ЗМІСТ

С.

ВСТУП………………………………………………………………………..……4

1 ТЕХНІЧНІ ХАРАКТЕРИСТИКИ………………………………………….…..5

1.1 Постановка задачі……………………………………………………….….…5

1.1.1 Характеристика об’єкта автоматизації…………………………….……....5

1.1.2 Вимоги до програми………………………………………………………...5

1.2 Опис математичного методу рішення задачі………………………………..6

1.3 Визначення структури вхідних даних ………………………………………9

1.4 Визначення структури вихідних даних…………………………………….10

2 КОМП’ЮТЕРНА СИСТЕМА………………………………………………...12

2.1 Технічні характеристики комп’ютера та зовнішніх пристроїв ……….….12

2.2 Вибір програмних засобів та операційної системи……………………......12

3 ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОЕКТУВАННЯ…………………….……..14

3.1 Інтерфейс програми…………………………………………………………14

3.2 Алгоритм рішення задачі…………………………………………………...22

4 ПРОГРАМУВАННЯ ТА ТЕСТУВАННЯ …………………………………...25

4.1 Розробка програми…………………………………………………………..25

4.2 Етапи налагодження………………………………………………………....25

4.3 Типи помилок………………………………………………………………..26

5 ОЧІКУВАНІ ТЕХНІКО-ЕКОНОМІЧНІ ПОКАЗНИКИ…………………....27

5.1 Розрахунок вартості програмного продукту……………………………....27

5.2 Розрахунок економічної ефективності від впровадження програмного продукту………………………………………………………………………....36

5.3 Наукова організація праці на робочому місці програміста……………....42

6 ОХОРОНА ПРАЦІ…………………………………………………………....45

ВИСНОВКИ…………………………………………………..………………...57

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ………………………………………….. 58

Додаток А Схема алгоритму програми………………………………………. 59

Додаток Б Роздрук результатів роботи програми на

контрольному прикладі………………………………………………………. 60

Додаток В Значення коефіцієнта використання технологій

програмування та інструментів середовища розробки……………………. 64

Перелік інших частин:

1. Текст програми ДП.ХПФК.00452 –………………………………………. 65

2. Посібник програміста ДП.ХПФК.00452 – ………………………………..143

3. Посібник оператора ДП.ХПФК.00452– …………………………………...145

4. Специфікація ДП.ХПФК.00452 – …………………………………………147

# ВСТУП

Автоматизація є дуже важливою частиною в бізнесі, особливо в торговому. Також автоматизація процесів зменшить кількість помилок, спричинених людським фактором. Часто підприємці автоматизовують свій бізнес через збільшення даних та неможливість керування ними. У чому саме полягає автоматизація, це запровадження додатку в свій бізнес, до цього додатку прив’язується база даних, з якої будуть братись дані. Такий додаток буде застосовуватись в сфері торгівлі господарських товарів.

Область торгівлі продуктових товарів включає великий асортимент продукції. Саме тому автоматизація даної галузі є виправданою і доцільною. Тому що користувач може задовольнити більшість своїх потреб в одному магазині.

Метою дипломного проєкту є створення програмного продукту, який буде використовуватись касиром та адміністратором продуктового супермаркету.

Завдання дослідження − розробити адекватний меті дослідження інструментарій: додаток, робота якого буде спрямована на мінімізацію помилок і на пришвидшення роботи магазину та підвищення прибутковості з

максимальним скороченням працівників, що призведе до зменшення витрат на заробітні плати та підвищення прибутковості магазину.

Методи дослідження даного об’єкта. Моніторинг та аналіз інформаційних джерел об’єкта. Аналіз дає змогу розділяти предмети дослідження на складові частини. Моніторинг - метод постійного спостереження за станом окремих об'єктів, перебігом певних процесів в окремих системах. Даний метод є дуже продуктивним, він допомагає оцінити стан, у якому перебуває предметна область на даний час.

# ТЕХНІЧНІ ХАРАКТЕРИСТИКИ

## Постановка задачі

### Характеристика об’єкта автоматизації

Продуктовий супермаркет - це комплексна торгівельна організація, яка спеціалізується на продажі різноманітних продуктів харчування, товарів для дому та особистої гігієни. Продуктові супермаркети мають розгалужену мережу підрозділів, які знаходяться в різних районах міста та регіонів.

Основні функції продуктового супермаркету включають продаж продуктів харчування, управління запасами та планування закупівель, керування продажами та клієнтською базою даних, ведення бухгалтерського обліку.

Для автоматизації діяльності продуктового супермаркету необхідне програмне забезпечення, яке дозволить ефективно управляти складським обліком, контролювати рух товарів, планувати закупівлі, керувати продажами та клієнтською базою даних, а також вести бухгалтерський облік. Важливим параметром є також маржинальність продуктів та обсяги продажів.

Для забезпечення ефективної автоматизації продуктового супермаркету, програмне забезпечення повинно бути інтегровано з програмами обліку та управління запасами, які вже використовуються в компанії. Це дозволить створити єдину інформаційну систему, яка забезпечить повну інтеграцію всіх підрозділів та ефективний контроль за управлінням компанією.

### 1.1.2 Вимоги до програми

Призначення програмного забезпечення та програмного продукту: розробка системи автоматизації управління продажами в продуктовому супермаркеті.

Перелік задач, які будуть вирішуватися в результаті використання розробленого програмного продукту:

* Облік товарів на складі;
* Керування закупівлями товарів;
* Ведення обліку продажів;
* Взаємодія з клієнтами та програма лояльності;
* Генерація звітів та аналіз даних.

Структура програмного продукту має наступний вигляд:

* База даних товарів та клієнтів;
* Модуль замовлення товарів;
* Модуль продажів та створення звітів;
* Модуль програми лояльності;
* Модуль аналізу даних.

Опис функціонування програмного продукту та його частин:

* Модуль замовлення товарів дозволяє закуповувати товари та зберігати інформацію про склад;
* Модуль продажів веде облік продажів та може генерувати звіти по продажам;
* Модуль програми лояльності дозволяє взаємодіяти з клієнтами та створювати пропозиції для їх мотивації до покупок;
* Модуль аналізу даних аналізує дані та генерує звіти для управління та оптимізації бізнес-процесів.

Користувачами програмного продукту можуть бути касир, що використовує програму на робочому комп’ютері за касою під час взаємодії з клієнтами. Також програма може використовуватись адміністратором супермаркету, який може виконувати всі вище згадані задачі.

## Опис математичного методу рішення задачі

При розробці поставленої задачі було використано метод об’єктно-орієнтованого програмування.

Об’єктно-орієнтоване програмування (ООП) — одна з парадигм програмування, яка розглядає програму як множину взаємодіючих об'єктів. У ній використано декілька технологій від попередніх парадигм, включаючи успадкування, модульність, поліморфізм та інкапсуляцію. Об'єктно-орієнтоване програмування сягає своїм корінням до створення мови програмування Симула в 1960-тих роках, одночасно з посиленням [дискусій](http://ua-referat.com/%D0%94%D0%B8%D1%81%D0%BA%D1%83%D1%81%D1%8B) про кризу програмного забезпечення. Незважаючи на те, що ця [парадигма](http://ua-referat.com/%D0%9F%D0%B0%D1%80%D0%B0%D0%B4%D0%B8%D0%B3%D0%BC%D0%B0) з'явилась в 1960-тих роках, вона не мала широкого застосування до 1990-тих.

Об'єктно-орієнтовані програми можна розглядати як множину взаємодіючих об'єктів, на відміну від традиційних поглядів, коли програма розглядається як набір підпрограм, або як перелік інструкцій комп'ютеру. [Відповідно](http://ua-referat.com/%D0%92%D1%96%D0%B4%D0%BF%D0%BE%D0%B2%D1%96%D0%B4%D1%8C) до парадигми ООП, кожен об'єкт здатен отримувати повідомлення, обробляти дані, та надсилати повідомлення іншим об'єктам. Кожний об'єкт можна розглядати як незалежний [автомат](http://ua-referat.com/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82) з окремим призначенням або відповідальністю.

Об'єктно-орієнтований підхід полягає в наступному наборі основних принципів:

* все є об'єктами;
* всі дії та розрахунки виконуються шляхом взаємодії (обміну даними) між об'єктами, при якій один об'єкт потребує, щоб інший об'єкт виконав деяку дію. Об'єкти взаємодіють, надсилаючи і отримуючи повідомлення. Повідомлення - це запит на виконання дії, доповнений набором аргументів, які можуть знадобитися при виконанні дії;
* кожен об'єкт має незалежну пам'ять, яка складається з інших об'єктів;
* кожен об'єкт є представником (екземпляром, примірником) класу, який виражає загальні властивості об'єктів;
* у класі задається поведінка (функціональність) об'єкта. Таким чином усі об'єкти, які є екземплярами одного класу, можуть виконувати одні й ті ж самі дії;
* класи організовані у єдину деревовидну структуру з загальним корінням, яка називається ієрархією [успадкування](http://uk.wikipedia.org/wiki/%D0%A3%D1%81%D0%BF%D0%B0%D0%B4%D0%BA%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%28%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%29). Пам'ять та поведінка, зв'язані з екземплярами деякого класу, автоматично доступні будь-якому класу, розташованому нижче в ієрархічному дереві.

Об’єктно-орієнтоване програмування засноване на трьох принципах:

* інкапсуляції;
* наслідуванні;
* поліморфізмі.

Спостережуване в об'єктах об'єднання даних і операцій в одне ціле було позначене [терміном](http://ua-referat.com/%D0%A2%D0%B5%D1%80%D0%BC%D1%96%D0%BD%D0%B8) інкапсуляція. Застосування інкапсуляції зробило об'єкти схожими на маленькі програмні модулі і забезпечило приховування їх внутрішньої структури. Для об'єктів з'явилося [поняття](http://ua-referat.com/%D0%9F%D0%BE%D0%BD%D1%8F%D1%82%D1%82%D1%8F) інтерфейсу, що значно підвищило їх надійність і цілісність.

Наслідування означає, що при створенні нового класу, немає необхідності в переписуванні заново всіх полів, методів і властивостей. Ви оголошуєте, що новий клас є нащадком (або дочірнім класом) наявного класу, так званим предком (або батьківським класом), і додаєте до нього нові поля, методи та властивості. Іншими словами, додається те, що потрібне для переходу від загального до приватного. [Процес](http://ua-referat.com/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81) створення нових класів на основі інших класів називається наслідуванням. Нові класи мають як успадковані ознаки, так і нові.

Поліморфізм означає, що в похідних класах можна змінювати роботу методів, що вже існують в базовому класі. При цьому весь програмний код, що керує об'єктами батьківського класу, придатний для управління об'єктами дочірнього класу без будь-якої модифікації.

Також у програмі використано ієрархію класів для створення динамічного графічного інтерфейсу та взаємодії даних.

## Визначення структури вхідних даних

Вхідними даними називають такі змінні (параметри, дані), які надаються виконавцеві до початку його роботи над алгоритмом.

Застосування алгоритму до неприпустимих(або неперевірених) вхідних даних може призводити до того, що алгоритм ніколи не зупиниться, або потрапить в тупиковий стан (зависання), з якого не зможе продовжитися виконання процесу.

Таблиця 1.1 – Вхідна інформація

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва | Форма подання | Термін і частота використання |
| 1 | Товар | Масив оперативної інформації | Не обмежено |
| 2 | Працівники | Масив нормативно-довідкової інформації | Не обмежено |
| 3 | Клієнт | Масив нормативно-довідкової інформації | Не обмежено |
| 4 | Товарна накладна | Масив нормативно-довідкової інформації | Не обмежено |

Вхідні дані зазвичай поділяють на два класи: довідкову та оперативну інформацію. Довідкові дані змінюються рідше, ніж оперативні, які можуть змінюватися навіть в режимі реального часу.

Таблиця 1.2 – Структура вхідних даних

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № з/п | Ідентифікатор | Тип | Діапазон значень | Пояснення |
| 1 | 2 | 3 | 4 | 5 |
| 1. | Pr\_Id | Integer | 0..2147483647 | Код товару |
| 2. | Pr\_BarCode | String | 0..20 | Штрих-код |
| 3. | Pr\_Description | String | 0..200 | Опис товару |
| 4. | Pr\_Brand | String | 0..30 | Виробник |
| 5. | Pr\_Title | String | 0..200 | Назва товару |
| 6. | Pr\_Category | String | 0..30 | Категорія |
| 7. | Pr\_Price | Decimal | 0..2147483647 | Ціна |
| 8. | Pr\_PriceOpt | Decimal | 0..2147483647 | Оптова ціна |
| 9. | Pr\_Qty | Integer | 0..2147483647 | Кількість |
| 10. | Emp\_Id | Integer | 0..2147483647 | Код працівника |
| 11. | Emp\_Name | String | 0..50 | Ім’я працівника |
| 12. | Emp\_Postion | String | 0..30 | Посада |
| 13. | Emp\_Salary | Decimal | 0..100000 | Зарплата |
| 14. | Sti\_Id | Integer | 0..2147483647 | Код замовлення |
| 15. | Sti\_Date | DateTime |  | Дата замовл. |
| 16. | Sti\_StockInBy | String | 0..30 | Замовник |
| Продовження таблиці 1.2 | | | | |
| 17. | Sti\_Status | String | 0..20 | Статус замовлення |
| 18. | Sti\_Supplier | String | 0..50 | Постачальник |
| 19. | Sti\_IsConfirmed | Bolean | true/false | Підтвердження замовлення |
| 20. | Cl\_Id | Integer | 0..2147483647 | Код клієнта |
| 21. | Cl\_Name | String | 0..30 | Ім’я клієнта |

## Визначення структури вихідних даних

Вихідною інформацією задачі для користувача будуть звіти та форми, які будуть відображатися на під час роботи з програмою як таблиці та форми. Також вихідна інформація може бути роздрукована . Вихідні дані – це дані, які виконавець видає як результат здійсненої роботи. Вихідними даними у програмному додатку є інформація про товари, чеки, клієнтів та працівників. В таблиці 1.3 показані вихідні дані, які використовуються в дипломному проєкті. В таблиці 1.4 показана структура вихідніх даних, які використовуються в даному дипломному проєкті.

Таблиця 1.3 - Вихідна інформація

|  |  |  |  |
| --- | --- | --- | --- |
| № | Назва | Форма | Звіт |
| 1 | Товари | + | + |
| 2 | Чек |  | + |
| 3 | Залишки на складі | + | + |
| Продовження таблиці 1.3 | | | |
| 4 | Клієнти |  | + |
| 5 | Транзакції покупок | + | + |
| 7 | Працівники | + | + |

Таблиця 1.4 – Структура вихідних даних

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № з/п | Ідентифікатор | Тип | Діапазон значень | Пояснення |
| 1 | 2 | 3 | 4 | 5 |
| 1. | Pr\_Id | Integer | 0..2147483647 | Код товару |
| 2. | Pr\_BarCode | String | 0..20 | Штрих-код |
| 3. | Pr\_Description | String | 0..200 | Опис товару |
| 4. | Pr\_Brand | String | 0..30 | Виробник |
| 5. | Pr\_Title | String | 0..200 | Назва товару |
| 6. | Pr\_Category | String | 0..30 | Категорія |
| 7. | Pr\_Price | Decimal | 0..2147483647 | Ціна |
| 8. | Pr\_Image | Binnary |  | Зображення товару |
| 8. | Pr\_PriceOpt | Decimal | 0..2147483647 | Оптова ціна |
| 9. | Pr\_Qty | Integer | 0..2147483647 | Кількість |
| 10. | Emp\_Id | Integer | 0..2147483647 | Код працівника |
| 11. | Emp\_Name | String | 0..50 | Ім’я працівника |
| 12. | Emp\_Postion | String | 0..30 | Посада |
| Продовження таблиці 1.4 | | | | |
| 13. | Emp\_Salary | Decimal | 0..100000 | Зарплата |
| 14. | Sti\_Id | Integer | 0..2147483647 | Код замовлення |
| 15. | Sti\_Date | DateTime |  | Дата замовлення |
| 16. | Sti\_StockInBy | String | 0..30 | Замовник |
| 17. | Sti\_Status | String | 0..20 | Статус замовлення |
| 18. | Sti\_Supplier | String | 0..50 | Постачальник |
| 19. | Sti\_IsConfirmed | Bolean | true/false | Підтвердження замовлення |
| 20. | Cl\_Id | Integer | 0..2147483647 | Код клієнта |
| 21. | Cl\_Name | String | 0..30 | Ім’я клієнта |

# КОМП’ЮТЕРНА СИСТЕМА

## Технічні характеристики комп’ютера та зовнішніх пристроїв

Для роботи розробленої прикладки використовувався персональний комп’ютер (ПК) з наступними характеристиками:

* Процесор AMD Ryzen 7 4800 H 2.9 GHz;
* оперативна пам’ять 16 ГБ DDR4 3200 MHz;
* відеокарта NVIDIA GTX 1650 Ti;
* монітор 1920x1080 60Hz;
* операційна система Windows 11 Pro;

Програма може функціонувати без втрати всіх функціональних можливостей на комп’ютері з такими технічними характеристиками:

- процесор з частотою 1ггц;

- RAM 128 MB ;

- відеокарта Intel HD Graphics;

- операційна система Windows, Mac, Linux, Chrome OS.

## Вибір програмних засобів та операційної системи

Для розробки веб-сайту було обрано такі засоби реалізації:

* мова програмування С# на платформі .NET;
* СУБД MSSQL Server;
* Microsoft RDLC дизайнер звітів.

C# є об'єктно-орієнтованою мовою програмування, що дозволяє розробникам створювати програми, які можуть бути легко розширені та підтримувані з часом. Мова підтримує багато функцій, які дозволяють розробникам писати якісний та ефективний код. C# використовується для створення різноманітних програм, включаючи веб-додатки, мобільні додатки, ігри та програми для роботи з базами даних. Мова є основою для платформи .NET, що дозволяє розробникам створювати програми, які можуть працювати на різних операційних системах, таких як Windows, Linux та macOS.

Переваги мови програмування C#:

1. Об'єктно-орієнтований підхід: C# є повноцінною об'єктно-орієнтованою мовою програмування, що дозволяє розробникам створювати більш прості, ефективні та складні програми.
2. Безпека: C# має вбудований механізм безпеки, який дозволяє розробникам створювати більш безпечні програми, знижуючи ризик вразливостей та атак зловмисників.
3. Платформно-незалежний код: С# розроблена для роботи на платформі .NET, що дозволяє розробникам створювати програми, які можуть працювати на різних операційних системах.
4. Бібліотеки та фреймворки: C# має велику кількість сторонніх бібліотек та фреймворків, що дозволяє розробникам створювати програми швидко та ефективно.

Недоліки мови програмування C#:

1. Залежність від платформи: C# була розроблена для платформи .NET, тому програми, написані на C#, можуть працювати тільки на операційній системі Windows або на платформі .NET.
2. Швидкість виконання: Порівняно з мовами програмування, такими як C та C++, виконання програм на C# може бути менш ефективним та повільним.
3. Низький рівень контролю над пам'яттю: Хоча C# має вбудований збірник сміття, що допомагає управляти пам'яттю, розробникам може бути складніше контролювати пам'ять.

MS SQL є однією з найпоширеніших реляційних баз даних, розроблених корпорацією Microsoft. Ця система управління базами даних дозволяє ефективно зберігати та організовувати великі об'єми даних. MS SQL забезпечує високу надійність, безпеку та продуктивність завдяки своїм функціям, таким як механізми резервного копіювання, відновлення даних та захист від вторгнень. Ця система має широкий спектр можливостей, включаючи розширені можливості звітування, інструменти аналізу даних та можливість створювати складні запити для отримання потрібної інформації.

Переваги MS SQL:

1. Надійність та стабільність: MS SQL має високий рівень надійності та стабільності завдяки своїм механізмам резервного копіювання та відновлення даних.
2. Висока продуктивність: MS SQL забезпечує високу продуктивність завдяки своїм оптимізаційним можливостям та підтримці паралельної обробки даних.
3. Розширені можливості звітування: MS SQL має вбудовані інструменти для створення різноманітних звітів та аналізу даних.
4. Інтеграція з продуктами Microsoft: MS SQL інтегрується з іншими продуктами Microsoft, що дозволяє ефективно працювати з даними та ділитися ними між різними системами.
5. Масштабованість: MS SQL може легко масштабуватися залежно від потреб користувачів та організацій.

Недоліки MS SQL:

1. Високі витрати: MS SQL є комерційним продуктом, що означає високі витрати на ліцензування та підтримку.
2. Складність: MS SQL може бути складним у використанні для новачків, оскільки має велику кількість функцій та можливостей.
3. Обмежена підтримка інших платформ: MS SQL більш спрямований на платформу Windows та менш підтримує інші операційні системи.
4. Невелика спільнота користувачів: порівняно з деякими іншими реляційними базами даних, MS SQL має меншу спільноту користувачів, що може ускладнити пошук допомоги та розв'язання проблем.

Microsoft RDLC (Report Definition Language Client-Side) Designer є інструментом для створення звітів у середовищі Visual Studio. Він дозволяє створювати різноманітні звіти, використовуючи різні джерела даних, такі як бази даних, веб-сервіси, Excel-файли тощо. RDLC Designer забезпечує зручний інтерфейс для налаштування форматування звітів, додавання тексту, зображень, графіків, таблиць, графіків та інших елементів. Після створення звіту, його можна зберегти в форматі RDLC або скомпілювати в DLL-файл, щоб використовувати його в програмному коді. RDLC Designer є частиною Microsoft SQL Server Reporting Services (SSRS) та може бути використаний для створення звітів у багатьох типах проектів, включаючи ASP.NET, WinForms, WPF, Silverlight тощо.

# КОМП’ЮТЕРНА СИСТЕМА

## Технічні характеристики комп’ютера та зовнішніх пристроїв

Цінність будь-якої програми визначається не тільки її здатністю безпомилково і якісно вирішувати поставлену задачу, а й тим, наскільки легко і зручно цією програмою можна користуватися. Навіть найпростіша, і в той же час, функціональна програма виявиться незатребуваною, якщо користувач (тобто кінцевий споживач програми) при роботі з нею швидко втомлюється, дратується, відчуває себе некомфортно. Під призначеним для користувача інтерфейсом (англ. User interface) будемо розуміти своєрідний комунікаційний канал (сполучна ланка), за допомогою якого користувач взаємодіє з комп'ютером.

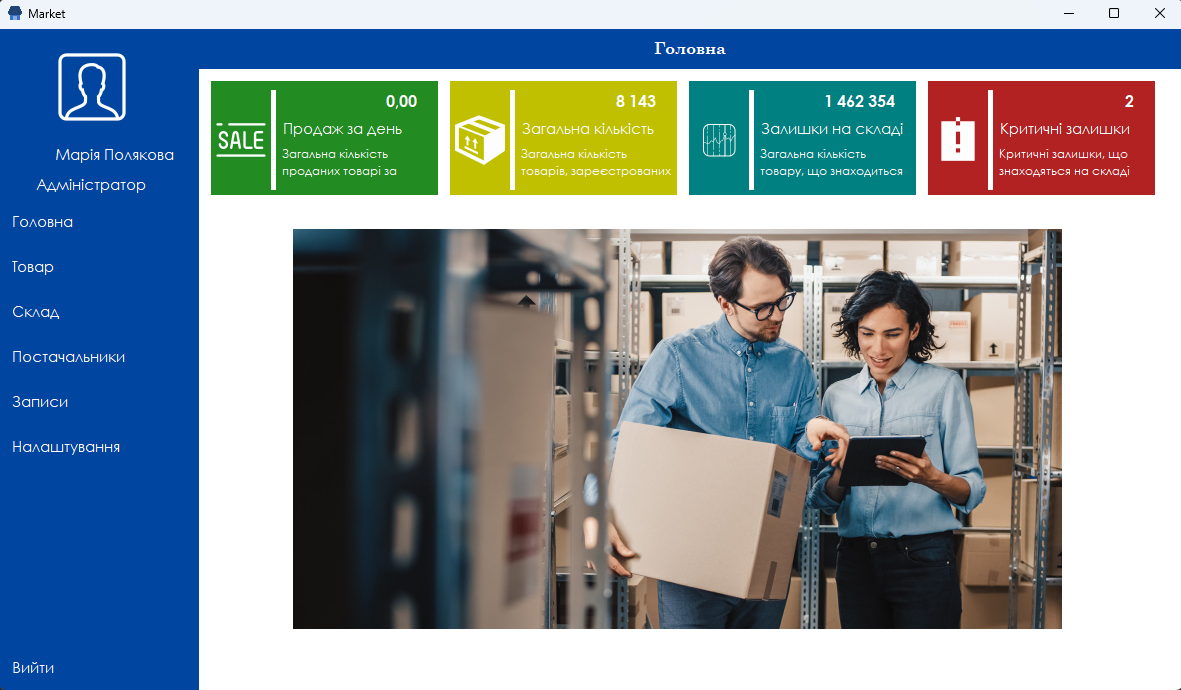
З точки зору користувача саме інтерфейс є кінцевим продуктом. Інтерфейс являє собою сукупність використовуваних в програмі засобів для введення даних, способів відображення інформації на екрані дисплея, елементів для управління різними етапами рішення задачі. Основне призначення (головна мета) призначеного для користувача інтерфейсу - забезпечити короткий і зручний шлях до отримання результату.

Інтерфейс користувача комп’ютерного додатку включає:

* Ясність;
* Виразність;
* Відповідність;
* Послідовність;
* Продуктивність;

Невдалі інтерфейси сильно уповільнюють роботу користувачів, сприяють швидкої стомлюваності, великому числу помилок і, як наслідок, відбивають у користувачів подальше бажання працювати з програмою. У зв'язку з цим розробнику програми слід враховувати відомі рекомендації провідних фахівців в області створення призначеного для користувача інтерфейсу.

Саме тому було розроблено максимально комфортний і зручний для користування інтерфейс (див. рис. 4.1)



3.2 Алгоритм рішення задачі

Програмний додаток складається з багатьох екранних форм, кожна за яких виконує певну функцію та має унікальну логіку роботи.

При запуску програми спочатку відкривається форма де користувачу потрібно ввести логін та пароль. Після чого на основі ролі користувача відкриється або головна форма для адміністратора або головна форма для касира. Кожна з яких має свій унікальний функціонал та інтерфейс.