

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА
ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни “Об’єктно -орієнтоване програмування”

Магазин з одним продавцем

Керівник, професор

Бондарев В. М.

Студент гр. ПЗПІ-19-4

Кашепаров Р.І.

Комісія:

старший викладач	_____	Черепанова Ю.Ю.,
професор	_____	Бондарев В. М.,
доцент	_____	Побіженко І. О.

Харків 2020

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ**

Кафедра: *Програмної інженерії*

Дисципліна: *Об'єктно-орієнтоване програмування*

Спеціальність: *121 Інженерія програмного забезпечення*

Освітня програма: *Програмна інженерія*

Курс 1.

Група *ПЗПІ-19 -4*. Семестр 2.

ЗАВДАННЯ
на курсовий проект студента

(Прізвище, Ім'я, По батькові)

1 Тема проекту: *Магазин з одним продавцем*

2 Термін здачі студентом закінченого проекту: ***"19" - червня - 2020 р.***

3 Вихідні дані до проекту:

Специфікація програми, методичні вказівки до виконання курсової роботи

4 Зміст розрахунково-пояснювальної записки:

Вступ, специфікація програми, проектна специфікація, інструкція користувача, висновки

5 Перелік графічного матеріалу:

Схема об'єктної моделі, алгоритми, приклади екранних форм

КАЛЕНДАРНИЙ ПЛАН

<i>№</i>	<i>Назва етапу</i>	<i>Термін виконання</i>
1	Видача теми, узгодження і затвердження теми	21-02-2020 р.
2	Формулювання вимог до програми	21-02-2020 – 31-03-2020 р.
3	Розробка підсистеми зберігання та пошуку книг.	01-04-2020 – 08-04-2020 р.
4	Розробка підсистеми зберігання та пошуку читачів.	09-04-2020 – 23-04-2020 р.
5	Розробка функцій видачі та повернення книг	24-04-2020 – 27-04-2020 р.
6	Розробка функцій зберігання та завантаження даних	28-04-2020 – 31-04-2020 р.
7	Тестування і доопрацювання розробленої програмної системи.	01-05-2020 – 15-05-2020 р.
8	Оформлення пояснювальної записки, додатків, графічного матеріалу	20-05-2020 – 05-06-2020 р.
9	Захист	01-06-2018 – 19-06-2020 р.

Студент _____

Керівник _____

(Прізвище, Ім'я, По батькові)

« 21 » лютого _____ 2020 р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 35 с., 15 рис., 2 додатки, 5 джерел.

АДМІНІСТРАТОР, КОРИСТУВАЧ, ЗАМОВЛЕННЯ, ТОВАР, КЛАС, МОВА ПРОГРАМУВАННЯ C#, ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, ЗВІТ, ПЛАТФОРМА .NET, ПРОГРАМА.

Метою роботи є розробка програми «Магазин з одним продавцем» на засадах об'єктно-орієнтованого програмування для управління та експлуатації магазину.

Методи розробки базуються на використанні середовища розробки Microsoft Visual Studio 2019, Windows Forms, платформи .NET Framework 4.7.2, мови програмування C#.

У результаті отримана програма під назвою «Магазин з одним продавцем», яка дозволяє користувачу переглядати товари в магазині, та робити замовлення. А продавець має змогу створювати нові товари, змінювати вже існуючі, замовляти поставки та отримувати звіти.

ЗМІСТ

ВСТУП	7
1 СПЕЦИФІКАЦІЯ ПРОГРАМИ	8
1.1 Користувачі програми	8
1.2 Функції програми.....	8
1.3 Функції адміністратора	9
Функція «Вхід в програму»	9
Функція «Зміна асортименту».....	10
Функція «Історія зміни цін»	12
Функція «Облік поставок товарів»	12
Функція «Облік продажів товарів».....	13
Функція «Уцінка та списання».....	13
Функція «Вихід з програми»	13
Функція «Отримання звіту»	14
1.4 Функції покупця.....	14
Функція «Вхід в програму»	14
Функція «Історія»	16
Функція «Вихід з програми»	16
2 ПРОЕКТНА СПЕЦИФІКАЦІЯ	17
2.1 Архітектура	17
2.2 Структура проекту	17
2.3 Об'єктна структура.....	18
2.3.1 Клас User:	19
2.3.2 Клас Admin:.....	19
2.3.3 Клас Item:.....	19
2.3.4 Клас Portion:	20
2.3.5 Клас Supply:	20
2.3.6 Клас Dao:	21
2.3.6 Клас Shop:.....	21
2.4 Діаграма класів.....	22
3 ІНСТРУКЦІЯ КОРИСТУВАЧА	23
3.1 Встановлення, запуск та головне вікно	23
3.2 Використання користувачами	24
3.2.1 Додавання продукту до козиини	24
3.2.2 Зміна продукту до козиини.....	25
3.2.3 Видалення продукту з козиини	25

3.2.4 Закінчення покупки	25
3.3 Використання адміністратором	26
3.3.1 Додавання нового товару	26
3.3.2 Змінна існуючого товару	27
3.3.3 Видалення товару	27
3.3.4 Дізнатися історію зміни цін.....	27
3.3.5 Створення нової поставки.	28
3.3.6 Змінна існуючої поставки	29
3.3.7 Видалення поставки	29
3.3.8 Створення звіту.....	30
3.3.9 Закінчення сеансу	30
ВИСНОВКИ.....	31
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	32
ДОДАТОК А	33
Методи класу Shop	33
ДОДАТОК Б Методи класу Dao	35

ВСТУП

Зараз все більше магазинів бажають діджіталізувати своє підприємство. Продавці мають змогу об'єднати процес редагування бази наявних товарів, замовлення поставок, відстеження замовлень та створення звітів.

Розроблена програма дозволить користувачу виконувати усі згадані вище дії. Адміністратор отримує можливість зменшити свої обов'язки – програма буде самостійно реєструвати товари (як нові, так і старі), оформлювати покупки та інвентаризувати залишки товари. Покупець має змогу робити замовлення дистанційно та цілодобово.

Окрім розробки вище описаної програмної системи метою даної курсової роботи є освоїти мову програмування C#, навчитися об'єктно-орієнтованому програмуванню, розвинути навички роботи в Windows Forms.

Задачі роботи:

- Розробка системи зчитування та запису даних;
- Створення логіки роботи системи;
- Робота з користувачем;
- Розробка логіки додавання, пошуку та видалення інформації;
- Тестування готового продукту;
- Створення стійкості програми.

1 СПЕЦИФІКАЦІЯ ПРОГРАМИ

1.1 Користувачі програми

У проекту є дві категорії користувачів, яким необхідні різні функції програми - це адміністратор і покупці.

Продавець додає товари до бази. Відстежує наявність товару, редагує його, створює поставки, отримує звіти.

Про товар відомо: найменування, зображення, одиниця виміру, вартість продажу, наявна кількість на складі.

Про покупця відомо - ім'я, пароль, історія.

Покупець шукає товари, наповнює кошик, робить замовлення.

Належність користувача до покупців або адміністраторам визначається в момент входу в програму.

Питання безпеки, такі як хешування паролів і безпеку даних в програмі не вирішуються.

1.2 Функції програми

Програма надає покупцеві такі можливості:

1. Реєстрація.
2. Вхід до програми.
3. Пошук товару.
4. Робота з кошиком.
5. Відстежування історії.
6. Вихід з програми.

Програма надає адміністратору наступні можливості:

1. Вхід до програми.
2. Ведення списку можливих товарів.

3. Облік поставок товарів.
4. Облік продажу товарів.
5. Отримання звіту.
6. Вихід з програми.

Загальні функції:

1. Завантаження даних з постійної пам'яті в оперативну.
2. Збереження поточного стану даних в постійній пам'яті.
3. Генерація тестових даних.

1.3 Функції адміністратора

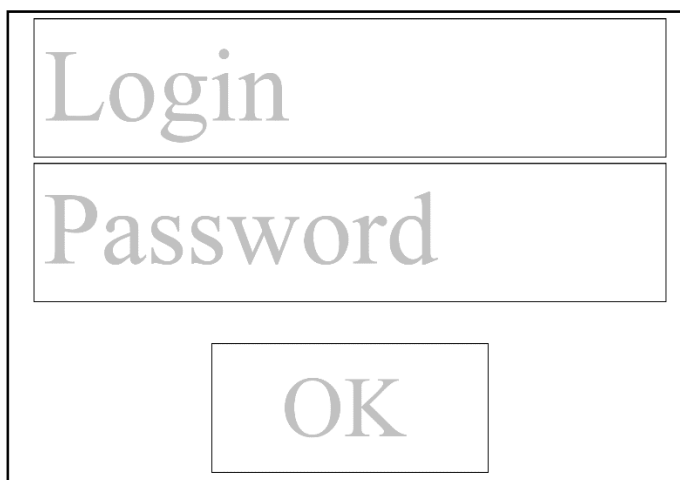
Функція «Вхід в програму»

Адміністратор входить в програму за допомогою логіна та пароля. Програма автоматично визначає роль користувача.

Пароль адміністратора зберігаються в даних програми.

Основний сценарій

1. Адміністратор вводить логін та пароль у вікні входу (рис. 1.1) і натискає кнопку "ОК".
2. Введені дані перевіряються, і відкривається адмінпанель - головне вікно адміністратора (рис. 1.2).
4. Якщо дані не пройшли перевірку, на формі введення з'являється повідомлення про це і можна спробувати увійти знову. Число спроб не обмежена.



The diagram shows a rectangular window with a thin black border. Inside the window, there are three rectangular input fields. The top field contains the text 'Login', the middle field contains 'Password', and the bottom field contains 'OK'. The text is in a large, serif font and is centered within each field. The fields are stacked vertically with some space between them.

Рисунок 1.1 – Вікно входу адміністратора

Функція «Зміна асортименту»

Список товарів включає в себе товари взагалі, а не тільки ті, що доступні в даний момент.

Цей список використовується при введенні надходжень, формуванні кошика, звітів, тощо.

Адміністратор повинен мати можливість поповнювати список по мірі необхідності. Зокрема, така необхідність напевно буде виникати при введенні даних про чергове надходження.

Реалізована можливість не тільки поповнювати список, але і виправляти данні, оновлювати зображення товарів і навіть видаляти окремі товари, але тільки ті, які ще не беруть участь в поставках зараз.

Menu

All products	Users	All orders
<div>New product</div>	<div>User 1</div>	<div>New order</div>
<div>Item 1</div>	<div>User 2</div>	<div>Order 1</div>
<div>Item 2</div>	<div>History</div>	<div>Order 2</div>
<div>Item 3</div>	<div>Item 1</div>	<div>Order 3</div>
<div>Item 2</div>	<div>Item 2</div>	

Report

Рисунок 1.2 – Адмінпанель

Сценарій «Додавання товару»

1. На адмінпанелі натискає кнопку «Новий Товар».
2. Відкривається форма додавання товару. Поля форми порожні.
3. Заповнює поля форми, обирає файл зображення.
4. Натискає кнопку «Зберегти».

Сценарій «Зміна товару»

1. На адмінпанелі вибирає товар із загального переліку.
2. Натискає кнопку «Змінити» або робить подвійне клацання на обраному товарі.
3. Відкривається вікно інформації про товар. Відкриває форму зміни товару. Поля форми заповнені.
4. Змінює поле форми, змінює файл зображення.
5. Натискає кнопку «Зберегти».

Сценарій «Видалення товару»

1. Якщо товар не бере участь в транзакціях, то на адмінпанелі натискає кнопку «Видалити Товар», інакше програма заборонне його видаляти.
2. Підтверджує видалення товару.
3. Після отримання підтвердження товар видаляється, список оновлюється.
4. Обраним стає товар, що передує віддаленим комп'ютером.

Функція «Історія зміни цін»

Адміністратор має можливість продивлятися зміну цін, відкривши вікно редагування товару та відкривши відповідне вікно.

Функція «Облік поставок товарів»

Поставка товарів відбувається раз на 2 дні.

Інформація про постачання включає дату поставки і перелік поставлених товарів.

Один пункт переліку містить: Дату та кількість товарів у поставці.

Сценарій «Нова поставка»

1. На адмінпанелі адміністратор натискає кнопку «Нова поставка»
2. Відкривається вікно введення нової поставки (рис. 1.3).
3. Вводиться перший пункт переліку - із загального списку товарів вибирається товар, вказується кількість.
4. Вводяться інші пункти переліку, і у вікні введення формується список пунктів.
5. Елементи цього списку можна редагувати і видаляти.

6. Коли список поставки остаточно сформований, адміністратор натискає кнопку «Зберегти».

7. Список всіх поставок поповнюється новим елементом.

New order		
Name	Amount	Price
<input type="text"/>		
<input type="text"/>		
<input type="text"/>		
Total:		
		<input type="button" value="OK"/> <input type="button" value="Cancel"/>

Рисунок 1.3 – Вікно нового заказу

Функція «Облік продажів товарів»

Після того, як покупець зробив замовлення, він з'являється в історії зроблених замовлень даного покупця.

Адміністратор бачить цей список в колонці адмінпанелі, обравши необхідного покупця.

Функція «Уцінка та списання»

Адміністратор у праві робити уцінку (змінювати ціну) товар, або навіть списувати (видаляти) його за непотрібністю. Програма самостійно проаналізує можливість даних операцій.

Функція «Вихід з програми»

У процесі роботи користувач змінює дані, які знаходяться в оперативній пам'яті.

Якщо дані змінені, але ще не введено постійної пам'яті, то перед закриттям програми користувач отримує пропозицію зберегти дані.

Сценарій повинен бути таким же, як при завершенні роботи з будь-яким текстовим редактором, тому тут не наводиться.

Функція «Отримання звіту»

Звіт - це текстовий файл з потрібними даними. Перед збереженням на диску адміністратор обирає шлях збереження файлу.

Сценарій

1. У меню адмінпанелі вибирається пункт Report
2. Створюються меню вибору шляху до файлу.
3. Користувач натискає кнопку ОК і звіт зберігається у вигляді текстового файлу MyShop.txt в обраному каталозі.

1.4 Функції покупця

Функція «Вхід в програму»

Покупець входить в програму за допомогою логіна та пароля. Програма автоматично визначає роль користувача.

Данні покупця зберігаються в даних програми.

Основний сценарій

1. Покупець вводить логін та пароль у вікні входу (рис. 1.1) і натискає кнопку "ОК".
2. Введені дані перевіряються, і відкривається панель покупок - головне вікно покупця(рис. 1.4).
4. Якщо дані не пройшли перевірку, на формі введення з'являється повідомлення про це і можна спробувати увійти знову. Число спроб не обмежена.

Користувач додає покупки в кошик.

Також може переглядати свої минулі покупки.

Menu											
<table border="1"> <thead> <tr> <th>Basket</th> </tr> </thead> <tbody> <tr> <td>New item</td> </tr> <tr> <td>Item 1</td> </tr> <tr> <td>Item 2</td> </tr> <tr> <td> </td> </tr> </tbody> </table>	Basket	New item	Item 1	Item 2		<table border="1"> <thead> <tr> <th>History</th> </tr> </thead> <tbody> <tr> <td>Item 1</td> </tr> <tr> <td>Item 2</td> </tr> <tr> <td>Item 3</td> </tr> <tr> <td>OK</td> </tr> </tbody> </table>	History	Item 1	Item 2	Item 3	OK
Basket											
New item											
Item 1											
Item 2											
History											
Item 1											
Item 2											
Item 3											
OK											

Рисунок 1.4 – Вікно покупця

Сценарій «Додавання покупок»

1. На панелі натискає кнопку «Додати до кошика»
2. Відкривається вікно додавання нового предмету (рис. 1.5).
3. Вводиться перший пункт переліку - із загального списку товарів вибирається товар, вказується кількість.
4. Список всіх покупок поповнюється новим елементом.

New item	
Name	Price
<div style="border: 1px solid black; height: 20px; width: 100%;"></div> <div style="text-align: right;">▼</div>	
<div style="border: 1px solid black; height: 20px; width: 100%;"></div> <div style="text-align: right;">▼</div>	
<div style="border: 1px solid black; height: 20px; width: 100%;"></div> <div style="text-align: right;">▼</div>	
Total:	
<div style="border: 1px solid black; padding: 2px 10px;">OK</div> <div style="border: 1px solid black; padding: 2px 10px;">Cancel</div>	

Рисунок 1.5 – Вікно додавання покупки

Сценарій «Видалення товару»

1. Користувач вибирає товар із загального переліку.
2. Користувач обирає відповідну команду у меню.
3. Після отримання підтвердження товар видаляється, список оновлюється.
4. Обраним стає товар, що передує віддаленим комп'ютером.

Функція «Історія»

Програма запам'ятовує покупки користувача, що надає можливість в майбутньому продивлятися історію своїх замовлень.

Функція «Вихід з програми»

Якщо користувач закінчує сеанс, натикнувши кнопку ОК, тобто бажаючи придбати товар, то саме в цей час данні які знаходяться в оперативній пам'яті вводяться до постійної пам'яті.

Якщо користувач закінчує сеанс, натикнувши кнопку відміни, то з'являється пропозиція зберегти кошик для наступного придбання.

Сценарій повинен бути таким же, як при завершенні роботи з будь-яким текстовим редактором, тому тут не наводиться.

2 ПРОЕКТНА СПЕЦИФІКАЦІЯ

2.1 Архітектура

Програма буде створюватися на мові C # з використанням середовища розробки Visual Studio 2019 на платформі .NET Framework 4.7.2 з використанням інтерфейсу програмування додатків Windows Forms.

Вибір мови обумовлений об'єктно-орієнтованим підходом до написання програми. На даний момент мова C # є одним з найпопулярніших, зручних і повноцінних серед об'єктно-орієнтованих мов програмування.

В процесі проектування розглядалися 2 платформи, з якими мова C # має найкраще взаємодію, а саме .NET Core і .NET Framework. Наш вибір припав на останню, через збільшеної кількості керуючих елементів і більш високого рівня розвитку платформи. На жаль, це означає що додаток буде працювати виключно на машинах з операційною системою Windows.

Для реалізації призначеного для користувача інтерфейсу було вирішено використовувати графічний інтерфейс так як він простий для сприйняття і інтуїтивно зрозумілий. Для реалізації інтерфейсу ми будемо використовувати інтерфейс Windows Forms який є частиною платформи .NET Framework. З його допомогою у нас буде можливість створити повноцінний інтерфейс для користувачів програми.

2.2 Структура проекту

Буде розумно поділити проект, описаний в специфікації, на три окремих додатки: вхід, для адміністратора і для клієнтів, оскільки функції цих типів користувачів між собою не перетинаються.

На цій підставі, можемо скласти наступну структуру проекту (рис. 2.1):

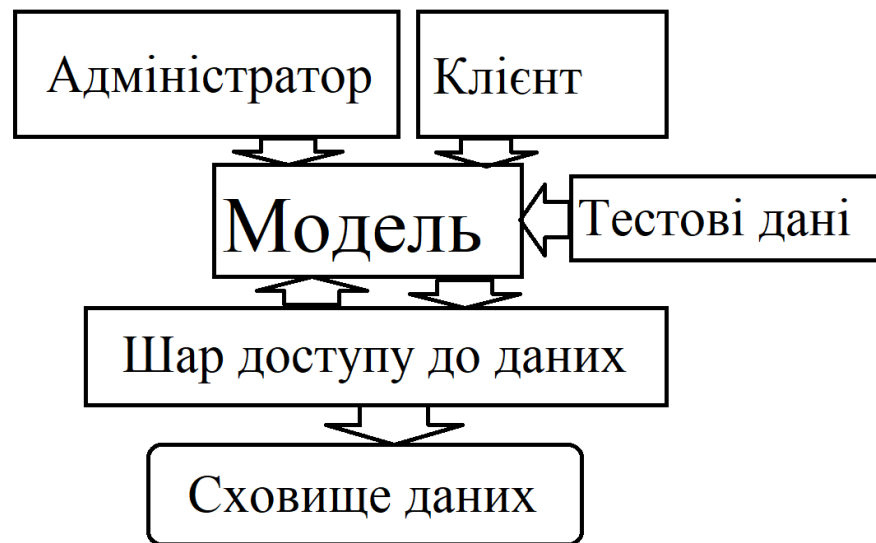


Рисунок 2.1 - Об'єктна модель

2.3 Об'єктна структура

Клієнт обирає бажані товари, тим самим наповнюючи свою корзину. Закінчує сеанс створенням замовлення.

Адміністратор додає нові товари, редагує вже створенні найменування, отримує звіти про наявність та продаж.

- Про товар відомо: назва, фото, наявна кількість, кількість придбань, одиниця виміру, історія цін.
- Про клієнта відомо - ім'я, пароль, історія купівель.
- Про адміністратора відомо - ім'я, пароль.
- Поставка - це дата + колекція порцій товару.
- Порція - це товар + його кількість.
- Магазин - це кілька колекцій: товарів, покупців, поставок, замовлень.

У моделі зручно виділити кореневий клас, який представляє модель в цілому.

Інші частини програми будуть працювати з моделлю через методи і властивості кореневого класу.

Він ніби є фасадом, за яким ховаються інші класи моделі.

У нашому прикладі очевидно, що це Магазин.

2.3.1 Клас User:

Клас User описує користувача.

```
//Ім'я користувача.  
public string Name { set; get; }  
  
//Пароль користувача.  
public string Password { set; get; }  
  
//Список історій купіль.  
public List<Portion> History { set; get; }  
  
//Кошик користувача.  
public List<Portion> Basket { set; get; }
```

2.3.2 Клас Admin:

Клас Admin описує користувача.

```
//Ім'я адміністратора.  
public string Name { set; get; }  
  
//Пароль адміністратора.  
public string Password { set; get; }
```

2.3.3 Клас Item:

Клас Item описує товар.

```
//Кількість проданих одиниць.  
public decimal Sold = 0;  
  
//Назва товару.
```

```

public string Name { set; get; }

//Одиниці вимірювання товару.
public string Unit { set; get; }

//Зображення товару.
public Image Image { set; get; }

// Список історій цін товару.
public List<decimal> Price { set; get; }

//Кількість доступного товару.
private decimal _available;
public decimal Available
{
    set => _available = value;
    get => _available - Sold;
}

```

2.3.4 Клас Portion:

Клас Admin описує порцію замовлення.

```

//Кількість товару.
public decimal Amount { set; get; }

//Обраний товар.
public Item Item { set; get; }

```

2.3.5 Клас Supply:

Клас Supply описує поставку.

```

//Список порцій.
public List<Portion> Portions { private set; get; }

//Дата створення замовлення.
public DateTime DateTime { set; get; }

//Дата виконання замовлення.

```

```
public DateTime DateTimeEnd => DateTime +
    TimeSpan.FromDays(2);
```

2.3.6 Клас Dao:

Клас Dao забезпечує доступ к даним, що знаходяться у ROM, і навпаки.

```
//Об'єкт магазину.
private Shop shop;

//Назва файлу, що буде збережено.
private const string path = "shop.bin";
```

2.3.6 Клас Shop:

Клас Supply зберігає усі данні про магазин.

```
//Список усіх товарів у магазині.
public List<Item> Items { private set; get; }

//Список існуючих користувачів.
public List<User> Users { private set; get; }

//Список ще не здійснених поставок.
public List<Supply> Supplies { private set; get; }

//Список виконаних поставок.
public List<Supply> HistorySupplies {private set;
get;}

//Дані адміністратора.
public Admin Admin;
```

Клас Shop має також методи (див. додаток А):

- public void FillTest(int n) – Створює тестові дані;
- public void AddUser(User user) – Додає нового користувача;
- public void AddItem(Item item) – Додає новий товар;

- `public void AddSupplyFirst(Supply supply)` – Додає нову поставку;
- `public void UpdateSupplies()` – Оновлю дані про виконані поставки;
- `public void Save()` – Зберігає магазин у постійну пам'ять;
- `public void Load()` – завантажує магазин з постійної пам'яті.

2.4 Діаграма класів

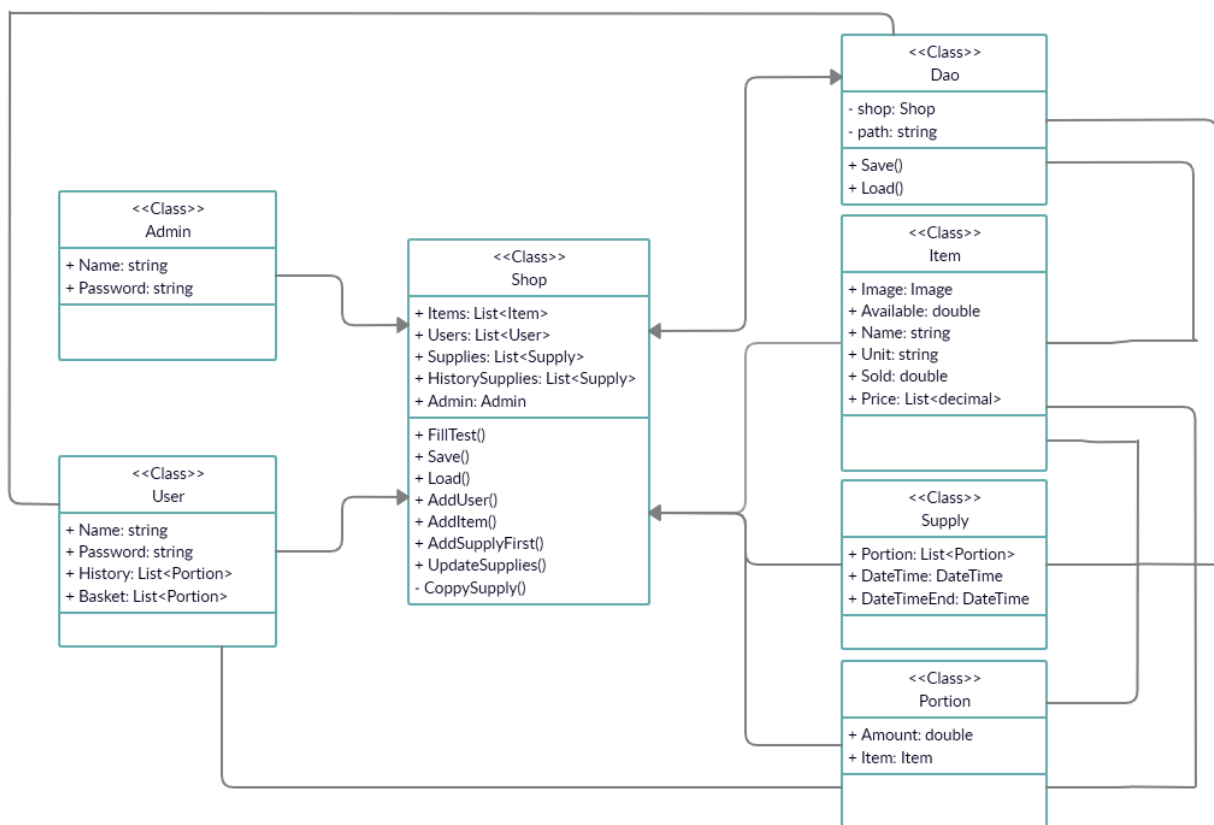


Рисунок 2.2 - Діаграма класів

Сховище і доступ до даних:

Всі дані, зазначені в роботі словом «магазин», а також «список користувачів» будуть представлені у вигляді бінарного файлу в каталозі проекту. Дані будуть переводитися з текстових в потік байтів за допомогою серіалізації з використанням вбудованої бібліотеки `BinaryFormatter`. Файл під назвою «shop.bin» розташований за адресою - «\bin\ Debug». Захист даних в рамках роботи не розглядається.

3 ІНСТРУКЦІЯ КОРИСТУВАЧА

3.1 Встановлення, запуск та головне вікно

Для користування програмою її необхідно завантажити на ваш комп'ютер. Після цього необхідно відкрити файл за шляхом \MyShop\LoginForm\bin\Debug\LoginForm.exe.

Після запуску програми відкривається вікно входу (рис. 3.1).

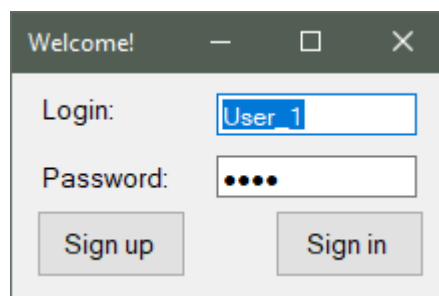


Рисунок 3.1 – Вікно входу в аккаунт

У ньому користувач входить у свій аккаунт або створює новий.

Бажаючи створити новий обліковий запис, користувач натискає “Sign up”.

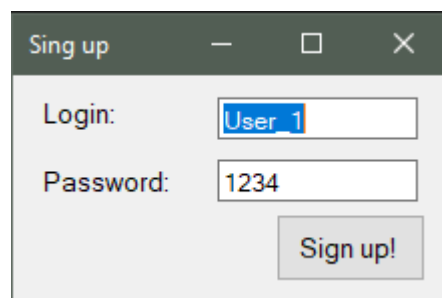


Рисунок 3.2 – Вікно реєстрації

Користувач вводить свої дані. Якщо дані введені правильно, користувача повертає до форми входу.

Під час реєстрації слід пам'ятати про наступне:

- Логін має містити хоча б 1 букву, та містити від 3 до 22 символів;
- Пароль має містити рівно 4 символи;
- Логіни в системі не можуть повторюватися.

3.2 Використання користувачами

Увійшовши до свого акаунта користувач потрапляє до головної сторінки (рис. 3.2.1).

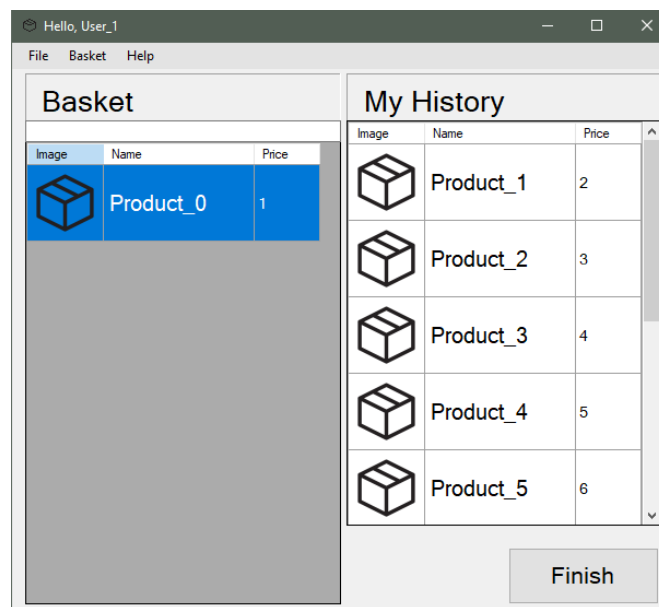


Рисунок 3.2.1 – Головне вікно користувача

3.2.1 Додавання продукту до корзини

У меню необхідно обрати “Basket”, згодом “Add”, після чого відкривається вікно додавання товару до корзини (рис. 3.2.2).

Обравши необхідний товар та його кількість, натискаймо кнопку «Save». Після чого знов відкритеся головне вікно (рис. 3.2.1), де з’явиться обраний товар.

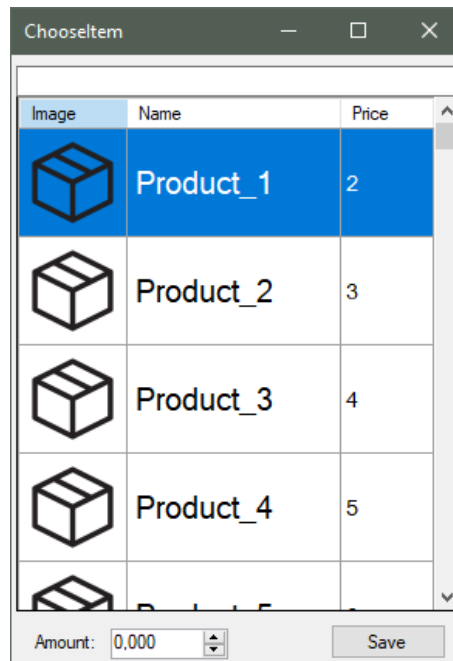


Рисунок 3.2.2 – Вибір продукту

3.2.2 Зміна продукту до корзини

За необхідністю є можливість змінити кількість придбання обраного товару, кликнувши на “Edit” в меню “Basket”. Після чого обрати бажану кількість.

3.2.3 Видалення продукту з корзини

Також можливо видалити товар, обравши “Delete” в меню “Basket”. Для попередження випадкового дотику необхідно буде підтвердити свій вибір.

3.2.4 Закінчення покупки

Щоб завершити процес покупки, необхідно обрати “Finish”, и після підтвердження покупець отримує список своїх купівель.

Щоб повернутися до форми авторизації(рис 3.1) необхідно натиснути кнопку «To sign in» в меню “File”.

Щоб вийти з програми, натисніть кнопку «To Desktop» в меню “File”.

3.3 Використання адміністратором

Увійшовши до свого аккаунта користувач потрапляє до адмінпанелі (рис. 3.3.1).

Items				Users		Supplies	
Name	Unit	Available	Image	Name	Password	Supply date	Count
Product_0	kg	9405		User_0	1234	26.05.2020 21:59	10
Product_1	l	9305		User_1	1234	25.05.2020 21:59	10
Product_2	kg	9205		User_2	1234	24.05.2020 21:59	10
Product_3	l	9105		User_3	1234	23.05.2020 21:59	10
Product_4	kg	9005		User_4	1234	22.05.2020 21:59	10
Product_5	l	8905		User_5	1234	21.05.2020 21:59	10
Product_6	kg	8805		User_6	1234	20.05.2020 21:59	10
Product_7	l	8705		User_7	1234	19.05.2020 21:59	10
Product_8	kg	8605				18.05.2020 21:59	10
Product_9	l	8505				17.05.2020 21:59	10
Product_10	kg	8405				16.05.2020 21:59	10
Product_11	l	8305				15.05.2020 21:59	10
Product_12	kg	8205				14.05.2020 21:59	10
Product_13	l	8105				13.05.2020 21:59	10
Product_14	kg	8005				12.05.2020 21:59	10
Product_15	l	7905				11.05.2020 21:59	10
Product_16	kg	7805				10.05.2020 21:59	10
Product_17	l	7705				09.05.2020 21:59	10
Product_18	kg	7605				08.05.2020 21:59	10

Recent Purchases		
Image	Name	Amount
	Product_0	1
	Product_1	2
	Product_2	3
	Product_3	4
	Product_4	5
	Product_5	6
	Product_6	7
	Product_7	8

Рисунок 3.3.1 – Адмінпанель

3.3.1 Додавання нового товару

Щоб додати новий товар до магазину необхідно натикнути кнопку “New” у меню “Item”. У вікні, що відкрилось, заповнити данні про товар: Назва, одиниці вимірювання, цину, зображення, та при необхідності доступна кількість. Після завершення заповнення натиснути кнопку “Save”

3.3.2 Змінна існуючого товару

Щоб виправити інформацію про товар необхідно натиснути кнопку “Edit” у меню “Item” або зробити подвійний клік на бажаний товар.

У вікні, що відкрилось, можливо виправити одиниці вимірювання, доступна кількість, зображення, та при умови, що товар не використовується у поточних поставках.

3.3.3 Видалення товару

Щоб видалити товар необхідно натиснути кнопку “Delete” у меню “Item”. Для попередження випадкового дотику необхідно буде підтвердити свій вибір. Видалити можливо лише той товар, що не використовується у поточних поставках.

3.3.4 Дізнатися історію зміни цін.

Щоб продивитися історію цін на обраний товар необхідно натиснути кнопку “Edit” у меню “Item” або зробити подвійний клік на бажаний товар, після чого обрати “Prices”. У вікні, що відкрилось (рис. 3.3.2), є можливість побачити історію цін.

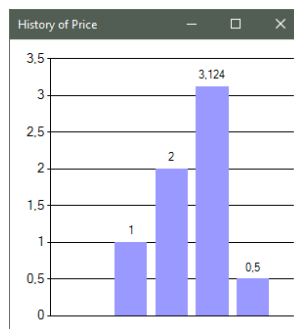


Рисунок 3.3.2 - Вікно історії цін

3.3.5 Створення нової поставки.

Щоб створити нову поставку необхідно натиснути кнопку “New” у меню “Supply”. З’явиться вікно обраних товарів(рис. 3.3.3) у поставці.

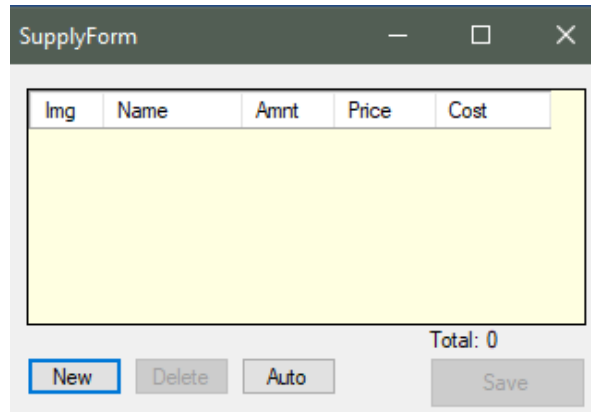


Рисунок 3.3.3 - Вікно поставки

Щоб додати товари до поставки необхідно обрати “New”, у вікні, що відкрилось (рис. 3.3.4), обрати бажаний товар та його кількість. Для полегшення створення поставки товари доступна кількість яких менша за 500 одиниць відмічений червоним, від 500 до 1000 одиниць – помаранчевим, інакше – зелений.

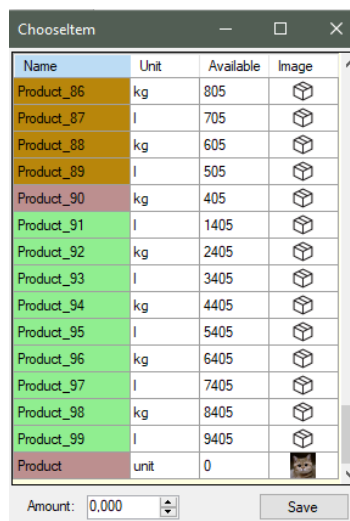


Рисунок 3.3.4 - Вікно додавання товару до поставки

Щоб видалити обраний товар с поставки необхідно обрати “Delete”, у вікні обраних товарів (рис. 3.3.3).

Присутнє авто-замовлення, що дозволяє автоматично додати до замовлення усі товари, присутня кількість котрих менше за 500 одиниць.

Для підтвердження поставки неохайно обрати “Save”, у вікні обраних товарів (рис. 3.3.3).

Створене замовлення додається до списку всіх поставок у адмінпанелі (рис. 3.3.1).

Ще не виконанні замовлення будуть відмічені червоним, виконані – зеленим.

3.3.6 Змінна існуючої поставки

Щоб виправити інформацію про поставку необхідно натикнути кнопку “Edit” у меню “Supply” або зробити подвійний клік на бажану поставку.

у вікні, що відкрилось, можливо додати/видалити товари у поставці.

Виправити можливо лише ту поставку, що була створена не раніше двох годин тому.

3.3.7 Видалення поставки

Щоб видалити поставку необхідно натикнути кнопку “Delete” у меню “Supply”.

Для попередження випадкового дотику необхідно буде підтвердити свій вибір.

Видалити можливо лише ту поставку, що була створена не раніше двох годин тому.

3.3.8 Створення звіту

Для створення звіту необхідно натиснути кнопку “Report” у адмінпанелі (рис.3.3.1).

Підтвердивши шлях створення файлу, звіт буде створений під назвою “MyShop.txt” у заданому каталозі.

3.3.9 Закінчення сеансу

Щоб повернутися до форми авторизації(рис 3.1) необхідно натиснути кнопку «To sign in» в меню “File”.

Щоб вийти з програми, натисніть кнопку «To Desktop» в меню “File”.

Щоб зберегти корегування необхідно натиснути кнопку «Save» в меню “File”.

Щоб загрузити збережені данні магазину необхідно натиснути кнопку «Load» в меню “File”.

ВИСНОВКИ

У ході виконання курсової роботи були освоєні методи створення програмних систем на засадах об'єктно-орієнтованого програмування, набулися навички роботи з програмною системою, створення в'язків між об'єктами.

У результаті отримана програма під назвою «Магазин з одним продавцем», яка дозволяє користувачу замовляти товари. Адміністратор має можливість коректувати базу існуючих товарів, замовляти поставки товарів, інвентаризувати залишки.

Програма написана на мові програмування C# на платформі .NET Framework 4.7.2 з використанням технологій Windows Forms у середі розробки Visual Studio 2019.

Усі дані програми зберігаються локально у бінарному вигляді. Програма може бути поліпшена застосуванням справжньої бази даних, яка синхронізується з базою даних товарів.

Галузь застосування – підприємницька діяльність.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Полное руководство по C# 8 и .NET Core [Электронный ресурс] / Режим доступа до ресурсу: <https://metanit.com/sharp/tutorial/>.
2. Бондарев В. М. Объектно-ориентированное программирование на C# : [учеб. пособие] / В. М. Бондарев. – Харьков : СМИТ, 2009. – 224 с.
3. Фаронов В. В. Создание приложений с помощью C#: Руководство программиста. / В. В. Фаронов. – Москва: Эксмо, 2008.
4. Нортроп Тони, Уилдермьюс Шон, Райан Билл. Основы разработки приложений на платформе Microsoft .Net Framework. Учебный курс Microsoft / Пер. с англ. - М.: «Русская редакция», Санкт-Петербург: «Питер», 2007.
5. Бондарев В. М. Основы программирования : [учеб. пособие] / В. М. Бондарев. – Харьков : Фолио, 1994. – 116 с.

ДОДАТОК А

Методи класу Shop

```
// Метод, що створює тестові дані.
public void FillTest(int n)
{
    n = Math.Min(n, 100);
    Items.Clear();
    Users.Clear();
    Supplies.Clear();
    var defImg = new
Bitmap(Path.GetFullPath("item.png"));
    for (int i = 0; i < n; i++)
    {
        Items.Add(new Item($"Product_{i}", i % 2 == 0 ?
"1" : "kg", i + 1, 10000, defImg));
        Users.Add(new User($"User_{i}", "1234"));
    }
    for (int i = 0; i < n; i++)
    {
        var k = new List<Portion>(10);
        for (int j = 0; j < 10; j++)
        {
            k.Add(new Portion {Item = Items[(i + j) %
n], Amount = 1 + (i + j) % 100});
            Items[i % n].Sold += 1 + (i + j) % 100;
        }
        Users[i].History = k;
        Supplies.Add(new Supply(k, DateTime.Now -
(TimeSpan.FromDays(n - i))));
    }
    Supplies.Reverse();
    UpdateSupplies();
}

// Метод, що додає нового користувача.
public void AddUser(User user) =>
    Users.Add(user);

// Метод, що додає новий товар.
public void AddItem(Item item) =>
    Items.Add(item);
```

```
// Метод, що додає нову поставку.
public void AddSupplyFirst(Supply supply) =>
    Supplies.Insert(0, supply);

// Метод, що оновлює дані про виконані поставки.
public void UpdateSupplies()
{
    for (int i = 0; i < Supplies.Count; i++)
    {
        if (Supplies[i].DateTimeEnd <= DateTime.Now) {
            HistorySupplies.Add(CopySupply(Supplies[i]));
            foreach (var portion in
                Supplies[i].Portions)
            {
                portion.Item.Available +=
                    portion.Amount;
            }
            Supplies.RemoveAt(i);
            i--;
        }
    }
}

// Метод, що зберігає магазин у постійну пам'ять.
public void Save() =>
    new Dao(this).Save();

// Метод, що завантажує магазин з постійної пам'яті.
public void Load()
{
    new Dao(this).Load();
    UpdateSupplies();
}
```

ДОДАТОК Б

Методи класу Dao

```
public void Save()
{
    using (Stream stream = File.Create(path))
    {
        var serializer = new BinaryFormatter();
        serializer.Serialize(stream, shop);
    }
}

public void Load()
{
    using (Stream stream = File.OpenRead(path))
    {
        var serializer = new BinaryFormatter();
        Shop st = (Shop)serializer.Deserialize(stream);
        Copy(st.Items, shop.Items);
        Copy(st.Users, shop.Users);
        Copy(st.Supplies, shop.Supplies);
        Copy(st.HistorySupplies, shop.HistorySupplies);
    }
    void Copy<T>(List<T> from, List<T> to)
    {
        to.Clear();
        to.AddRange(from);
    }
}
```

<https://github.com/RuslanProgrammer/MyShop>