

3.1. Програмна реалізація модулів

Модульна установка

У модульному інсталяторі Visual Studio ви вибираєте та встановлюєте потрібні робочі навантаження Робочі навантаження — це групи функцій, які потрібні мовам програмування або платформам для роботи. Ця модульна стратегія допомагає зменшити площу інсталяції Visual Studio, тому інсталяція та оновлення відбуваються швидше.

У Visual Studio є набір інструментів для легкого створення хмарних програм Microsoft Azure. Ви можете налаштовувати, створювати, налагоджувати, пакувати та розгортати програми та служби Azure безпосередньо з інтегрованого середовища розробки Visual Studio (IDE). Щоб отримати інструменти Azure та шаблони проектів, виберіть робоче навантаження розробки Azure під час інсталяції Visual Studio.

Вікно Cloud Explorer припинено у Visual Studio 2022. Щоб отримати додаткові відомості, перегляньте розділ Керування ресурсами, пов’язаними з обліковими записами Azure у Visual Studio Cloud Explorer

За потреби використовуйте портал Azure для доступу до ресурсів Azure. Ви можете продовжувати використовувати вузол Azure Server Explorer у попередніх версіях Visual Studio.

Ви можете використовувати служби Azure для своїх програм, додавши підключені служби наприклад:

Підключена служба Active Directory для використання облікових записів Azure Active Directory (Azure AD) для підключення до веб-програм

Підключена служба Azure Storage для зберігання blob-об’єктів, черг і таблиць

Підключена служба Key Vault для керування секретами веб-програм

Доступні підключені служби залежать від типу вашого проекту. Додайте службу, клацнувши проект правою кнопкою миші в Solution Explorer і вибравши Add > Connected Service

На екрані « Підключені служби » виберіть посилання або знак «плюс», щоб додати залежність служби На екрані « Додати залежність » виберіть службу, яку потрібно додати, і дотримуйтеся вказівок на екранах, щоб підключитися до передплати та служби Azure.

Щоб отримати додаткові відомості, перегляньте перехід до хмари за допомогою Visual Studio та Azure

Створення веб-додатків

Visual Studio може допомогти вам писати програми для Інтернету. Ви можете створювати веб-програми за допомогою ASP.NET, Node.js, Python, JavaScript і TypeScript. Visual Studio підтримує багато веб-фреймворків, таких як Angular, jQuery та Express.

ASP.NET Core iNET Core працюють в операційних системах Windows, Mac і Linux. ASP.NET Core — це велике оновлення для MVC, WebAPI та SignalR. ASP.NET Core розроблено з нуля, щоб забезпечити компактний і компонований стекNET для створення сучасних хмарних веб-програм і служб.

Щоб отримати додаткові відомості, перегляньте сучасні веб-інструменти

Створюйте кросплатформні програми та ігри

Visual Studio може створювати програми та ігри для macOS, Linux і Windows, а також для Android, iOS та інших мобільних пристроїв За допомогою Visual Studio ви можете створити:

ПрограмиNET Core які працюють у Windows, macOS і Linux.

Мобільні програми для iOS, Android і Windows у C# і F# за допомогою Xamarin

Двовимірні та 3D-ігри на C# за допомогою інструментів Visual Studio для Unity

Власні програми C++ для пристроїв iOS, Android і Windows. Поділіться загальним кодом у бібліотеках iOS, Android і Windows за допомогою C++ для кросплатформної розробки

Підключення до баз даних

Server Explorer допомагає переглядати екземпляри та активи сервера та керувати ними локально, віддалено, а також на Azure, Microsoft 365, Salesforce.com і веб-сайтах. Щоб відкрити Server Explorer виберіть « Перегляд » > «Server Explorer » Щоб отримати додаткові відомості про використання Server Explorer, див Додавання нових підключень

SQL Server Object Explorer надає перегляд об’єктів вашої бази даних, подібно до SQL Server Management Studio. За допомогою SQL Server Object Explorer ви можете виконувати легку роботу з адміністрування та проектування бази даних. Приклади включають редагування даних таблиці, порівняння схем і виконання запитів за допомогою контекстних меню.

Щоб відкрити SQL Server Object Explorer виберіть його піктограму у верхній частині вікна Server Explorer або виберіть View > SQL Server Object Explorer у верхньому меню Visual Studio.

SQL Server Data Tools (SSDT) — це потужне середовище розробки для SQL Server, бази даних SQL Azure та сховища даних SQL Azure. За допомогою SSDT ви можете створювати, налагоджувати, підтримувати та рефакторювати бази даних. Ви можете працювати з проектом бази даних або безпосередньо з підключеним екземпляром бази даних локально чи поза приміщенням. Щоб отримати SSDT, скористайтеся інсталятором Visual Studio, щоб установити робоче навантаження зберігання та обробки даних

Налагоджуйте, тестуйте та покращуйте свій код

Коли ви пишете код, ви повинні запустити його та перевірити на помилки та продуктивність. За допомогою системи налагодження Visual Studio ви можете налагоджувати код, що виконується у вашому локальному проекті, на віддаленому пристрої або в емуляторі пристрою Проходьте по коду

один оператор за раз і перевіряйте змінні по ходу. Або встановіть контрольні точки, які досягаються лише тоді, коли виконується певна умова. Ви можете керувати параметрами налагодження в самому редакторі коду, тому вам не потрібно залишати свій код.

Щоб отримати додаткові відомості про налагодження у Visual Studio, див. Перший погляд на налагоджувач

Щоб підвищити продуктивність програми, скористайтеся функцією профілювання Visual Studio

Visual Studio пропонує такі варіанти тестування як модульне тестування, живе модульне тестування, IntelliTest, а також тестування навантаження та продуктивності. Visual Studio також має розширені можливості аналізу коду для пошуку недоліків дизайну, безпеки та інших.

Розгорніть готову програму

У Visual Studio є інструменти для розгортання вашої програми для користувачів або клієнтів через Microsoft Store, сайт SharePoint або технології InstallShield або Windows Installer. Ви можете отримати доступ до всіх цих параметрів через Visual Studio IDE. Щоб отримати додаткові відомості, перегляньте розділ Розгортання програм, служб і компонентів

Керуйте своїм вихідним кодом і співпрацюйте з іншими

У Visual Studio ви можете керувати своїм вихідним кодом у сховищах Git, розміщених будь-яким постачальником, включаючи GitHub. Ви також можете знайти сервер Azure DevOps для підключення.

Щоб отримати повну інформацію, перегляньте сторінку Git у Visual Studio та навігаційну сторінку документації з керування версіями Visual Studio Щоб отримати покроковий посібник із підключення до сховища Git або Azure DevOps за допомогою Visual Studio, перегляньте сторінку « Відкрити проект зі сховища»

Ми продовжуємо створювати набір функцій Git і повторюємо його на основі ваших відгуків. Щоб отримати додаткові відомості про нещодавнє

оновлення функції, а також посилання на опитування, де ви можете поділитися своїми відгуками про це, перегляньте публікацію блогу Visual Studio про підтримку кількох сховищ

Windows Forms — це набір керованих бібліотек yNET Framework, розроблених для розробки багатофункціональних клієнтських програм. Це графічний API для відображення даних і керування взаємодією користувачів зі спрощеним розгортанням і кращою безпекою в клієнтських програмах.

Windows Forms пропонує розширену клієнтську бібліотеку, що забезпечує інтерфейс для доступу до рідних елементів графічного інтерфейсу Windows і графіки з керованого коду. Він створений з архітектурою, керованою подіями, подібною до клієнтів Windows, і, отже, його програми чекають введення користувача для свого виконання.

Windows Forms схожа на бібліотеку Microsoft Foundation Class (MFC) у розробці клієнтських програм. Він надає оболонку, що складається з набору класів C++ для розробки програм Windows. Однак він не забезпечує стандартну структуру додатків, як MFC.

Techopedia пояснює Windows Forms

Кожен елемент керування в програмі Windows Forms є конкретним екземпляром класу. Розташуванням елемента керування в графічному інтерфейсі користувача та його поведінкою керують за допомогою методів і засобів доступу. Windows Forms надає різноманітні елементи керування, такі як текстові поля, кнопки та веб-сторінки, а також параметри для створення спеціальних елементів керування. Він також містить класи для створення пензлів, шрифтів, значків та інших графічних об’єктів (наприклад, лінії та кола).

Windows Forms Designer — це інструмент у Visual Studio.NET, який використовується для вставлення елементів керування у форму та їх розташування відповідно до бажаного макета з можливістю додавання коду для обробки їхніх подій, які реалізують взаємодію користувача. Табличні дані,

прив’язані до XML, бази даних тощо, можна відображати за допомогою елемента керування DataGrid View у формі рядків і клітинок.

Параметри програми — це ще одна функція Windows Forms для створення, зберігання та підтримки інформації про стан виконання у формі XML, яку можна використовувати для отримання бажаних користувачем налаштувань, таких як положення панелі інструментів і останні використані списки. Ці параметри можна повторно використовувати в майбутніх програмах.

Деякі з найкращих практик для створення програм Windows Forms включають:

Класи Windows Forms можна розширити за допомогою успадкування, щоб спроектувати структуру програми, яка може забезпечити високий рівень абстракції та багаторазового використання коду.

Форми мають бути компактними, а елементи керування обмежені розміром, який забезпечує мінімальну функціональність. Крім того, динамічне створення та видалення елементів керування може зменшити кількість статичних елементів керування.

Форми можна розбити на частини, зібрані в збірки, які можуть автоматично оновлюватися та керувати ними з мінімальними зусиллями.

Розробка програми без стану забезпечує масштабованість і гнучкість із легкістю налагодження та обслуговування.

Програми Windows Forms слід розробляти на основі необхідного рівня довіри, необхідності запитувати дозволи та обробляти винятки безпеки, де це необхідно.

Windows Form не можна передати через межі домену програми, оскільки вони не призначені для маршалінгу між доменами програми.

Windows presentation Framework (WPF) — це найновіша технологія для візуалізації інтерфейсів користувача в програмах Windows GUI з такими функціями, як підтримка 2D/3D, інтерактивна візуалізація даних і

читабельність вмісту. Він покладається на DirectX, а не на об’єкти GDI (Graphic Device Interface) для забезпечення моделі програмування, де інтерфейс користувача відокремлений від бізнес-логіки. Однак завдяки можливості взаємодії з WPF (усюди, де це потрібно), Windows Forms є хорошим вибором для додатків, яким не потрібен мультимедійний графічний інтерфейс та інші функції WPF, такі як шаблони даних/контролю, типографічні функції та функції відтворення тексту.

3.2. Програмна реалізація систем

Універсальна мова моделювання (Unified Modelling Language або UML) (рис 3.1-3.2)— це мова позначень або побудови діаграм, призначена для визначення, візуалізації і документування моделей зорієнтованих на об’єкти систем програмного забезпечення. UML не є методом розробки, іншими словами, у конструкціях цієї мови не повідомляється про те, що робити першим, а що останнім, і не надається інструкцій щодо побудови вашої системи, але ця мова допомагає вам наочно переглядати компонування системи і полегшує співпрацю з іншими її розробниками. Розробкою UML керує Object Management Group (OMG). Ця мова є загальноприйнятим стандартом графічного опису програмного забезпечення.

UML розроблено для розробки структури зорієнтованого на об’єкти програмного забезпечення, ця мова має дуже обмежену користь для програмування на основі інших парадигм.

Конструкції UML створюються з багатьох модельних елементів, які позначають різні частини системи програмного забезпечення. Елементи UML використовуються для побудови діаграм, які відповідають певній частині системи або точці зору на систему. У Umbrello UML Modeller реалізовано підтримку таких типів діаграм:

Діаграма випадків використання показує дієвих осіб (людей або інших користувачів системи), випадки використання (сценарії використання системи) та їх взаємодію

Діаграми класів, на яких буде показано класи та зв’язки між ними

Діаграми послідовності, на яких показано об’єкти і послідовність методів, якими ці об’єкти викликають інші об’єкти.

Діаграми співпраці, на яких буде показано об’єкти та їх взаємозв’язок з наголосом на об’єкти, які беруть участь у обміні повідомленнями

Діаграми стану, на яких буде показано стани, зміну станів і події у об’єкті або частині системи

Діаграми діяльності, на яких буде показано дії та зміни однієї дії іншою, які є наслідком подій, що сталися у певній частині системи

Діаграми компонентів, на яких буде показано програмні компоненти високого рівня (на зразок KParts або Java Beans).

Діаграми впровадження, на яких буде показано екземпляри компонентів та їх взаємодію.

Діаграми взаємозв’язку сутностей, на яких буде показано дані, взаємозв’язки і умови обмеження зв’язків між даними.

Складемо алгоритм роботи програми та діаграму учасників:

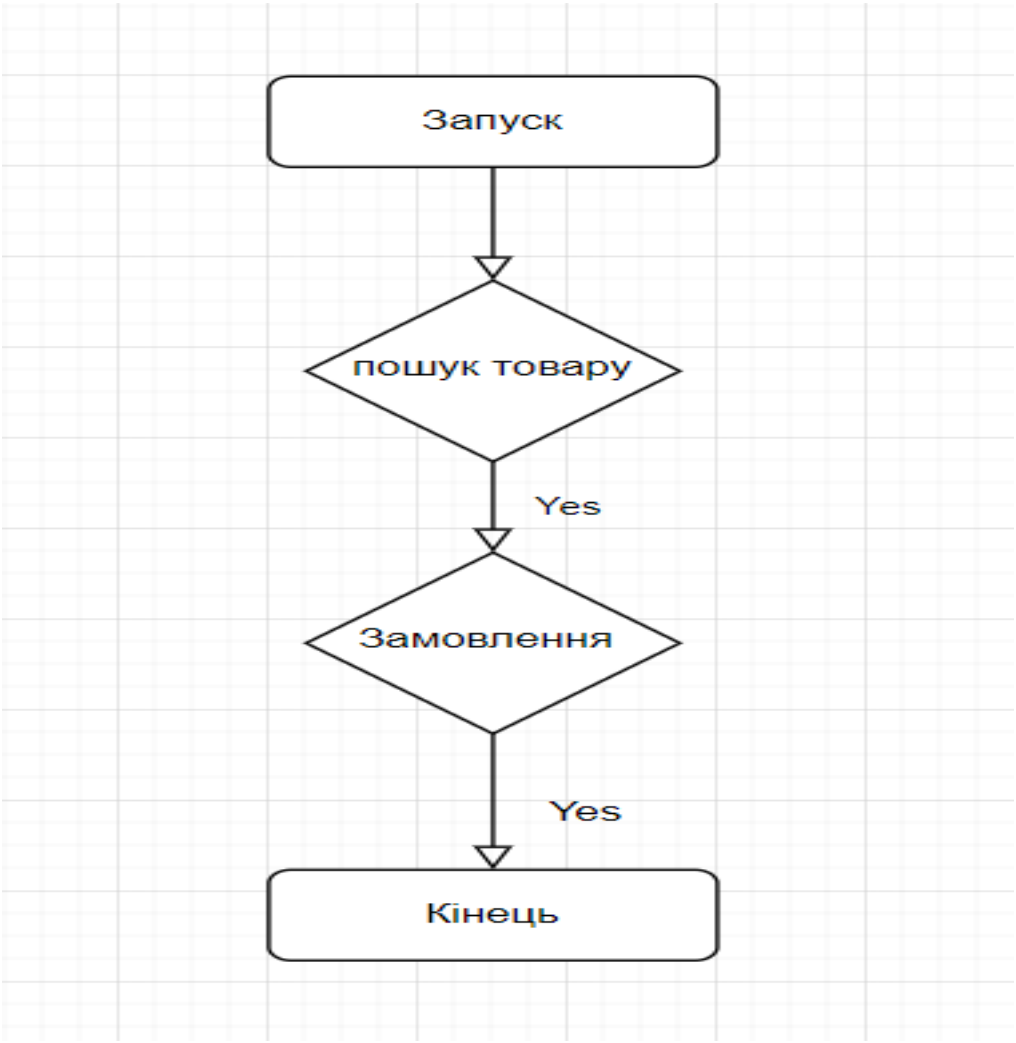


Рисунок 3.1.Алгоритм роботи програми

Рисунок 3.2-Діаграма прецедентів

3.3. Вибір та обґрунтування методів тестування додатку

Методології тестування програмного забезпечення — це різні стратегії або підходи, які використовуються для тестування програми, щоб переконатися, що вона поводиться та виглядає так, як очікувалося. Вони охоплюють усе: від переднього до внутрішнього тестування, включаючи тестування пристроїв і систем.

Тестування системи — це метод тестування «чорної скриньки», який використовується для оцінки завершеної та інтегрованої системи в цілому, щоб переконатися, що вона відповідає заданим вимогам. Функціональність програмного забезпечення перевіряється від кінця до кінця і зазвичай проводиться окремою командою тестування, а не групою розробників, перш ніж продукт буде запущено у виробництво.

Приймальне тестування є останнім етапом функціонального тестування, яке використовується для оцінки того, чи готова остаточна частина програмного забезпечення до доставки. Це передбачає забезпечення відповідності продукту всім первинним бізнес-критеріям і відповідності потребам кінцевого користувача. Це вимагає, щоб продукт пройшов випробування як всередині, так і зовні, тобто вам потрібно буде передати його кінцевим користувачам для бета-тестування разом із командою контролю якості. Бета-тестування є ключовим для отримання реальних відгуків від потенційних клієнтів і може вирішити будь-які остаточні проблеми щодо зручності використання.

3.4. Тестування додатка

Після розробки необхідно переконатися чи працює та виконує всі ті фуркції програма котру задумав розробник.

Запускаємо програму(рис 3.3).

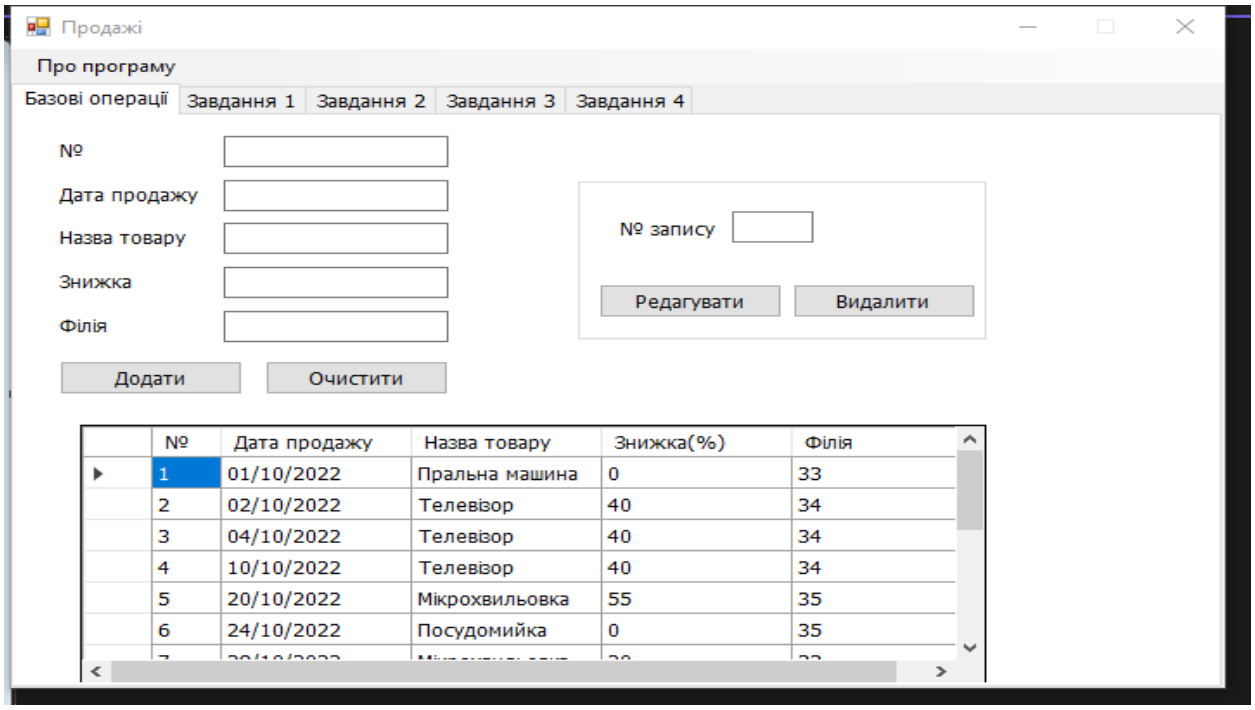


Рисунок 3.3- Головне меню

В даному вікні ми можемо переглядати записи,додавати,редагувати та видаляти інформацію.

Також переходимо до перевірки інших функції які необхідні для виконання завдання.(рис 3.4)

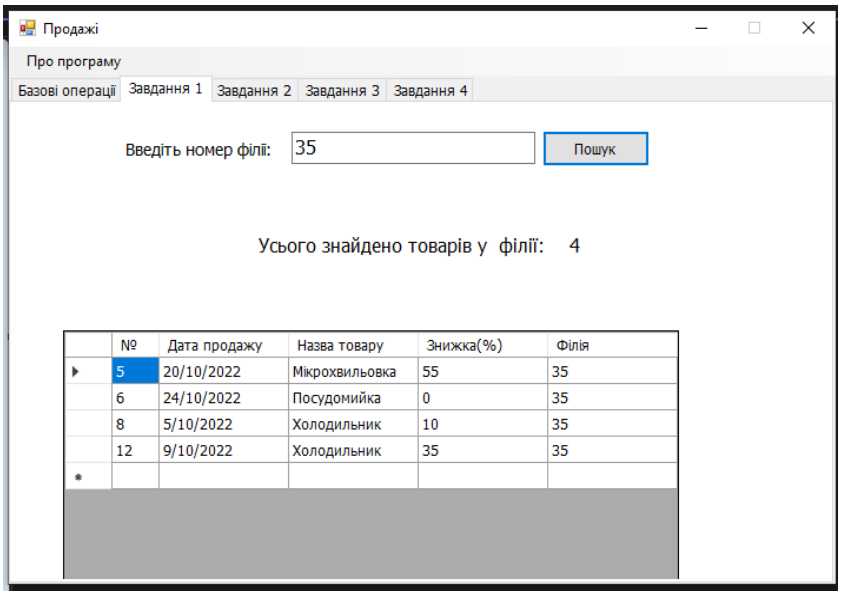


Рисунок 3.4-Пошук за критерієм філії

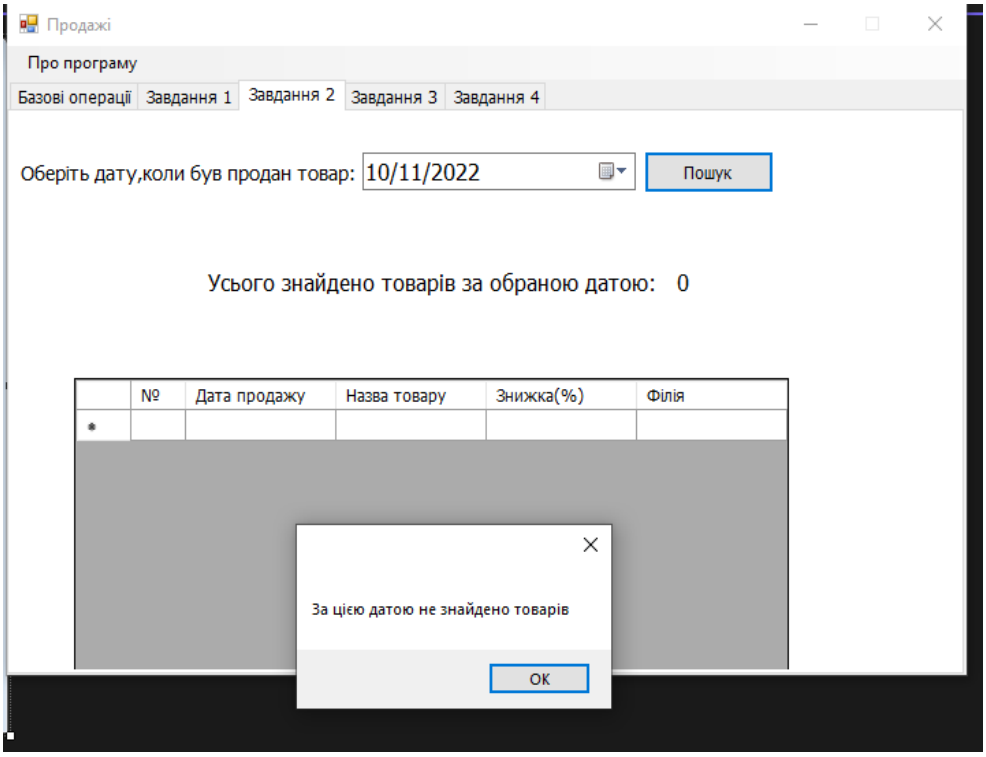


Рисунок 3.5- Помилка у випадку якщо за параметром інформація не існує

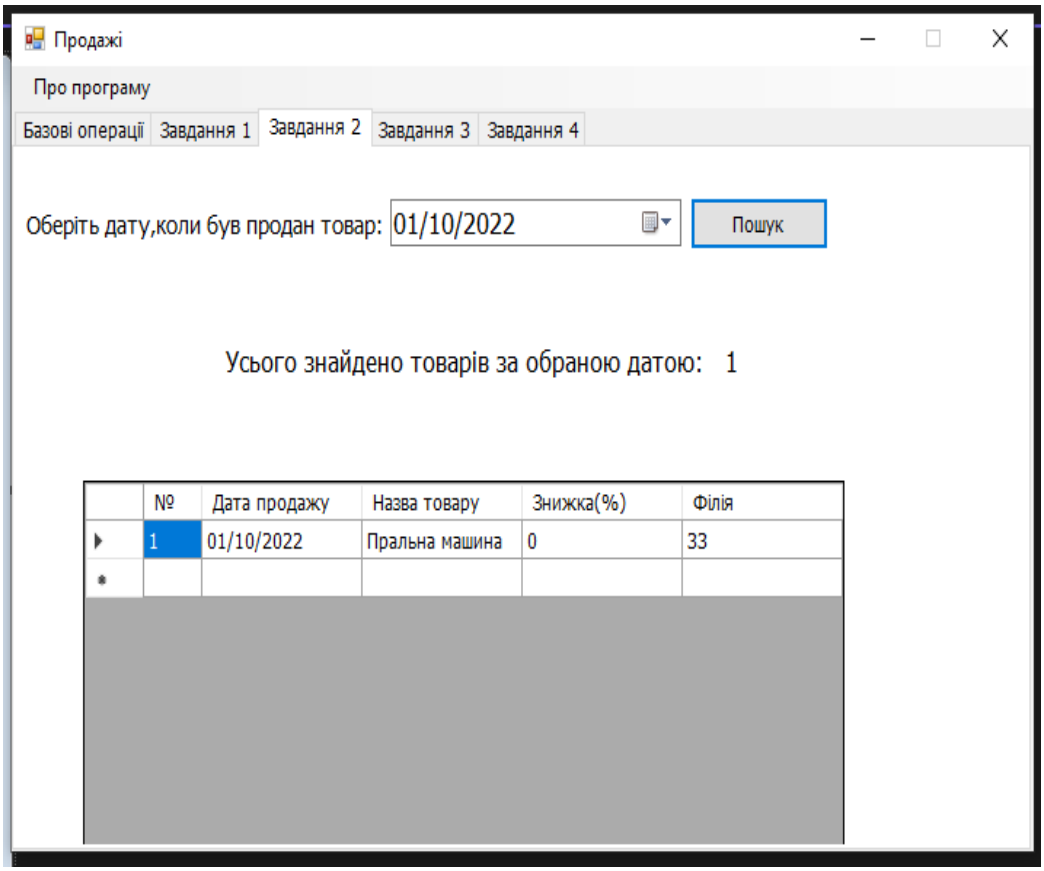


Рисунок 3.6-Пошук за датою

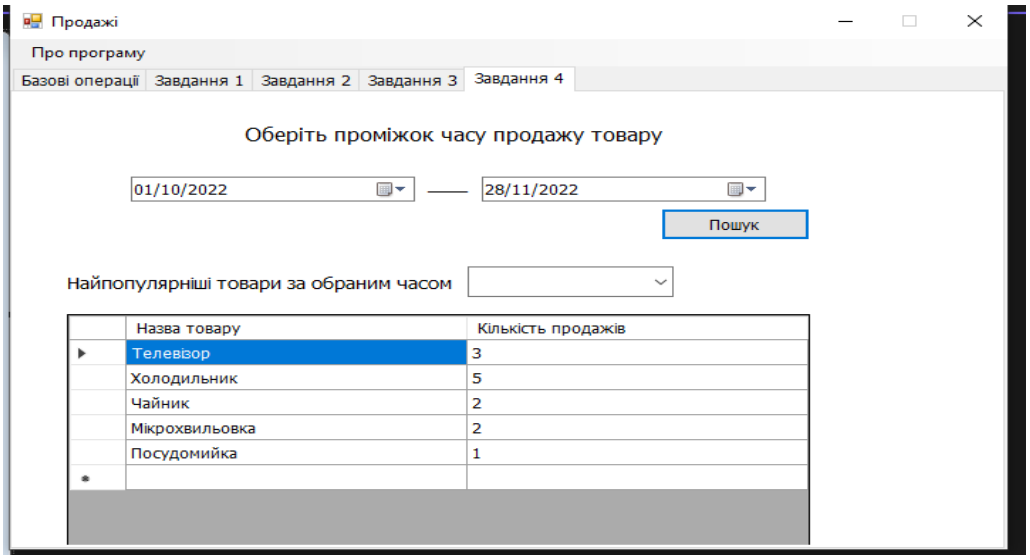


Рисунок 3.7-Пошук за проміжком періоду

Після тестування переконуємося що програма працює та виконує всі функції які необхідні розробнику

3.5 Аналіз результатів тестування

Ця архітектура визначає загальні компоненти WMCS, їхні зв'язки та загальну поведінку. WMCS можна спроектувати, визначивши параметри компонентів і поведінки. Параметри поведінки - це бізнес-правила для конкретних додатків, які створюють конкретну поведінку з абстрактної поведінки скелету. Перші експерименти успішно продемонстрували переадресацію робіт через ієрархію компонентів і поповнення локальних запасів на основі замовлень.

Експерименти також показали, що еталонна архітектура може значно скоротити зусилля з розробки WMCS. Архітектура забезпечує комунікаційний фреймворк, який зменшує ймовірність двозначності, обмежуючи сферу застосування локальними бізнес-правилами. Більше того, багато загального коду можна використовувати повторно: у нашому експерименті лише 22% коду було специфічним для конкретного додатку.

Решту коду можна повторно використовувати для інших сховищ, конфігуруючи його компоненти, їхні зв'язки та поведінку. За допомогою редактора, який ми розробляємо, частина специфічного коду і весь код для повторного використання може бути згенерований автоматично.

3.6 Керівництво користувача

Система управління складом — це програмна платформа, створена для управління та оптимізації роботи складу або розподільного центру. Він забезпечує бачення рівня запасів у режимі реального часу, допомагає

відстежувати рух товарів на складі та автоматизує різні складські процеси, такі як отримання, складування, комплектування, пакування та відправлення.

У результаті організації можуть підвищити ефективність своїх складських операцій, зменшити витрати на зберігання запасів і підвищити рівень задоволеності клієнтів, забезпечуючи точне та своєчасне виконання замовлень.

Програмне забезпечення WMS також пропонує аналітику та інформацію, щоб допомогти менеджерам складів приймати керовані даними рішення щодо управління запасами, розподілу робочої сили та оптимізації процесів.

MS допомагає відстежувати рівень і розташування запасів, щоб менеджери складів могли оптимізувати простір для зберігання та запобігти вичерпанню запасів. Менеджери можуть навіть відстежувати рух запасів на ходу, інтегрувавши його з мобільною системою управління запасами

Управління замовленнями

Управління замовленнями – ще одна важлива функція WMS. Ви можете отримувати замовлення з різних каналів, наприклад онлайн-ринків, і автоматично спрямовувати працівників на вибір потрібних продуктів і їх ефективне пакування.

Оптимізація складу

Завдяки належному програмному забезпеченню WMS компанії можуть оптимізувати планування складу та направляти працівників на найефективніші шляхи переміщення товарів з одного місця в інше.

Сканування штрих-коду та відстеження RFID

WMS може використовувати сканування штрих-кодів і технологію RFID (радіочастотна ідентифікація) для відстеження продуктів під час їхнього руху складом, забезпечуючи точний підрахунок запасів і знижуючи ймовірність помилок.

Звітність та аналітика

Створення звітів за допомогою WMS про різні складські показники, такі як рівень запасів, продуктивність працівників і рівень виконання замовлень, пропонує цінну інформацію для покращення операцій з часом.

Інтеграція з іншими системами

І останнє, але не менш важливе: ви можете інтегрувати його з іншими системами у своїй бізнес-екосистемі, щоб забезпечити цілісне уявлення про весь ланцюг поставок.

Компанії, які мають склад або розподільний центр для зберігання та управління запасами, зазвичай отримують найбільшу користь від WMS або систем управління складом. Це стосується роздрібних торговців, виробників, оптовиків та інших підприємств, яким потрібно зберігати, відстежувати та відправляти великі обсяги товарів.

Роздрібна торгівля – окрім програмного забезпечення для керування роздрібною торгівлею роздрібні продавці використовують програми для керування складами, щоб керувати запасами та замовленнями у своїх онлайн-магазинах або звичайних підприємствах. Таким чином, WMS допомагає продавцям покращити ланцюг поставок, мінімізувати ризик дефіциту запасів і оптимізувати запаси.

Електронна комерція – системи управління складом електронної комерції допомагають підприємствам ефективно відстежувати замовлення та виконувати відправлення для своїх онлайн-магазинів, щоб вони могли задовольняти швидкі вимоги клієнтів.

Оптова торгівля та розповсюдження. Іншою галуззю, яка використовує програмне забезпечення WMS, є оптові торговці та дистриб’ютори. Це допомагає їм оптимізувати завдання інвентаризації, відстежувати відправлення та оптимізувати виконання замовлень.

Виробництво. Системи управління складським господарством є популярним технічним винаходом у виробничих програмних рішеннях для

багатьох випадків використання, таких як відстеження виробничих процесів і управління ланцюгами поставок.

Логістика та транспортування – для логістичних і транспортних компаній WMS може планувати транспортування та відстежувати відправлення, коли вони просуваються через ланцюжок поставок.

Як система управління складом вписується в ланцюг поставок

Система управління складом є важливою складовою управління ланцюгом поставок. Він допомагає організаціям оптимізувати переміщення та зберігання товарів на складі чи розподільчому центрі, працюючи як важлива ланка між діяльністю на першому та нижньому рівнях.

Крім того, програмне забезпечення системи управління складом відстежує рівень запасів, стан товарів на складі та забезпечує видимість у реальному часі для всіх зацікавлених сторін у ланцюжку поставок.

Таким чином WMS гарантує, що потрібний продукт буде доступний у потрібних кількостях, у потрібний час і в потрібному місці на складі. Це допомагає підприємствам мінімізувати витрати на зберігання запасів, скоротити час циклу замовлення та покращити обслуговування клієнтів. Це також дає вам змогу приймати кращі рішення щодо управління запасами, планування складу та розподілу робочої сили.

3.7. Вимоги до технічних та програмних засобів

Оцифровка складського господарства означає відмову від паперу та впровадження таких технологій, як радіочастотні сканери, щоб зробити введення даних простішим, точнішим і швидшим. Ваша WMS повинна підтримувати зусилля з оцифрування.

Оцифровка також допоможе компаніям заощадити час і гроші завдяки меншій кількості помилок при введенні даних, скороченню паперової роботи та автоматизації процесів.

Впровадження системи управління складом, яка сприяє оцифровці, має важливе значення для компаній, які прагнуть досягти успіху на сучасному конкурентному ринку.

Використовуючи такі технології, як пристрої RF або Android, голосовий вибір, put/pack-to-light, підключення до бази даних і ERP, а також автоматичні сповіщення електронною поштою, компанії можуть автоматизувати свої бізнес-процеси та відмовитися від використання паперових відомостей матеріалів (BOM), Коносаменти (BOL), аркуші тощо.

Система управління складом повинна підтримувати такі функції прийому, як перевірка та перевірка вхідних відправлень, створення товарних квитанцій, керування утриманнями та наступні інші функції.

Отримання на складі – це більше, ніж просто замовлення запасів і відправлення їх на ваш склад. Необхідно дотримуватися належного процесу, щоб гарантувати, що ви отримуєте правильні товари та кількість, а також правильно їх зберігаєте.

Інтерфейс із радіочастотною ідентифікацією (RFID) та пристроями

PO / BOL/ Прийом і звірка виробництва

Створіть SKU вручну – додайте SKU до бази даних із сканування та попереднього повідомлення про доставку (ASN)

Отримати ASN

Можливість ручного введення даних

Розташований по зонах і місцях розташування

Програмні системи управління складом повинні надавати підприємствам необхідні інструменти та функції для повного контролю та видимості своїх запасів. Система забезпечує видимість інвентаризації, підтримуючи в режимі реального часу точний запис рівнів запасів на складі.

Це дозволяє керівникам складів та іншому персоналу в будь-який момент часу бачити, скільки кожного товару є на складі, де він знаходиться на складі та коли очікується його отримання або відправлення.

Автоматизований підрахунок циклів – підрахунки повинні генеруватися та підтримуватися автоматично системою, щоб дозволити підприємствам точно та регулярно керувати рівнем запасів на складі

Запит щодо інвентаризації місцезнаходження – можливість сканувати місцезнаходження та переглядати елементи в ньому або сканувати предмет і бачити місцезнаходження, де він має бути.

Відстеження запасів – надає дані в режимі реального часу, щоб товари можна було швидко й ефективно знайти на складі

Направлене розміщення – гарантує, що продукти зберігаються в оптимальному місці залежно від таких факторів, як розмір, температура, швидкий чи повільний рух тощо.

Відстеження життєвого циклу означає відстеження руху товарів від моменту їх отримання до моменту відправлення. Це важливо, оскільки допомагає гарантувати, що рівні запасів залишаються точними, а також допомагає забезпечити кращу видимість того, як використовуються запаси.

Маючи точні записи про транспортування кожного продукту на складі, компанії можуть легше виявляти проблеми з відправленнями та вживати коригувальні заходи, якщо це необхідно.

Номери партій – слід зберігати для кожного товару, щоб забезпечити точне відстеження

Серійні номери – забезпечують додатковий рівень деталізації під час відстеження життєвого циклу продукту на складі

Відстеження терміну придатності – гарантує відправлення лише тих товарів, термін придатності яких ще вийшов

Дати виготовлення – стежте за запасами та переконайтеся, що ви не надсилаєте продукти, які надто довго лежали на складі.

Відповідальність за належне виконання замовлень і їх своєчасну відправку лежить на керівниках складів. Стратегії комплектування допомагають покращити керування замовленнями, оптимізуючи спосіб вибору товарів зі складських приміщень і відправлення клієнтам.

Системи управління складом дозволяють підприємствам використовувати розширені стратегії комплектування, які допомагають підвищити точність замовлень і продуктивність для більш ефективної роботи.

Пакетні замовлення – партії в кошику, поповнення магазину проти електронної комерції, однорядкові та багаторядкові, великі відправники

Управління хвилями – системно встановлені розміри хвиль і автовідсікання

Комплектування з радіочастотним керуванням – усіма завданнями з комплектування керує радіочастотний пристрій

Оптимізація шляху вибору – складний шлях вибору, оптимізований у всьому розподільчому центрі.

ВИСНОВКИ

Системи управління запасами відіграють важливу роль у забезпеченні ефективного та ефективного управління запасами компанії. Ці системи надають підприємствам інструменти та функції для відстеження, контролю та оптимізації рівня запасів. Хоча вони відрізняються за функціональністю, більшість з них дозволяють користувачам обліковувати всі вхідні та вихідні запаси, що впливає на все, від обліку та виробництва до продажів і задоволеності клієнтів.

Система управління запасами — це програмне рішення, яке дозволяє підприємствам контролювати та контролювати свої запаси від закупівлі до виконання замовлення. Він діє як централізований центр, забезпечуючи видимість у реальному часі та точні дані щодо запасів щодо рівня запасів, руху та наявності. Завдяки вдосконаленому відстеженню запасів компанії можуть оптимізувати продуктивність бізнесу, зменшити витрати та краще відповідати очікуванням клієнтів.

Визнання поточної господарської діяльності на складі має найбільш значний вплив на отримання прибутку та має вирішальне значення для продуктивності компанії. Компанія повинна знати, які функції програмного забезпечення Warehouse Management System (WMS) найбільше впливають на продуктивність. Складність бізнес-факторів слід розглядати з точки зору багатьох технічних аспектів, таких як обсяг повсякденної діяльності, необхідний ступінь залучення персоналу, витрати протягом життєвого циклу запасів і координація ланцюга постачання

Керівники складів розглянуть багато стратегічних питань щодо обсягу програмного забезпечення WMS. Вони стосуватимуться найбільш прийнятного рішення, призначеного для складських операцій, незалежно від галузі чи інтеграції з розподільчим чи виробничим підприємством, і мають

посилатися на конкретні функції WMS Функції WMS тісно пов'язані з внутрішніми процесами.

В ході виконання роботи було досліджено системи які покращують та пришвидшують діяльність складу.Також розроблено аналог застосунку використовуючи мову C# та SQL

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bellifemine F, Caire G, Greenwood D (2007) Developing multi-agent systems with JADE. Wiley, Chichester.
2. Fleetwood M, Kotak DB, Wu S, Tamoto H (2003) Holonic system architecture for scalable infrastructures. In: IEEE international conference on systems, man and cybernetics 2003, vol 2, p 146.
3. Graves RJ, Wan VK, van der Velden J, van Wijngaarden B (2008) Control of complex inte- grated automated systems—system retro-fit with agent-based technologies and industrial case experiences. In: Proceedings of the 10th international material handling research colloquium.
4. Kim BI, Graves RJ, Heragu SS, St. Onge A (2002) Intelligent agent modeling of an industrial warehousing problem. IIE Trans 34.p 60.
5. Leitao P, Restivo F (2006) ADACOR: A holonic architecture for agile and adaptive manufac- turing control. Comput Ind 57,p 346.
6. Liang HL (2011) A graphical specification tool for decentralized warehouse control systems. SAI Technical Report, Eindhoven University of Technology, Eindhoven.
7. Moneva H, Caarls J, Verriet J (2009) A holonic approach to warehouse control. In: 7th interna- tional conference on practical applications of agents and multi-agent systems (PAAMS 2009), Advances in intelligent and soft computing vol 55. Springer, Berlin, p45.
8. Ten Hompel M, Schmidt T (2006) Warehouse management: Automation and organisation of warehouse and order picking systems. Springer, Berlin.
9. Van Brussel H, Wyns J, Valckenaers P, Bongaerts L, Peeters P (1998) Reference architecture for holonic manufacturing systems: PROSA. Comput Ind 3,p274

10. Verriet J, van Wijngaarden B, van Heusden E, Hamberg R (2011) Automating the development of agent-based warehouse control systems. In: Trends in practical applications of agents and multiagent systems, Advances in intelligent and soft computing, vol 90. Springer, Berlin, p 66.

11. Kusuma, Y., Hidayat, R., and Budiarto, Y. (2020). “Sistem Informasi Inventory Menggunakan Qr Code Dengan Metode Prototype,” Remik (Riset dan E-Jurnal Manaj. Inform. Komputer), vol. 5, no. 1, pp. 96–103, doi: 10.33395/remik.v5i1.10724.

12. Yuniartini, P. A., Dharma, E. M., and Dewi, I. G. I. S. (2020). “Implementasi Sistem Informasi Inventory berbasis Web pada UD. Upakara Bali”. Jurnal Teknologi Informasi Dan Komputer, 6(3). https://doi.org/10.36002/jutik.v6i3.1160.

13. Fauzi, A., Indriyani, N., Yanto, H., and Bayu, A. (2020). “Implementasi Sistem Informasi Inventory Berbasis Web (Studi Kasus: CV. Sinar Abadi Cemerlang)”. Jurnal Teknologi dan Open Source, 3(2), p. 144.

14. Rahman, U., Suhaeb, S., and Erawati. (2020). “Implementasi Sistem Informasi Inventaris Barang Berbasis Web Di Jurusan Dan Laboratorium Pendidikan Teknik Elektronika”. JETC, 5(1), p278.

15. Sujarwadi, A., Fatoni. (2020). “Sistem Informasi Inventory Barang Berbasis Web (Studi Kasus : Koperasi Kasongan Usaha Bersama Bantul)”. SAINTEKOM, 5(3), p 649.

16. Saputra, A. H. (2020). “Rancang Bangun Sistem Informasi Inventory Barang Menggunakan Metode First In First Out (FIFO) Berbasis Web Pad PT. Cipta Rasa Multindo”. SAINSHIJA, 3(2), p351.

17. East Ventures, “Daya Saing Digital Semakin Merata, Era Keemasan Semakin Dekat,” Jakarta, 2022. Accessed: Feb. 23, 2023. [Online]. Available: https://east.vc/press-release/east-ventures-digital-competitiveness-index-2022-id/ (Дата звернення 16.05.2024)

18. Simamora, V., Albab, M. U., Putra, R. J., and Diansyah. (2020). “Pengaruh Digital Marketing Terhadap Daya Saing Pelaku UMKM Kripik Singkong di Kecamatan Kebon Bawang, Jakarta Utara”. Journal of Business And Entrepreneurship, 4(2).

19. Sulaksono, J. (2020). “Peranan Digital Marketing Bagi Usaha Mikro, Kecil, Dan Menengah (Umkm) Desa Tales Kabupaten Kediri”. Generation Journal, 4(1), pp. 41-47. https://doi.org/10.29407/gj.v4i1.13906.

20. Maharani, M., Ali, A.H.N., and Astuti, H.M.(2019). “Faktor-Faktor Pengaruh Media Sosial Terhadap Keunggulan Bersaing: Studi Kasus Coffee Toffee Indonesia”. Jurnal Teknik POMITS, 1(1), P 16.

21. Huwae, H. N., Ramadhani, I. A., and Matahari. (2021). “Perancangan Sistem Informasi Penginputan Data Kapal pada PT Barakomindo Shipping Cabang Sorong Berbasis Web”. Jurnal PETISI (Pendidikan Teknologi Informasi), 2(1), p 723.

22. Prasetyo, F. S., and Informasi, S. (2019). “Rancang Bangun Sistem Informasi Pendataan Alumni Pada Stie Prabumulih Berbasis Website Dengan Menggunakan Bootstrap”. Jurnal Informatika, 17(1), p. 110. https://doi.org/10.30873/ji.v17i1.972

23. Prasetyo, B., Pattiasina, T. J., and Soetarmono, A. N. (2021). “Perancangan dan Pembuatan Sistem Informasi Gudang (Studi Kasus : PT. PLN (Persero) Area Surabaya Barat)”. Teknika. https://doi.org/10.34148/teknika.v4i1.30

24. Novendri, M. S., Saputra, A., and Firman, C. E. (2019). “Aplikasi Inventaris Barang Pada Mts Nurul Islam Dumai Menggunakan Php Dan Mysql”. Lentera Dumai, 10(2), p. 457.

25. Lokman, S., Hakim, G., and Yehia, M. (2022). "Integrating Cognitive Radio MIMO UAVs in Cellular Networks for 5G and Beyond", International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), p 16.

26. Panduardi, F., and Haq, E. S. (2016). “Wireless Smart Home System Menggunakan Raspberry Pi”. Jurnal Teknologi Informasi Dan Terapan, 3(1), p.325

27. Efendi, Y. (2018). “Internet Of Things (IoT) Sistem Pengendalian Lampu Menggunakan Raspberry Pi Berbasis Mobile”. Jurnal Ilmiah Ilmu Terapan, 4(10), p 926.

28. B. S. Fauziah, G. Abdillah, and F. Renaldi, “Perancangan Dan Implementasi Warehouse Management Sistem Pada Pt. Feedmill Indonesia,” Pros. SNST, vol. 4, pp. 146–151, 2017, [Online]. Available: https://publikasiilmiah.unwahas.ac.id/index.php/PROSIDING_SNST_FT/article/view/1893

29. Kurniawati., Badrul, M. (2021). “Penerapan Metode Waterfall Untuk Perancangan Sistem Informasi Inventory Pada Toko Keramik Bintang Terang”. Jurnal PROSISKO, 8(2), p 752.

30. Y. Prasetyo, P. S. Ivan, and Q. H. R. Al Hazmi, “Sistem Pemantauan Suhu dan Kelembaban Ruangan Secara Real-Time Berbasis Web Server,” J. Technol. Informatics, vol. 1, no. 1, pp. 56–60, 2020, doi: 10.37802/joti.v1i1.12.

31. Chaouchi, H. (2013).”The Internet of Things: Connecting Objects. Wiley”. Retrieved fromhttps://books.google.co.id/books?id=EGNm4iT8TC8C

32. Saptadi, A. H. (2019). “Perbandingan Akurasi Pengukuran Suhu dan Kelembaban antara SensorDHT11 dan DHT22 Studi Komparatif pada Platform ATMEL AVR dan Arduino”. Jurnal INFOTEL ST3 Telkom Purwokerto, 6(2).

33. Sparkfun. (n.d.). “Digital-output relative humidity & temperature sensor/module”. Retrieved from https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf (Дата звернення 16.05.2024)

34. Prasetyo., Ivan, P. S., and Hazmi, Q. A. (2019). “Sistem Pemantauan Suhu dan Kelembaban Ruangan Secara Real-Time Berbasis Web Server”. Journal of Technology and Informatics (JoTI), 1(1).

35. Ivory, “Penggunaan Sensor Suhu Bayi Pada Inkubator,” J. Tek. elektro, vol. 10, p. 194, 202

ДОДАТОК А.
(обов’язковий)

ЛІСТИНГ ПРОГРАМНОГО КОДУ

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

namespace Contacts
{
   public partial class Form1 : Form
   {
      Sells[] sells;
      Dictionary<string, int> popularSales;

      public Form1()
      {
         InitializeComponent();
         FillSells();
         ReloadDataGridView();
```

```
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Add(textBox1.Text, textBox2.Text, textBox3.Text,
int.Parse(textBox4.Text), textBox6.Text);
            ReloadDataGridView();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            Edit(textBox5.Text, textBox1.Text, textBox2.Text, textBox3.Text,
int.Parse(textBox4.Text), textBox6.Text);
            ReloadDataGridView();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            ClearTextBoxes();
        }

        private void FillSells()
        {
            sells = new Sells[14]
            {
                new Sells(1,"Пральна машина","01/10/2022","33",0),
```

```
        new Sells(2,"Телевізор","02/10/2022","34",40),
        new Sells(3,"Телевізор","04/10/2022","34",40),
         new Sells(4,"Телевізор","10/10/2022","34",40),
        new Sells(5,"Мікрохвильовка","20/10/2022","35",55),
        new Sells(6,"Посудомийка","24/10/2022","35",0),
        new Sells(7,"Мікрохвильовка","29/10/2022","33",20),
        new Sells(8,"Холодильник","5/10/2022","35",10),
        new Sells(9,"Холодильник","6/10/2022","33",10),
        new Sells(10,"Холодильник","7/10/2022","34",90),
        new Sells(11,"Холодильник","8/10/2022","34",5),
        new Sells(12,"Холодильник","9/10/2022","35",35),
        new Sells(13,"Чайник","10/10/2022","33",5),
          new Sells(14,"Чайник","11/10/2022","34",50),
    };

}
private void LoadGridByAffiliate()
{
    dataGridView2.Rows.Clear();
    var subset = from sel in sells where sel.Affiliate == textBox7.Text select sel;
    foreach (var item in subset)
    {
       dataGridView2.Rows.Add(item.Key,
          item.DateOfSell, item.NameOfTechnic, item.Discount, item.Affiliate);
    }
    label7.Text = (dataGridView2.Rows.Count - 1).ToString();
    if (label7.Text == "0")
    {
```

```
            MessageBox.Show("У цій філії немає товарів");
            return;
        }
    }
    private void LoadGridByDateSell()
    {
        dateTimePicker1.CustomFormat = "dd/MM/yyyy";
        dataGridView3.Rows.Clear();
        var subset = from sel in sells where sel.DateOfSell == dateTimePicker1.Text
select sel;

        foreach (var item in subset)
        {
            dataGridView3.Rows.Add(item.Key,
              item.DateOfSell, item.NameOfTechnic, item.Discount, item.Affiliate);
        }

        label12.Text = (dataGridView3.Rows.Count - 1).ToString();
        if (label12.Text == "0")
        {
            MessageBox.Show("За цією датою не знайдено товарів");
            return;
        }
    }
    private void LoadGridBetweenDateSell()
    {
        dateTimePicker2.CustomFormat = "dd/MM/yyyy";
        dateTimePicker3.CustomFormat = "dd/MM/yyyy";
```

```
        for (int i = 0; i < (dateTimePicker3.Value.Subtract(dateTimePicker2.Value)).Days; i++)
        {
            for (int j = 0; j < sells.Length; j++)
            {

                if (DateTime.Parse(sells[j].DateOfSell) == DateTime.ParseExact(dateTimePicker2.Text, "dd/MM/yyyy", CultureInfo.InvariantCulture).AddDays(i) && sells[j].Discount > 0)
                {
                    dataGridView4.Rows.Add(sells[j].Key,
                sells[j].DateOfSell, sells[j].NameOfTechnic, sells[j].Discount, sells[j].Affiliate);
                }
            }
        }
        label16.Text = (dataGridView4.Rows.Count - 1).ToString();
        if (label16.Text == "0")
        {
            MessageBox.Show("За цією датою не знайдено товарів");
            return;
        }
    }
    private void LoadGridMostPopularSells()
    {
        dataGridView5.Columns.Clear();
        dataGridView5.Columns.Add("1", "Назва товару");
        dataGridView5.Columns[0].Width = 240;
        dataGridView5.Columns.Add("2", "Кількість продажів");
        dataGridView5.Columns[1].Width = 240;
```

```
        dataGridView5.Rows.Clear();
        comboBox1.Items.Clear();
        dateTimePicker4.CustomFormat = "dd/MM/yyyy";
        dateTimePicker5.CustomFormat = "dd/MM/yyyy";
        popularSales = new Dictionary<string, int>();

        for (int i = 0; i < (dateTimePicker5.Value.Subtract(dateTimePicker4.Value)).Days; i++)
        {
            for (int j = 0; j < sells.Length; j++)
            {
                if (DateTime.Parse(sells[j].DateOfSell) == DateTime.ParseExact(dateTimePicker4.Text, "dd/MM/yyyy", CultureInfo.InvariantCulture).AddDays(i))
                {
                    if (j > 0)
                    {
                        if (popularSales.ContainsKey(sells[j].NameOfTechnic))
                        {
                            popularSales[sells[j].NameOfTechnic] += 1;
                        }
                        else
                        {
                            popularSales.Add(sells[j].NameOfTechnic, 1);
                        }
                    }
                }
            }
        }
```

```
if (popularSales.Count == 0)
{
    MessageBox.Show("За цією датою не знайдено товарів");
    return;
}
```

ДОДАТОК Б.
(обов’язковий)

ЛІСТИНГ БД

```
-- Створення бази даних
CREATE DATABASE WarehouseDB;
GO
-- Використання новоствореної бази даних
USE WarehouseDB;
GO
-- Створення таблиці Suppliers
CREATE TABLE Suppliers (
    SupplierID INT PRIMARY KEY IDENTITY(1,1),
    SupplierName NVARCHAR(100) NOT NULL,
    ContactName NVARCHAR(100),
    Address NVARCHAR(255),
    City NVARCHAR(100),
    PostalCode NVARCHAR(20),
    Country NVARCHAR(50),
    Phone NVARCHAR(50)
);
GO

-- Створення таблиці Customers
CREATE TABLE Customers (
    CustomerID INT PRIMARY KEY IDENTITY(1,1),
    CustomerName NVARCHAR(100) NOT NULL,
    ContactName NVARCHAR(100),
    Address NVARCHAR(255),
    City NVARCHAR(100),
```

```
    PostalCode NVARCHAR(20),
    Country NVARCHAR(50),
    Phone NVARCHAR(50)
);
GO

-- Створення таблиці Products
CREATE TABLE Products (
    ProductID INT PRIMARY KEY IDENTITY(1,1),
    ProductName NVARCHAR(100) NOT NULL,
    SupplierID INT,
    Category NVARCHAR(50),
    QuantityPerUnit NVARCHAR(50),
    UnitPrice DECIMAL(10, 2),
    UnitsInStock INT,
    UnitsOnOrder INT,
    ReorderLevel INT,
    Discontinued BIT,
    FOREIGN KEY (SupplierID) REFERENCES Suppliers(SupplierID)
);
GO

-- Створення таблиці Inventory
CREATE TABLE Inventory (
    InventoryID INT PRIMARY KEY IDENTITY(1,1),
    ProductID INT,
    Quantity INT,
    Location NVARCHAR(100),
    LastUpdated DATETIME,
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
```

```
);
GO

-- Створення таблиці Orders
CREATE TABLE Orders (
    OrderID INT PRIMARY KEY IDENTITY(1,1),
    CustomerID INT,
    OrderDate DATETIME NOT NULL,
    ShipDate DATETIME,
    ShipAddress NVARCHAR(255),
    ShipCity NVARCHAR(100),
    ShipPostalCode NVARCHAR(20),
    ShipCountry NVARCHAR(50),
    FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);
GO

-- Створення таблиці OrderDetails
CREATE TABLE OrderDetails (
    OrderDetailID INT PRIMARY KEY IDENTITY(1,1),
    OrderID INT,
    ProductID INT,
    UnitPrice DECIMAL(10, 2),
    Quantity INT,
    Discount DECIMAL(5, 2),
    FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
    FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
GO
```

ДОДАТОК В
(обов’язковий)

ПРЕЗЕНТАЦІЙНИЙ МАТЕРІАЛ

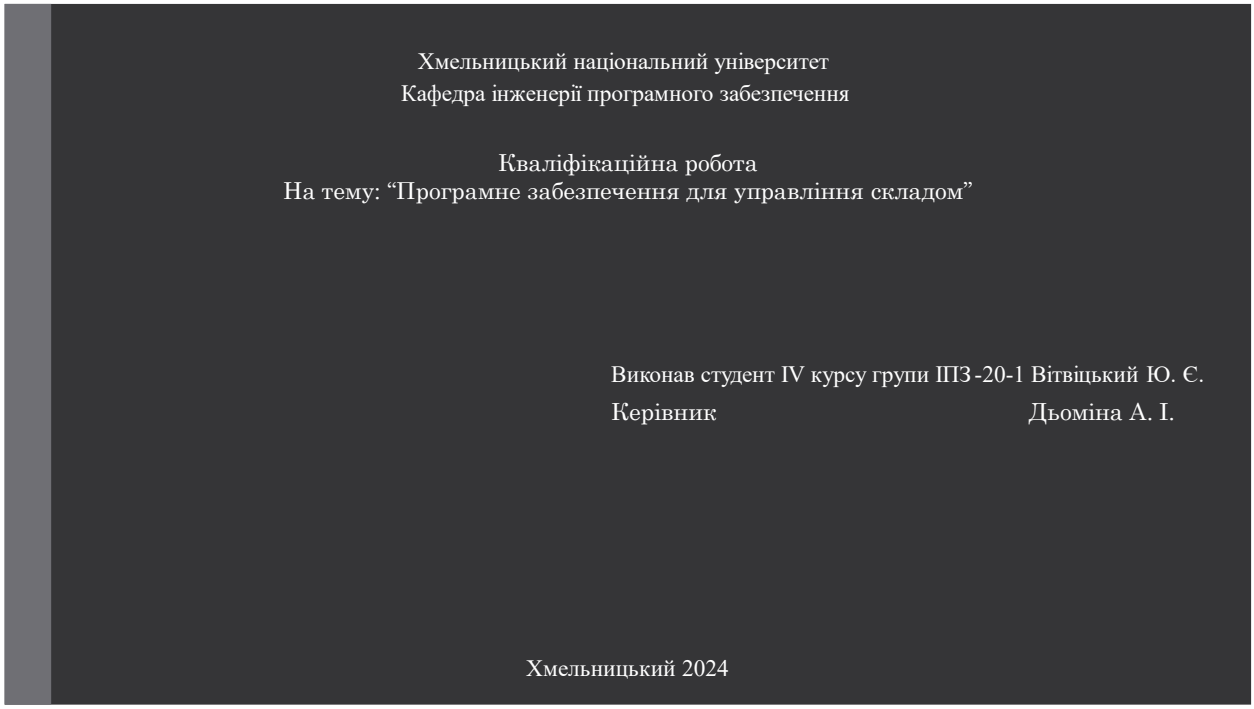


Рисунок Г.1 — Слайд 1

Рисунок Г.2 — Слайд 2

Наявні рішення

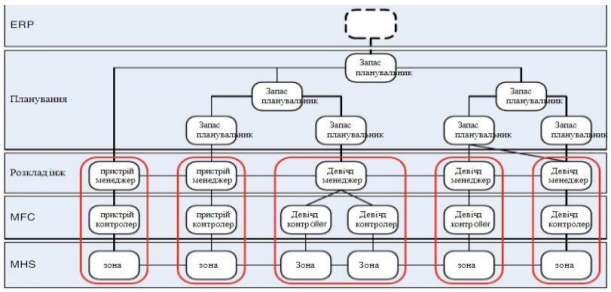
У ході підготовки до розробки програмного забезпечення був проведений аналіз ринку на наявні продукти. Були розглянуті такі наявні продукти як inFlow inventory та Wasp InventoryCloud

Перевагами inFlow inventory є інтуїтивний інтерфейс, віддалене керування за допомогою онлайн порталу, аналіз наявних продуктів на складі і його стеження.

Переваги Wasp InventoryCloud є мобільність, введення хмарних технологій, аналіз витрат та стеження за доставками

Рисунок Г.3 — Слайд 3

Архітектура програмного забезпечення



Була розроблена архітектура даного програмного забезпечення, яка стандартизує не лише компоненти та їхні інтерфейси, але й функціональність компонентів. Дана архітектура розрізняє чотири типи компонентів, кожен з яких відповідає своєму рівню ієрархії: планувальники запасів, менеджери пристроїв, контролери пристроїв і зони обробки матеріалів.

Рисунок Г.4 — Слайд 4

Алгоритм роботи

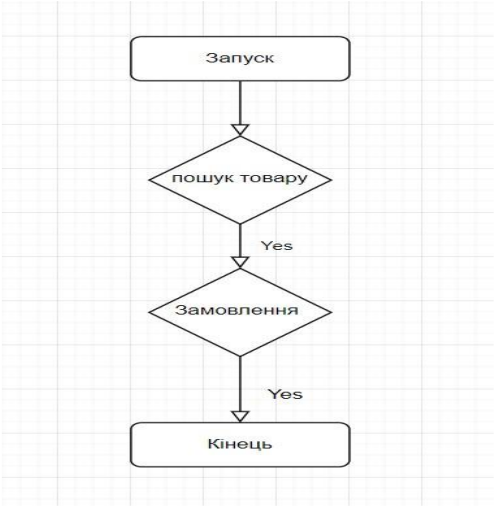


Рисунок Г.5 — Слайд 5

Діаграма прецедентів

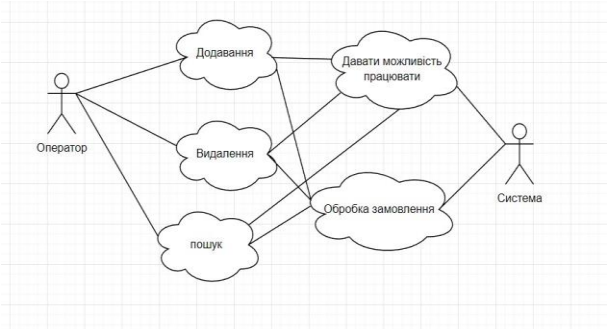


Рисунок Г.6 — Слайд 6

Діаграма баз даних

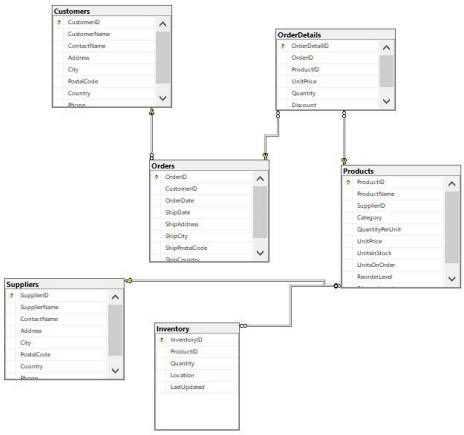


Рисунок Г.7 — Слайд 7

Вибір технологій і методів реалізації

Для розробки програмного забезпечення було вибрано середовище Microsoft Visual Studio , а мовою програмування – C# та SQL.

Переваги C# є широкий спектр бібліотек, інструментів та фреймворків, середня складність мови і вона активно розвивається

Переваги SQL є можливість швидко отримувати необхідну інформацію, оцінка якості даних, можливість розраховувати метрики, візуалізувати дані та приймати зважені рішення на основі отриманої інформації.

Перевагами Microsoft Visual Studio є безліч інструментів, таких як редактор коду, відладчик, дизайнер інтерфейсу користувача, практичний інтерфейс та інтегація з мовами на базі C

Рисунок Г.8 — Слайд 8

Інтерфейс головного меню

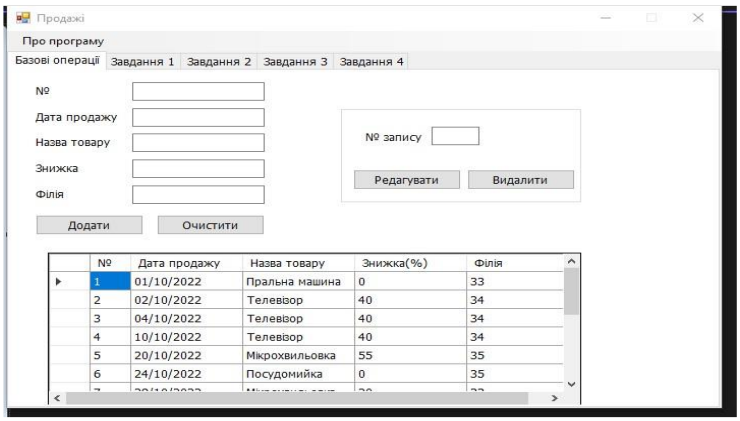


Рисунок Г.9 — Слайд 9

Пошук за критерією філії

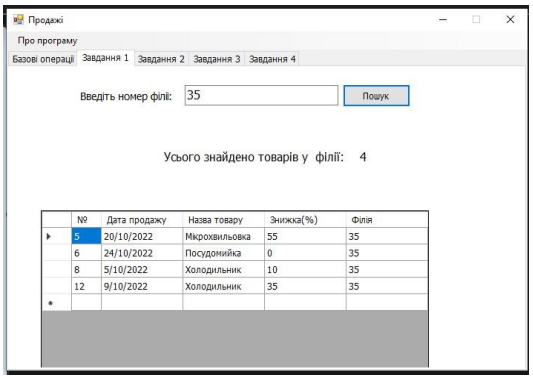


Рисунок Г.10 — Слайд 10

Пошук за датою

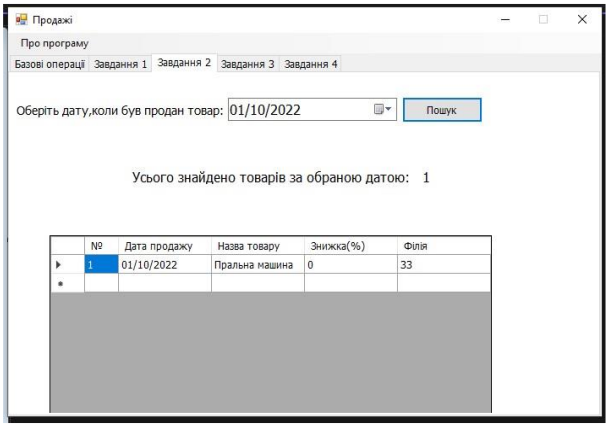


Рисунок Г.11 — Слайд 11

Висновок

При виконання кваліфікаційної роботи на тему «Програмне забезпечення для управління складом» була розроблена архітектура програмного забезпечення та її реалізація

Середовище розробки Microsoft Visual Studio. Мови C# та SQL

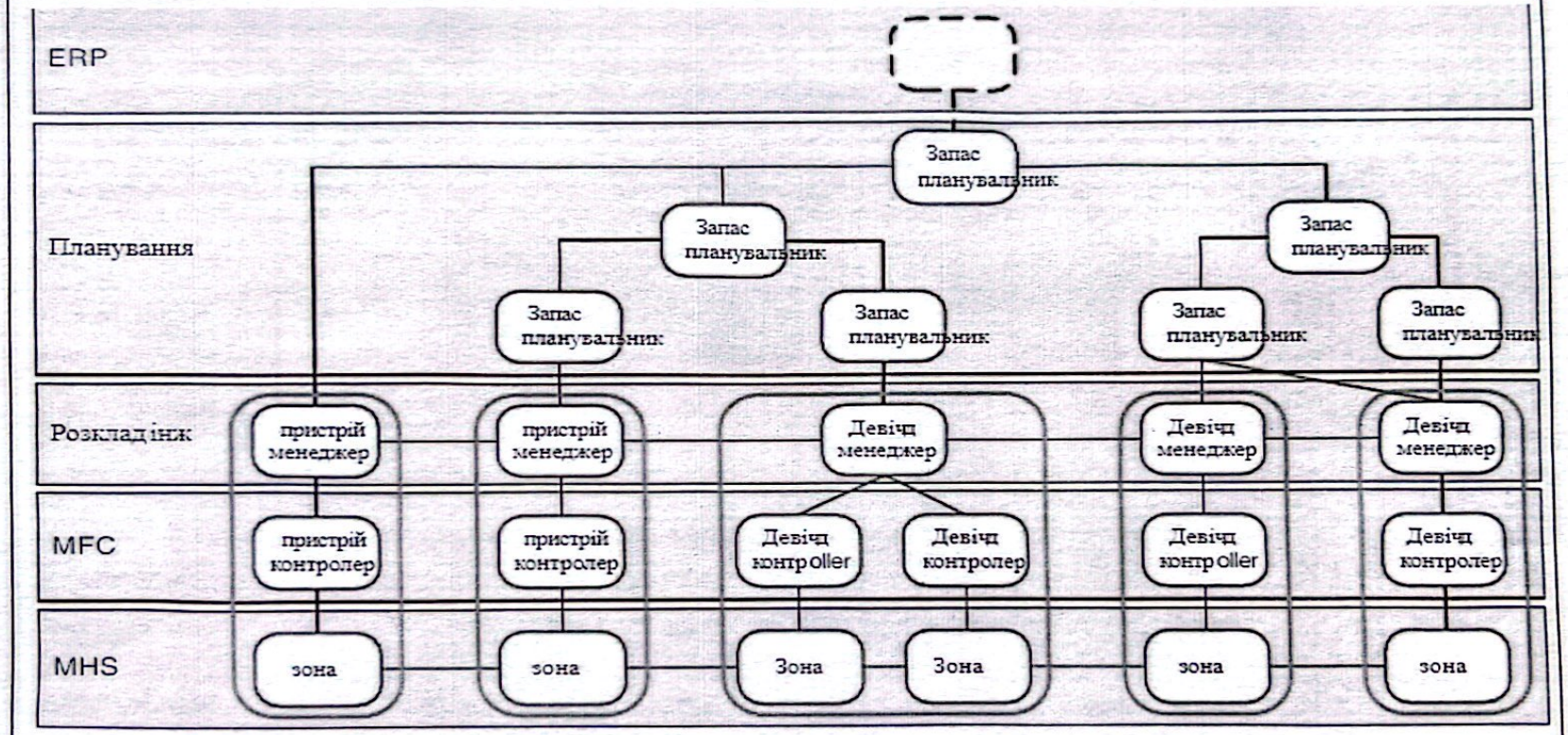
Дане програмне забезпечення надаватиме можливість управляти продукцією на складі і ввести звіт.

Рисунок Г.12 — Слайд 12

Дякую за увагу

Рисунок Г.13 — Слайд 13

ГРАФІЧНІ МАТЕРІАЛИ



Зм.	Арк.	№ докум.	Підпис	Дата
Розробив		Вітвіцький Ю.Є.		
Керівник		Дьоміна А.І.		
Консульт.				
Н. Контр.		Яшина О.М.		
Зав. каф.		Бедратюк Л.П.		

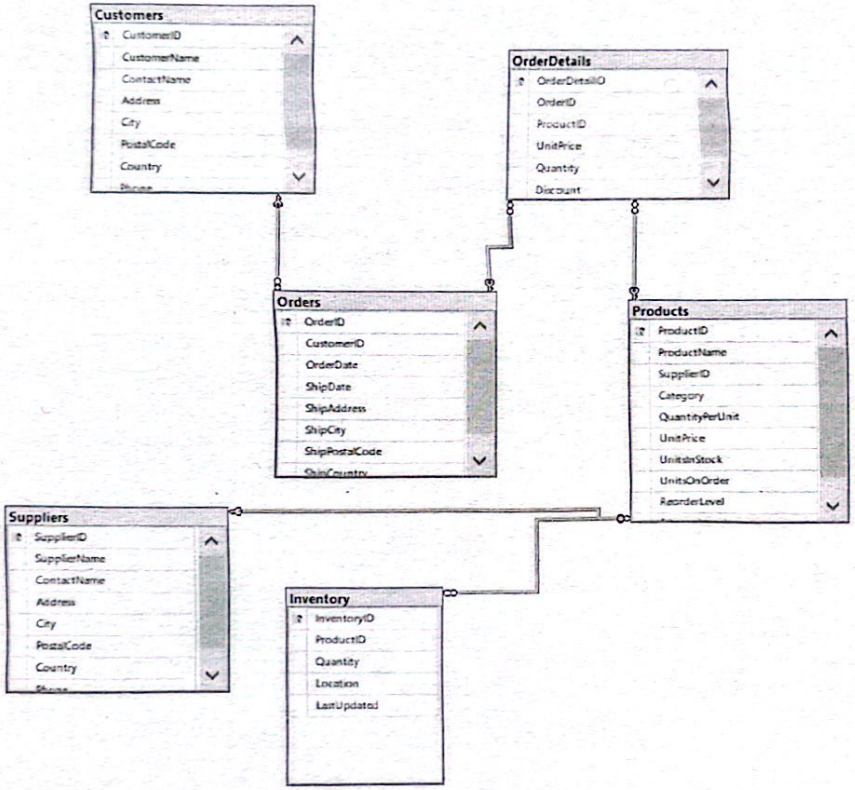
КвРІПЗ.200176.01.25.ПЗ

Програмне забезпечення для управління складом

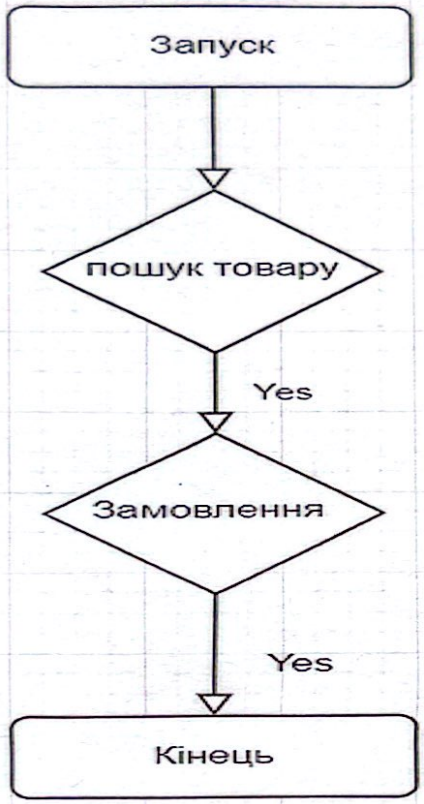
Відомість документів

Літера	Маса	Масштаб
Аркуш 1	Аркушів 4	

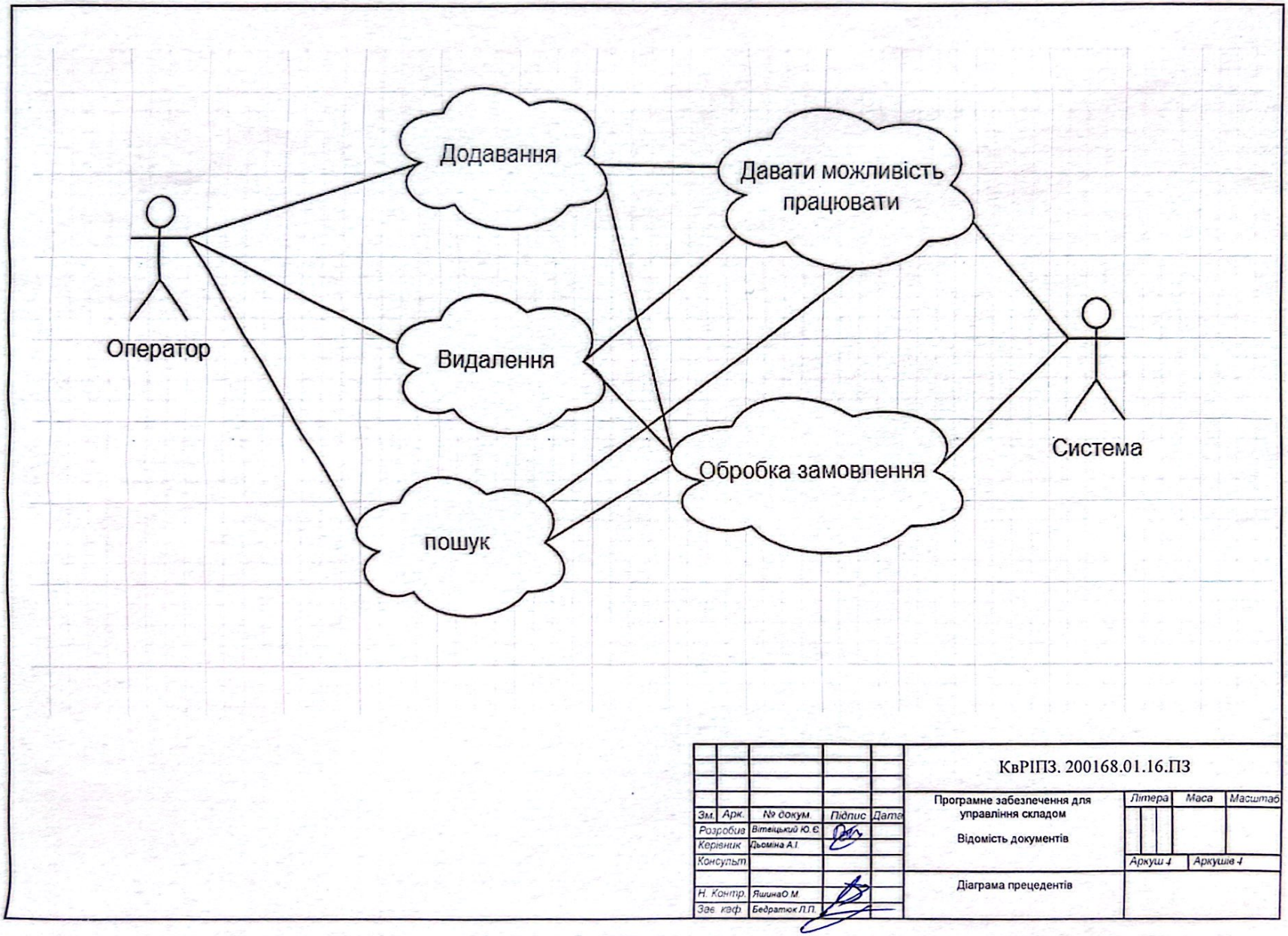
Еталонна архітектура



					КвРІПЗ. 200168.01.16.ПЗ		
Зм.	Арк.	№ докум.	Підпис	Дата	Програмне забезпечення для управління складом	Літера	Маса	Масштаб
Розробив		Вітецький Ю.С.						
Керівник		Дьоміна А.І.			Відомість документів			
Консульт.						Аркуш 2	Аркушів 4	
Н. Контр.		Яшина О.М.			Логічна схема даних бази для складу			
Зав. каф.		Бедратюк Л.П.						



					КвРІПЗ. 200168.01.16.ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата	Програмне забезпечення для управління складом	Літера	Маса	Масштаб
Розробив		Вітецький Ю.С.			Відомість документів			
Керівник		Дьоміна А.І.				Аркуш 1	Аркушів 1	
Консульт.								
Н. Контр.		ЯшинаО.М.			Алгоритм роботи програми			
Зав. каф.		Бедратюк Л.П.						



					КвРІПЗ. 200168.01.16.ПЗ		
Зм.	Арк.	№ докум.	Підпис	Дата	Програмне забезпечення для управління складом	Літера	Маса / Масштаб
Розробив		Вітецький Ю.Є.			Відомість документів		
Керівник		Дьоміна А.І.				Аркуш 1	Аркушів 1
Консульт.							
Н. Контр.		ЯшинаО.М.			Діаграма прецедентів		
Зав. каф.		Бедратюк Л.П.					

СУПРОВІДНІ ДОКУМЕНТИ

Завідувачу кафедри
інженерії програмного забезпечення
проф. Бедратюку Л. П.
студента групи ІПЗ-20-1
Вітвіцький Ю.Є.
Прізвище, ініціали

ЗАЯВА

Прошу закріпити за мною тему кваліфікаційної роботи освітнього ступеня «бакалавр» за спеціальністю 121 «Інженерія програмного забезпечення»: Програмне забезпечення для управління складом

(керівник роботи – Дьоміна Анастасія Іванівна)
Прізвище, ім'я, по батькові

18.06.2024
Дата

Підпис студента

Завідувачу кафедри інженерії програмного забезпечення проф. Бедратюку Л. П.

здобувача вищої освіти
Вітвіцький Ю. Є.
Прізвище, ініціали

факультет ІТ, 4 курс, група ІПЗ-20-1

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності в Хмельницькому національному університеті», згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщений та надаю свою згоду на обробку й збереження університетом моєї роботи в інституційному репозитарії Хмельницького національного університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та/або Anti-Plagiarism) і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення текстових збігів у роботах.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

18.06.2024
дата

підпис

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
освітнього ступеня «Бакалавр»

Дипломник Вітвіцький Юрій Євгенович

Тема Програмне забезпечення для управління складом

Спеціальність 121 – Інженерія програмного забезпечення

Обсяг кваліфікаційної роботи:

Кількість листів креслень 4; кількість сторінок записки 81

1.Короткий зміст пояснювальної записки та прийнятих рішень У кваліфікаційній роботі досліджено і проаналізовано предметну область розроблення програмного забезпечення для управління складом. Був проведений аналіз існуючих програм на ринку, розглянуто їх переваги і недоліки, та доведено актуальність розробки нового програмного забезпечення. Розглянуто інструменти для реалізації спроектованих рішень, в результаті чого створено програмне забезпечення. Також було проведено тестування програми, за результатами якого доведено, що розроблене програмне забезпечення працює коректно та готове до експуатації.

2. Висновок про відповідність роботи поставленому завданню Кваліфікаційна робота виконана відповідно до поставленого завдання та з дотриманням всіх вимог.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки та передових методів роботи У вступі доведено актуальність теми, визначено мету та завдання дипломного проектування. У першому розділі проведено аналіз предметної області, розглянуто існуючі рішення та визначені функціональні і нефункціональні вимоги до розроблюваного програмного забезпечення. У другому розділі проведено аналіз сучасних архітектур, розглянуто їх переваги і недоліки та визначено, що система буде відповідати монолітній архітектурі та моделі клієнт-сервер. У третьому розділі підготовлено всі залежності для написання коду та виконано практичну розробку програмних модулів і описано їх особливості, в результаті чого створено програмний продукт. Також у цьому розділі виконано модульне тестування системи та проведено його у відповідності до функціональних вимог, в результаті було підтверджено коректну роботу програми.

4. Позитивні сторони роботи Тематика кваліфікаційної роботи є актуальною, оскільки логістичний ринок і досі на стадії розвитку, а з розвитком цифрових технологій цей процес пришвидшився. Також було застосовано новітні технології для побудови програмного продукту та актуальні архітектурні рішеня.

5. Негативні сторони роботи У роботі були реалізовані лише базові можливості інструменти для управління складом.

6. Оцінка графічного оформлення та пояснювальної записки Графічне оформлення виконано відповідно до теми кваліфікаційної роботи та подано у вигляді діаграм і рисунків. Пояснювальна записка оформлена згідно вимог чинних стандартів.

7. Відгук про кваліфікаційну роботу в цілому Кваліфікаційна робота виконана на необхідному рівні і заслуговує позитивної оцінки. Матеріал пояснювальної записки структурований, послідовний, чіткий та простий, що дозволяє чітко зрозуміти викладений матеріал у рамках тематики проектування. Графічний матеріал дає можливість наочно побачити деталі проектування системи.

8. Інші зауваження

9. Оцінка кваліфікаційної роботи Кваліфікаційна робота виконана у повному обсязі, відповідає поставленій задачі та заслуговує на оцінку «добре».

РЕЦЕНЗЕНТ Професор каф. ІСК Едуард МАЦЮК

«18» 06 2024 р. (підпис)

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ
КАФЕДРИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатами звіту/звітів перевірки роботи, продукованими програмно-технічним засобом (ами), на наявність текстових збігів:

Назва кваліфікаційної роботи: «Програмне забезпечення для управління складом»
Автор: Вівтвіцький ЮрійЄвгенович
Освітня програма: Освітньо-професійна програма «Інженерія програмного забезпечення»
Спеціальність: 121 – Інженерія програмного забезпечення
Науковий керівник: Дьоміна Анастасія Іванівна, асистент
Після аналізу звіту/звітів зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	**відповідає**
2	Виявлені запозичення не є академічним плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої й електронної версії роботи	
3	Виявлені запозичення не є академічним плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнуті. Робота може бути допущена до захисту після того, як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття текстових запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Запозичення, виявлені у роботі, є законними і не є плагіатом, оскільки:

1) у тексті кваліфікаційної роботи системою перевірки на плагіат Unicheck виявлено схожість з деякими документами у частині загальновживаних обов'язкових словосполучень у стандартних бланках (титулка, відомість документів), у структурі змісту, назвах розділів/підрозділів, у рамках основних написів, у назвах публікацій переліку джерел посилання;

2) в якості запозичень системою Unicheck було зафіксовано деякі послідовності вихідного коду і посилання на бібліотеки, які є стандартними мовними конструкціями програмування та не можуть розглядатися як об'єкт авторських прав і, відповідно, їх порушення;

3) запозичення, виявлені в тексті роботи, є фрагментарними.

Максимальний обсяг запозичень, визначений системою Anti-Plagiarism, складає 2.0%. Обсяг запозичень, визначений системою Unicheck виявлення збігів ідентичності/схожості, складає 11.6% і адресується до 306 джерел з Інтернету і 116 джерела з бібліотеки, що, з урахуванням наведених обґрунтувань, відповідає характеру теми і свідчить на користь кваліфікаційної роботи.

Дата 10.06.2024 р.

Завідувач кафедри _____ Леонід БЕДРАТЮК

Гарант освітньої програми _____ Леонід БЕДРАТЮК

Керівник кваліфікаційної роботи _____ Анастасія ДЬОМІНА

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 2.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 9%**

	Документ		Сумарний збіг по Базі Даних	
ID: 130388 Назва: Програмне забезпечення для управління складом Додано в БД: 2024-06-14 Автора: Вітвіцький Ю.Є. Керівники: Дьоміна А.І. Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	93758	1408	4352 (5%)	56 (4%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK
by Turnitin

Ім'я користувача:
ІПЗ

ID перевірки:
1016359160

Дата перевірки:
14.06.2024 08:28:43 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.06.2024 08:48:25 EEST

ID користувача:
100012953

Назва документа: **БКР_Програмне забезпечення для управління складом_Вітвіцький_Дьоміна**

Кількість сторінок: 99 **Кількість слів:** 16290 **Кількість символів:** 129810 **Розмір файлу:** 1.40 MB **ID файлу:** 1016163801

11.6%
Схожість

Найбільша схожість: **2.16% з джерелом з Бібліотеки** (ID файлу: **1016121109**)

9.55% Джерела з Інтернету	306	Сторінка 101
4.21% Джерела з Бібліотеки	116	Сторінка 104

0% Цитат

Цитати	8	Сторінка 105
Посилання	1	Сторінка 105

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	25