НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра автоматизації та управління в технічних системах

АІС «Аптека “Medical Shop”»

Курсова робота

З дисципліни «Об’єктно-орієнтоване програмування»

|  |  |  |  |
| --- | --- | --- | --- |
| Керівник  Хмелюк В.С. |  | | Виконавець  Цитовцева А.С.  зал.книжка № ІТ-5130 |
| «Допущений до захисту»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Особистий підпис керівника) |  | | гр. ІТ-51  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Особистий підпис виконавця) |
| « »\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017р. |  | | « »\_\_\_\_\_\_\_\_\_\_\_\_\_\_2017р. |
| Захищений з оцінкою  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оцінка)  Члени комісії:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Особистий підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Особистий підпис) |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Розшифровка підпису)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Розшифровка підпису) | |

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КПІ»

КАФЕДРА АВТОМАТИКИ ТА УПРАВЛІННЯ В ТЕХНІЧНИХСИСТЕМАХ

Дисципліна «Об’єктно-орієнтоване програмування - 2»

Курс - 2 Група ІТ-51 Семестр - 4

ЗАВДАННЯ

на курсову роботу студентки

Цитовцевої Анни Сергіївни

1.Тема роботи*: АІС «Аптека ”Medical Shop”»*

2.Строк здачі студентом закінченої роботи *23.05.2017 р.*

3. Вихідні дані до роботи: *цільова платформа та платформа розробника – Microsoft.NET Framework 4.5; операційна система – Windows 7; мова програмування - С#; середовище програмування – Visual Studio 2015; забезпечити можливість менеджменту лікарськими засобами, та менеджент процесу клієнт-замовлення. Розробка функціоналу збереження інформації про: лікарський засіб, інгредієнт, тип лікарського засобу, пацієнта, лікаря, замовлення.*

4. Зміст пояснювальної записки (перелік питань, які підлягають розробці)

*Вступ. 1. Постановка задачі. 2. Структура програми. 3. Огляд існуючих рішень. 4. Перелік та призначення режимів і опис діалогів. 5. Структура даних і ресурсів програми. 6. Опис програми. 7. Інструкція програміста. 8. Керівництво користувача. Висновки. Список використаних джерел Додатки: Схема класів додатку. Значимий код додатку. Скріпти міграції бази даних*

5. Перелік графічного матеріалу *Діаграма класів. Зображення основних форм додатку.*

6. Дата видачі завдання *28.02.2017 р.*

КАЛЕНДАРНИЙ ПЛАН

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів курсової роботи | Термін виконання етапів роботи | Підписи керівника, студента |
| 1. | Отримання та узгодження теми курсової роботи | *До 25.03.2017* |  |
| 2. | Огляд існуючих рішень з тематики роботи | *До 27.03.2017* |  |
| 3. | Розробка структури програми та моделі даних | *До 03.04.2017* |  |
| 4. | Розробка діалогового інтерфейсу програми | *До 19.04.2017* |  |
| 5. | Кодування програми | *До 25.04.2017* |  |
| 6. | Тестування програми | *До 04.05.2017* |  |
| 7. | Оформлення пояснювальної записки | *До 12.05.2017* |  |
| 8. | Подання роботи на захист | *До 28.05.2017* |  |
| 9. |  |  |  |
| 10. |  |  |  |
| 11. |  |  |  |
| 12. |  |  |  |
| 13. |  |  |  |
| 14. |  |  |  |
| 15. |  |  |  |
| 16. |  |  |  |
| 17. |  |  |  |

Студент Цитовцева А.С.\_\_\_\_

(підпис)

(прізвище, ім’я, по батькові)

Керівник Хмелюк В.С.

(підпис) (прізвище, ім’я, по батькові)

«\_23\_» \_\_\_\_\_\_травня \_\_\_\_\_\_\_2017 р.

АНОТАЦІЯ

У курсовій роботі розглядається питання реалізації зручної программи , яка надає можливість обліку лікарських засобів аптеки. Задачею програми є регулювання бізнес процесом «Клієнт-Замовлення». Назва програми “Medical Shop”.

При створенні програми були використані основні принципи ООП: поліморфізм, інкапсуляція та наслідування. Алгоритм та інтерфейс додатку реалізовані на мові програмування високого рівня C# в середовищі Microsoft Visual Studio 2015. Пояснювальна записка виконана у розмірі 37 сторінок, 18 рисунків та 3 додатки.

ЗМІСТ

[ВСТУП 7](#_Toc483563948)

[1 ПОСТАНОВКА ЗАДАЧІ 8](#_Toc483563949)

[1.1Теоретична частина 8](#_Toc483563950)

[1.2 Практична частина 9](#_Toc483563951)

[2 МАТЕМАТИЧНА МОДЕЛЬ ТА СТРУКТУРА ПРОГРАМИ 10](#_Toc483563952)

[2.1 Огляд та аналіз існуючих рішень 10](#_Toc483563953)

[2.2 Порівняльна характеристика аналогу 12](#_Toc483563954)

[2.3 Методи розв’язання задачі 13](#_Toc483563955)

[3 ПЕРЕЛІК І ПРИЗНАЧЕННЯ ГРАФІЧНИХ ІНТЕРФЕЙСІВ 15](#_Toc483563956)

[3.1 Режими роботи програми 15](#_Toc483563957)

[3.2 Структура діалогу та меню користувача 16](#_Toc483563958)

[4 СТРУКТУРА ДАНИХ ТА РЕСУРСІВ ПРОГРАМИ 17](#_Toc483563959)

[4.1 Класи програми 17](#_Toc483563960)

[4.2 Ресурси 19](#_Toc483563961)

[5 ОПИС ПРОГРАМИ 21](#_Toc483563962)

[6 ІНСТРУКЦІЯ АДМІНІСТРАТОРА 22](#_Toc483563963)

[6.1 Загальні відомості 22](#_Toc483563964)

[6.2 Вимоги до інсталяції 22](#_Toc483563965)

[7 КЕРІВНИЦТВО КОРИСТУВАЧА 25](#_Toc483563966)

[7.1 Загальні відомості 25](#_Toc483563967)

[7.2 Послідовність дій користування програмою 25](#_Toc483563968)

[ВИСНОВКИ 30](#_Toc483563969)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 31](#_Toc483563970)

[ДОДАТОК А. Текст програми 32](#_Toc483563971)

[ДОДАТОК Б. Діаграма класів 37](#_Toc483563973)

[ДОДАТОК В. Діаграма наслідування 38](#_Toc483563975)

# ВСТУП

Темою курсової роботи є програма, яка надає можливість менеджменту лікарськими засобами аптеки, створення нових, редагування та видалення існуючих. Крім того є можливість здійснювати менеджемент бізнес процесом «Клієнт-Замовлення».

Метою курсової роботи є створення автоматизованої інформаційної системи, яка дає змогу зберігати інформацію про лікарські засоби аптеки, про клієнтів аптеки, про замовлення клієнтів та місцезнаходження працюючих аптек. Користувач має змогу додавати засоби до кошику, продивлятись загальну вартість з урахуванням дисконту та слідкувати за статусом замовлення. Додаток дозволяє менеджеру аптеки витрачати менше часу на пошук та підбір найкращого лікарського засобу для клієнта, та можливість своєчасно повідомляти клієнта, що його замовлення виконано.

Окрім цього в даній програмі містяться данні, які допоможуть не тільки зручно підібрати потрібну аптеку, де міститься даний препарат, але і вкаже ціну за якою продається даний лікарський засіб. До того ж дана програма вкаже чи міститься знижка на данні ліки, що є безумовно зручним для пересічного покупця.

Зовнішній інтерфейс користувача розроблений у сучасному вигляді та є ергономічним. Інтерфейс спроектований згідно правил проектування інтерфейсу, крім цього були врахованні особливості фармацевтичної галузі. Користуватися додатком досить просто, для цього не потрібні додаткові навички. Додаток вміє реагувати на непердбачувані ситуації та опрацьовувати помилки, при цьому не припиняючи свою роботу, і тим самим не втрачаючи дані, крім цього користувачеві повідомляється що потрібно зробити, щоб досягти бажаного результату.

Додаток реалізує всі функції які потрібні менеджеру та користувачу без зайвих, надлишкових надбудов, через що працювати з додатком досить легко.

# 1 ПОСТАНОВКА ЗАДАЧІ

* 1. Теоретична частина

Мета курсової роботи – розроблення додатку, який би задовольняв потреби менеджера аптеки у змозі легко бачити доступні лікарські засоби, замовлення клієнтів, контактні дані клієнтів та функцію контролю замовлень.

Додаток повинен у собі містити:

* Привабливий, зручний та легкий у використанні інтерфейс;
* Можливість створення та редагування записів лікарських засобів;
* Можливість створення записів клієнтів
* Можливість зберігати поточний стан програми, для легкого відновлення даних в разі необачного закриття програми
* Можливість перегляд файлу інструкції за допомогою інших програм;
* Можливість перегляду адреси діючих аптек на карті за допомогою браузера та інтернету;
* Можливість створення нового замовлення
* Можливість перегляду списку замовлень клієнтів
* Можливість видалення записів лікарських засобів та клієнтів
* Відстеження готовності замовлення клієнта
* Зручність при переході між вікнами, додаток повинен бути розроблений з дотриманням правил проектування інтерфейсу
* Високий рівень usability
* Вікно «About», де повинна містить інформація про додаток, та розробника
* Зручний граф переходу між вікнами

1.2 Практична частина

В даній курсовій роботі потрібно навчитись працювати створювати якісні, user-friendly додатки, що повністю виконують свої основні функції. В додатку планується використати наступні ідеї та підходи до роботи:

* Використання технології Windows Forms для розробки UI частини додатку;
* Використання моделей даних та створення зв’язків між ними;
* Розподілення програми на слої, для можливості розподіленої розробки додатку.
* XML серілізація даних у файл, та десеріалізація.
* Розробка інтерфейсу користувача згідно з правил проектування інтерфейсу

# 2 МАТЕМАТИЧНА МОДЕЛЬ ТА СТРУКТУРА ПРОГРАМИ

2.1 Огляд та аналіз існуючих рішень

Було проведено огляд та аналіз існуючих подібних додатків, але варто зазначити, що такі завдання вирішуються у великих системах для ведення бізнесу, наприклад «1С», але недоліками таких програм є їх складність, надлишковість (Рисунок 2.1). Вони, як правило, досить дорогі (Рисунок 2.4), і пропонують універсальні засоби вирішення заданих проблем, незалежно від сфери зайнятості користувача.

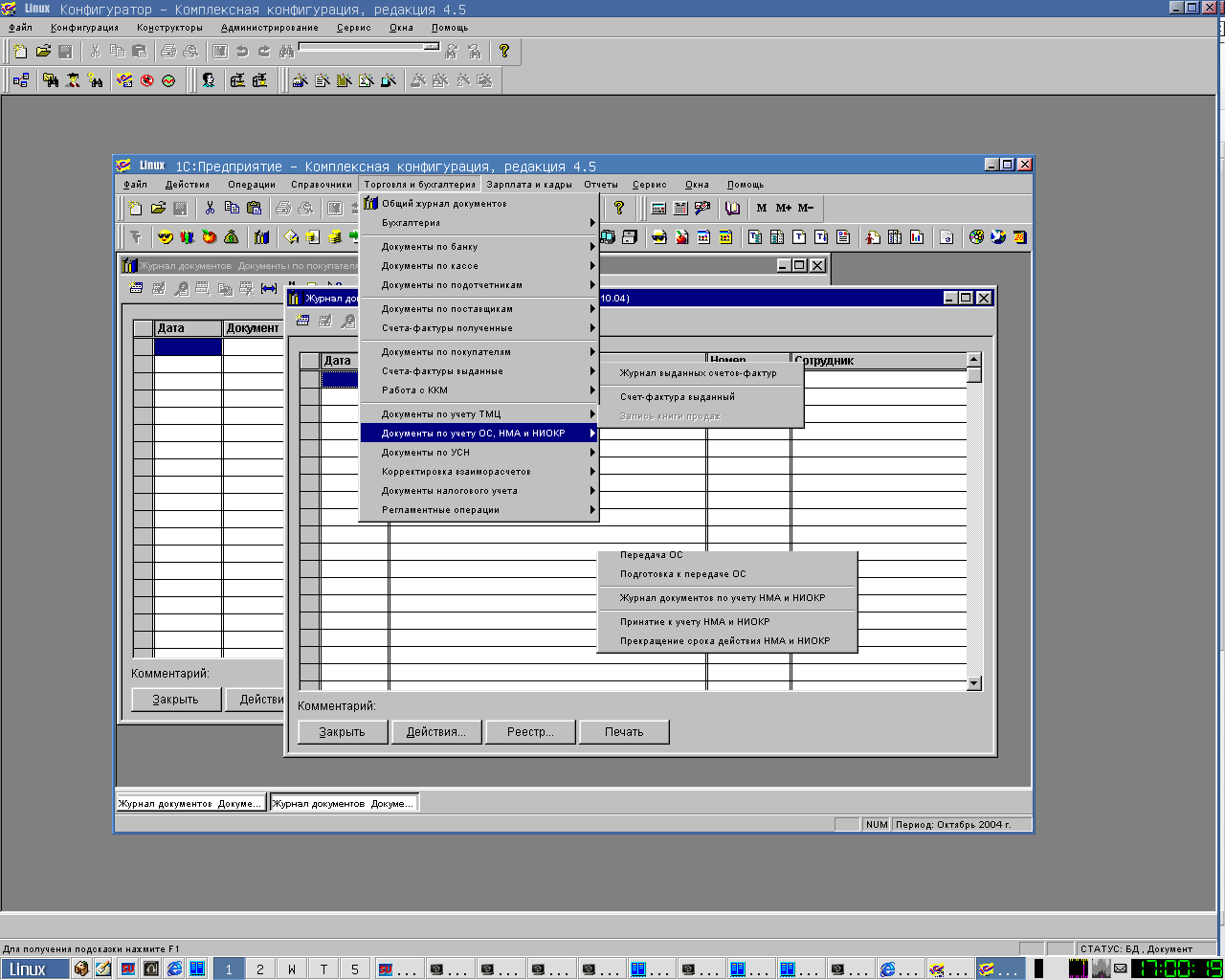


Рисунок 2.1 – Додаток «1С підприємство»

Провевши детальний аналіз ринку я дізналась, шо популярним аналогом є додаток «Аптека Моріон» (Рисунок 2.2), розробники якої надають можливість використовувати демо-версію 30 днів.

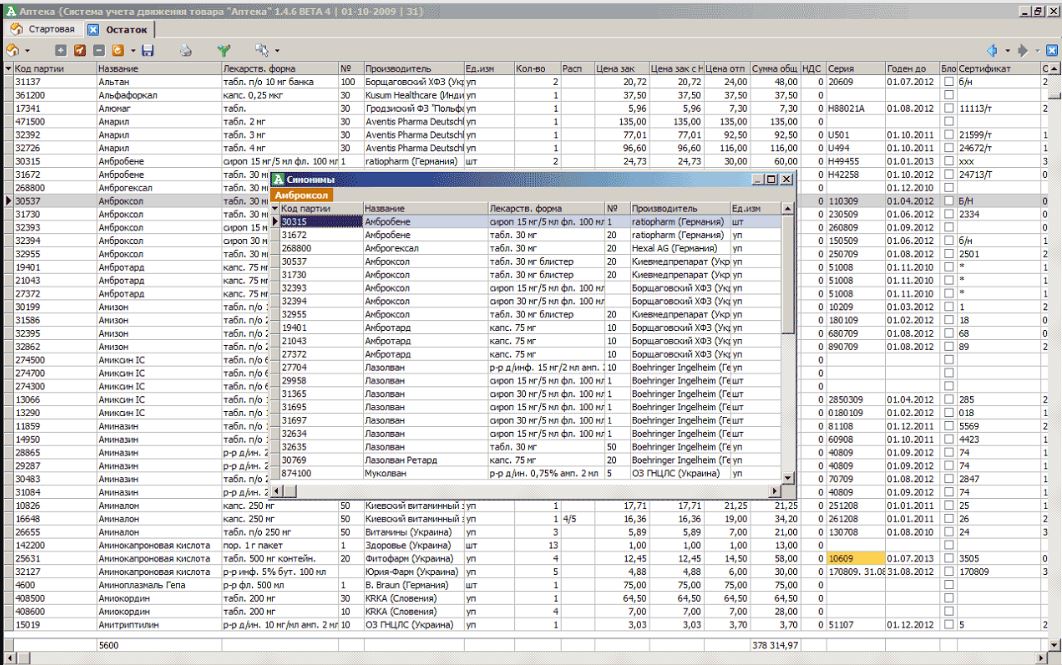


Рисунок 2.2 – Додаток «Аптека Моріон»

Деякі аптеки надають переваги додатоку «Парацельс» (Рисунок 2.3).

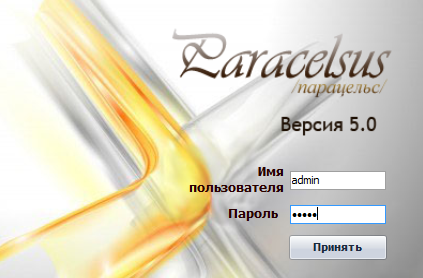


Рисунок 2.3 – Додаток «Парацельс»

Але, на мій погляд, їх недоліком є застарілий інтерфес, складність та надлишковість.



Рисунок 2.4 – Ціна на додаток «1С підприємство»

2.2 Порівняльна характеристика аналогу

На основі існуючого подібного додатку 1С було проведено характеристичний аналіз, який явно демонструє переваги і недоліки існуючих рішень (див. Таблиця 2.1). Основний недолік існуючих програм, це те, що вони намагаються узагальнити всі сфери, в яких може застосовуватись бухгалтерський облік. Крім цього, метою існуючих подібних додатків було створити один універсальний додаток. Це є недоліком, тому що в різних сферах бізнесу є свої нюанси, і програма повинна це враховувати. В розробці додатку «Medical Shop» було враховано всі недоліки існуючих програм.

Майбутній додаток має бути максимально простим та зрозумілим користувачу, оскільки це основний інструмент, що допомагає виконувати його роботу. Додаток має вирішувати поставлені питання.

Оскільки основні дії користувача у цій програмі – створення замовлень за рецептом, пошук лікарського засобу, відстеження замовлення, менеджмент клієнтів, тому було вирішено зробити саме ці дії найдоступнішими та найзручнішими.

Для зручності та привабливості додатку, було вирішено застосувати всі правила проектування інтерфейсу користувача, що дозволить зробити додаток зручним в користуванні, та зменшити час «привикання» до інтерфейсу. Метою розробленого інтерфейсу було зробити найголовніші дії найлегшими у виконанні та легкими для запам’ятовування.

Було проведено кілька інтерв’ю з потенційними користувачами, і на основі їх результатів було створено математичну модель, та поставлено вимоги до програми.

Таблиця 2.1 – Порівняльна характеристика програми 1С

|  |  |
| --- | --- |
| Переваги | Недоліки |
| Наявність готових налаштувань | Неможливість ведення обліку в динаміці |
| За невеликий період часу можливе отримання потрібних документів, з використанням певних форм | Відносно не найвища швидкість проведення обрахунків |
| Відсутність потреби у високій кваліфікації спеціалістів | Для певної діяльності використовують певну версію, тобто програма «1С» не являється універсальною. |
| Отримання програми в певний момент, яка відповідатиме максимально на завдання, які були висунуті. | Складність та надлишковість |

## 2.3 Методи розв’язання задачі

Методом розв’язання моєї задачі я обрала об’єктно-орієнтоване програмування. На відміну від традиційних поглядів, коли програму розглядали як набір підпрограм, або як перелік інструкцій комп'ютеру, ООП програми можна вважати сукупністю об'єктів. Відповідно до парадигми об'єктно-орієнтованого програмування, кожний об'єкт здатний отримувати повідомлення, обробляти дані, та надсилати повідомлення іншим об'єктам. Кожен об'єкт — своєрідний незалежний автомат з окремим призначенням та відповідальністю.

Об'єктно-орієнтоване програмування - це метод програмування, заснований на поданні програми у вигляді сукупності взаємодіючих об'єктів, кожен з яких є екземпляром певного класу, а класи є членами певної ієрархії наслідування. Я спочатку створив клас, а на його основі при виконанні програми створив конкретні об'єкти (екземплярів класів). На основі класів можна створювати нові, які розширюють базовий клас і таким чином створюється ієрархія класів.

# 

# 3 ПЕРЕЛІК І ПРИЗНАЧЕННЯ ГРАФІЧНИХ ІНТЕРФЕЙСІВ

3.1 Режими роботи програми

Після того, як була запущена програма, відкривається форма авторизації Login, звідки можна перейти до основної форми Main та до форми реєстрації нового користувача Register. Програма може працювати у двох режимах : режим адміністратора та режим користувача.

Можливості адміністратора :

* Вести облік лікарских засобів, а саме додавати, редагувати та видаляти існуючі препарати та забезпечувати детальну інформацію про них, прикріплювати інструкцію у PDF файлі.
* Представляти інформацію про працюючі аптеки, де клієт зможе забрати своє замовлення; вказати місцезнаходження аптеки на карті.
* Має доступ до всіх облікових записів користувачів.
* Має перелік поточних замовлень та може змінювати їх статус; здійснені замовлення він може видаляти зі свого списку.
* Переглянути причини відмов від замовлень
* Встановлювати знижку обраним клієнтам.

Можливості користувача :

* Редагувати інформацію свого акаунта.
* Додати товар у вибраній кількості до кошика.
* Переглянути кошик та загальну вартість чеку з урахуванням знижки.
* Очистити кошик
* Сформувати замовлення з кошику, при цьому вказавши зручну аптеку для отримання.
* Переглянути статус замовлення.
* Скасувати замовлення, вказавши причину відмови.

## 3.2 Структура діалогу та меню користувача

Інтерфейс розробленої програми створений у зручному та простому для користувача стилі. Меню користувачів має такі пункти:

* Акаунти/Профіль – відображає інформацію про користувачів або власний профіль;
* Замовлення – перехід до вікна, яке відображає записи замовлень;
* Товар – перехід до вікна, яке відображає список товарів, та можливість редагування для адміна;
* Аптеки – перехід до вікна, яке відображає список аптек, та можливість редагування для адміна;
* Кошик – перехід до вікна з товарами, які хочу замовити клієнт;
* Про програму – перехід до вікна з додатковою інформацією про програму та розробника.

Схема переходів між формами подана на рисунку 3.1:

Copy of forms

Рисунок 3.1 - Схема переходів між GUI-формами

# 4 СТРУКТУРА ДАНИХ ТА РЕСУРСІВ ПРОГРАМИ

## 4.1 Класи програми

Клас Base є батьківським класом для 5-ти класів. Він є загальним класом. Вміст класу:

Поля:

* Id : Guid – Унікальний ідентифікатор класу;
* Name : string – Імя класу;

Функції:

* Load() : return bool – Загрузка усіх класів з цим темплейтом із бази даних;
* Save() : void – Збереження усіх елементів з однаковим тесплейтом в базу даних;
* ToString() : return string – Перевантаження стандартної функції, для виведення поля Name;

Клас Client описує об’єкти клієнтів. Вміст класу:

Поля:

* Phone: String – Номер телефону клієнта;
* Discount : int – Знижка клієнта;
* IsAdmin : bool – Повертає значення, чи цей користувач адміністратор;
* Login : string – Логін клієнта;
* Password : string – Пароль клієнта, зберігається у закодованому форматі, який неможливо розкодувати;
* Orders : List<Order> – Список замовлень клієнта;

Функції:

* ToString() – Перевантаження базового класу, додається поле Altitude;

Клас Medicin описує об’єкти ліків. Вміст класу:

Поля:

* Producer : string – Виробник;
* UnitActiveSubstance : string – Одиниця виміру діючої речовини;
* CountOfActiveSubstance : int - Кількість діючої речовини;
* Packing : int - Кількість в упаковці;
* NeedRecipe : bool – Рецепт;
* Price : decimal – Ціна;
* Appointment : string - Призначення (короткий опис призначення);
* Instruction : string - Інструкція (посилання на файл з інструкцією);

Функції:

* ToString() – Перевантаження базового класу, додається поле Altitude;

Клас Purchase є розв’язним для класів Order та Medicin та містить їх Guid. Вміст класу:

Поля:

* Order : Order – Посилання на замовлення;
* Medicin : Medicin – Посилання на ліки;
* Count : int - Кількість упаковок;
* Price : decimal - Ціна;

Клас Pharmacy описує об’єкти аптеки. Вміст класу:

Поля:

* Address : string – Адреса аптеки;
* Description : string - Опис аптеки;
* MapUri : string – Посилання на карту;

Клас Order описує об’єкти замовлень. Містить у собі клас стану замовлення. Вміст класу:

Поля:

* OrderId : int – Ід замовлення;
* SecondId : int – Наступний Ід який не використовується;
* Client : Client – Посилання на клієнта;
* Pharmacy : Pharmacy – Посилання на аптеку;
* CreateTime : DateTime - Дата створення;
* CloseOrderTime : DateTime - Дата закриття замовлення;
* Status : OderStatus – Статус замовлення;
* Discount : int – Знижка
* Price : decimal - Сума заказу без знижки
* ReasonRejection : string - Причина відхилення замовлення
* NeedRecipe : bool - Потрібен рецепт для отримання
* Medicines : Dictionary<Medicin, int> -

Функції:

* ChangeStatus (OderStatus status) : void – Функція зміни статусу замовлення;

До приватних полів класів компілятор звертається через властивість.

Для відображення усіх зв’язків між класами та їх ієрархії у роботі присутня діаграма об’єктної моделі та наслідування (додатки Б та В відповідно).

## 4.2 Ресурси

Розроблена програма використовує такі зовнішні файли(папки):

- Microsoft.CodeDom.Providers.DotNetCompilerPlatform.1.0.1 - пакет налаштування Resharper під платформу C# 6.0;

- Client.xml - файл серiалiзацiї клієнтів в xml ;

- Medicin.xml - файл серiалiзацiї ліків в xml ;

- Order.xml - файл серiалiзацiї замовлень в xml ;

- Pharmacy.xml – файл серiалiзацiї аптек в xml ;

- Purchase.xml – файл серіалізації проміжного класу в xml ;

- Instructions – папка для збереження інструкцій ;

# 5 ОПИС ПРОГРАМИ

Програма магазин ліків розроблена дня фармацевтів в аптеці для більш швидкого та зручного використання. Дана програма легка в використанні тому будь-який працівник швидко і без зусиль навчиться використовувати дану програму.

Дана програма є досить легка для використання. Вона містить необхідні функції, які допоможуть вам без проблем користуватися саме цією програмою.

Програмний комплекс містить такі функції:

* додавання, редагування та видалення об’єктів ліків, клієнтів, аптек, замовлень;
* можливість придбати та замовити ліки у потрібну аптеку
* можливість використовувати інструкції до ліків з звязком з іншими програмами
* збереження та завантаження інформації у файл та з файлів, для подальшого використання.

Крім того вона дозволить швидко виводити потрібну інформацію на екран і бачити даний товар, на складі, і скільки товару є в наявності. В ході використання цієї програми можна вивести опис потрібного препарату, що дозволить фармацевту порекомендувати даний препарат.

Вхідні данні вводяться користувачем через клавіатуру та за допомогою миші.

Вихідні данні виводяться у listbox-ах, а також у listView-ах,

comboBox-ах, richtextBox-ві на відповідних вкладках.

# 6 ІНСТРУКЦІЯ АДМІНІСТРАТОРА

6.1 Загальні відомості

Створена програма називається «Medical Shop». Мова програмування – C#. Графічний інтерфейс побудований з використанням технології Windows Forms. Створювалася програма в IDE MS Visual Studio 2015. Дана програма не використовує сторонніх зовнішніх бібліотек або драйверів. Під час роботи автоматично створюються папка Saves, де існують від 1 до 5 \*.xml файлів та папка Instructions для збереження усіх інструкцій.

Мінімальні апаратні вимоги:

* процесор Pentium II 850 МГц;
* пам'ять 256 МБ ОЗП;
* мінімум 1.0 MБ пам’яті на жорсткому диску;
* VGA-монітор з роздільною здатністю 800x600 пікселів або вище;
* операційна система Windows 7 (або новіша версія Windows) з встановленим Microsoft. NET Framework 4.5

6.2 Вимоги до інсталяції

Запускаємо файл Setup.exe з CD диску, та бачимо вікно інсталятора програми (рисунок 6.1).



Рисунок 6.1 – Головне вікно інсталятора програми

Після цього натискаемо на кнопку “Дальше” та бачимо вікно де вказуємо, куди ми будемо зберігати програму (рисунок 6.2).

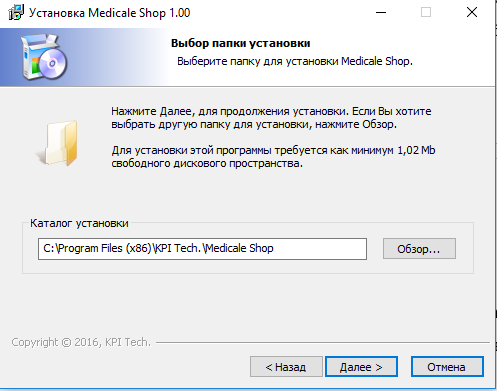


Рисунок 6.2 – Вікно зберігання програми

Пісня натиснення на кнопку “Дальше”, ми можемо вказати чи потрібен ярлик нашої програми на робочому столі (рисунок 6.3).

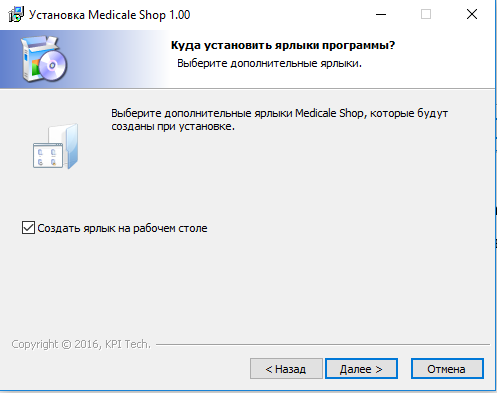


Рисунок 6.3 – Вікно з ярликом на робочому столі

Перед тим як встановити програму ми бачимо вікно з нашими параметрами, які ми встановлювали вище (рисунок 6.4).

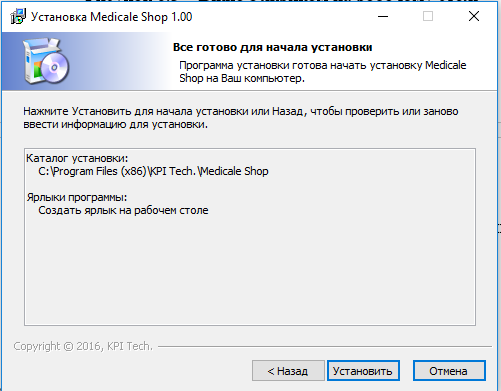


Рисунок 6.4 – Параметри установки

Після натисненя на кнопку “Установить”, ми бачино вікно завершення встановлення програми на комп’ютер (рисунок 6.5).

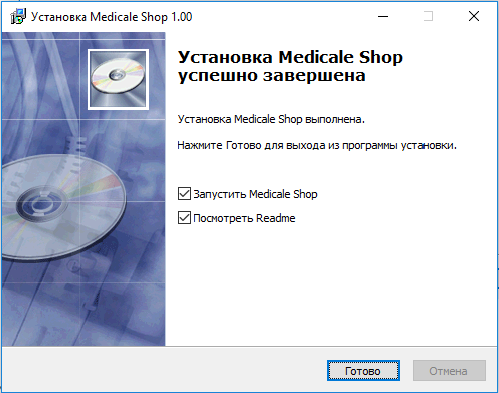


Рисунок 6.5 – Завершення установки

# 7 КЕРІВНИЦТВО КОРИСТУВАЧА

## 7.1 Загальні відомості

Створена програма називається «Medical Shop». Програма створена для спрощення роботи менеджерам аптек. Вона дозволяє зберігати, відтворювати та зручно маніпулювати екземплярами основних сутностей, необхідних у їх роботі. Також, програма дає можливість зручно шукати Лікарські засоби, відповідно до вимог клієнта.

7.2 Послідовність дій користування програмою

Додаток має кілька вікон, які забезпечують графічним інтерфейсом весь функціонал програми. При старті програми ми бачимо вiкно входу (рисунок 7.1).

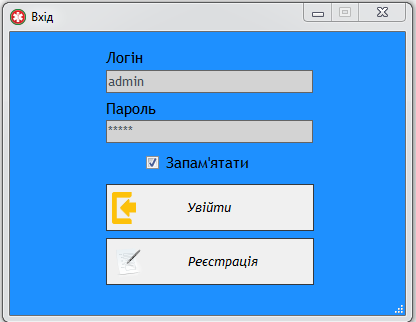


Рисунок 7.1 – Форма входу до аккаунту

Вводимо логін та пароль у відповідні поля і натискаємо кнопку “Увійти”. Якщо аккаунту немає, то натисність кнопку “Реєстація”, після цього з’явиться форма реєстрації ного аккаунту (рисунок 7.2).

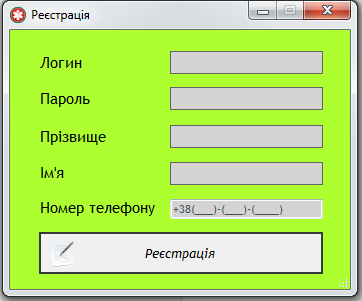


Рисунок 7.2 – Форма реєстрації нового аккаунту

Заповнивши відвопідні поля натискаємо кнопку “Реєстрація” і створюємо нового користувача.

Після входу бачимо основну форму програми (рисунок 7.3).

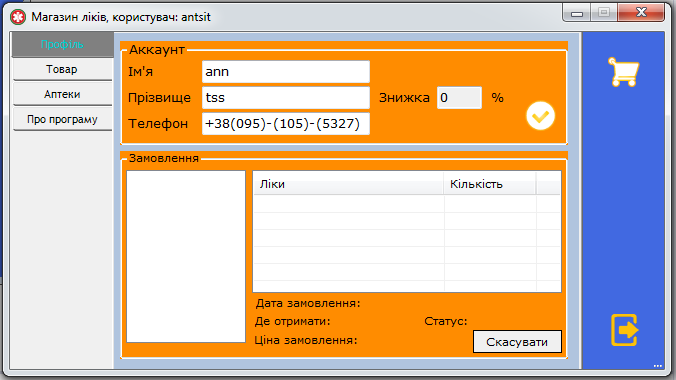


Рисунок 7.3 – Основна форма програми

На цій формі користувач може редагувати своє ім'я, прізвище та телефон. Також є можливість переглядати інформацію про всі свої замовлення, а також скасовувати замовлення.

Вкладка “Товар”, призначена для перегляду товару, який можливо замовити (рисунок 7.4).

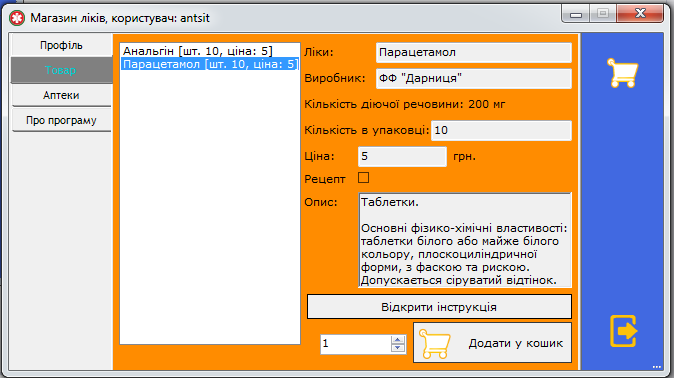


Рисунок 7.4 – Вкладка “Товар”

Користувач вибирає товар, вказує кількість і додає у кошик. Є можливість перегляду повної інструкції до нього.

Наступна вкладка “Аптеки”, на цій вкладці представлені усі аптеки, що є у программі (рисунок 7.5). Вибравши відповідну аптеку, користувач може ознайомитись з описом даної аптеки, а також переглянути її розміщення на мапі.

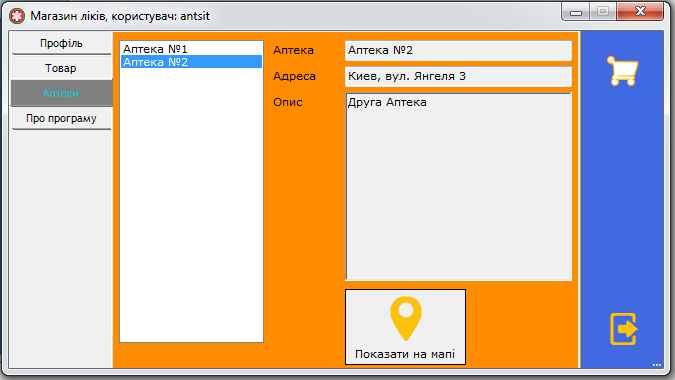


Рисунок 7.5 – Вкладка “Аптеки”

Остання вкладка - “Про програму”, на цій вкладці є опис самої програми (рисунок 7.6).

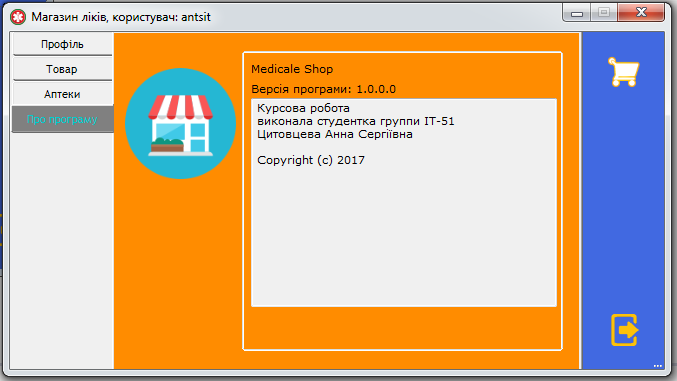


Рисунок 7.6 – Вкладка “Про програму”

На основній формі є кнопка “Кошик”, при натиснені на цю кнопку можна продивитися усі товари, які туди були додані, а також ціну замовлення зі знижкою та без знижки (рисунок 7.7).

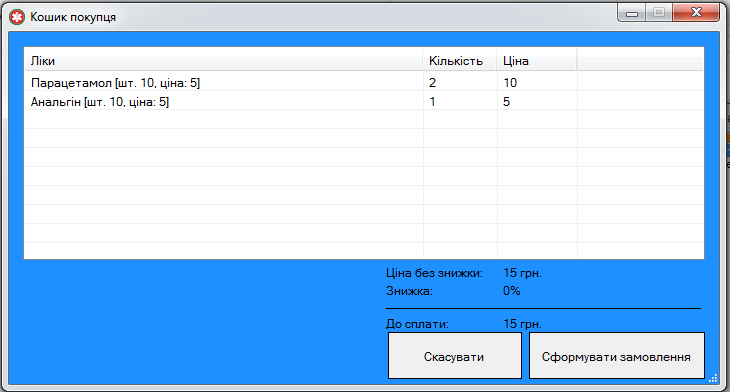


Рисунок 7.7 – Форма “Кошик”

На формі кошика можна як скасувати замовлення так і підтвердити, при натиснені на кнопку “Замовити”, відкриється форма для того щоб вибрати в яку аптеку доставити замовлення (рисунок 7.8).

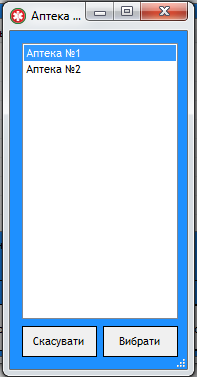


Рисунок 7.8 – Форма для обрання доставки

# ВИСНОВКИ

В ході виконання роботи були отримані теоретичні та практичні навички розроблення інформаційної системи.

Курсова робота допомогла розширити навички роботи у мові програмування C# та більше пізнати її структуру. Мною були здобуті навички проектування та створення класів даних, їх зв’язків та налаштування взаємодії прикладного коду з даними. Було поглиблено знання про побудову зручного інтерфейсу користувача, та проаналізовано різні підходи до його створення.

Були проведені експерименти на реакцію програми на введення некоректних вхідних даних, вивчено поведінку програми в таких умовах. В результаті цього мною були засвоєні навички знаходження місць в коді де можуть виникати помилки та перехоплення цих помилок і повідомлення про це користувача.

Відповідно до результатів дослідження, аналізу та розробки додатку, було створено програму, що дає змогу автоматизувати роботу аптеки, здійснювати менеджмент процесом Клієнт-Замовлення та Лікарським засобом.

Переваги додатку:

* Лаконічність та зручність інтерфейсу. Легкість та доступність у використанні користувачем;
* функціонал програми, який дає змогу виконувати будь-які потрібні дії.

Недоліки:

* додаток залежний від інтернету, якщо База Даних розміщенна Online;
* при великій кількості записів даних програму слід зробити асинхронною, щоб дані оброблялись в різних потоках;

Перспективи розвитку додатку:

* використання потоків для забезпечення швидкої роботи додатку;
* реалізація нових функцій для розвитку додатку та розширення;
* вдосконалення зовнішнього інтерфейсу додатку, використання новітніх розробок інтерфейсів;
* оптимізація інтерфейсу для роботи з великою кількістю даних;

Створений додаток відповідає всім вимогам, які висувалися для вирішення.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

* 1. Шилдт Г. C# 4.0 Полное руководство. [Текст] : Пер. С англ. – М.: ООО «И.Д.Вильямс», 2011. – 1056 с.
  2. Методичні вказівки до виконання та оформлення курсової роботи з дисципліни «Об’єктно-орієнтоване програмування» [Текст] / Уклад.: В.А.Жеребко, Н.М. Піщаєва. – К.: НТУУ «КПІ», 2010. – 70 с.
  3. Гради Буч, Роберт А. Максимчук, Майкл У. Энгл, Бобби Дж. Янг, Джим Коналлен, Келли А. Хьюстон. Объектно-ориентированный анализ и проектирование с примерами приложений. [Текст] : Пер. С англ. – М.: Вильямс, 2008.- 720с.
  4. Державний реєстр лікарських засобів України - [Електронний ресурс]. – Режим доступу: https://drlz.[com](http://www.drlz.com.ua/).ua
  5. Успадкування [Електронний ресурс]. – Режим доступу: <http://uk.wikipedia.org/wiki/Успадкування_(програмування)>
  6. Інкапсуляція [Електронний ресурс]. – Режим доступу: <http://uk.wikipedia.org/wiki/Приховування_інформації>
  7. Поліморфізм [Електронний ресурс]. – Режим доступу: <http://uk.wikipedia.org/wiki/Поліморфізм_(програмування)>
  8. Об’єктно-орієнтоване програмування [Електронний ресурс]. – Режим доступу: <https://uk.wikipedia.org/wiki/Об'єктно-орієнтоване_програмування>

# ДОДАТОК А

## Текст програми

Лістинг А.1. – Клас «Base» (файл Base.cs)

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization;

using System.Xml;

namespace MedicaleShop.Core

{

[DataContract]

public class Base<T> where T : Base<T>, IFileManager

{

public static Dictionary<Guid, T> Items = new Dictionary<Guid, T>();

public Base() : this("")

{

}

public Base(string name)

{

Id = Guid.NewGuid();

Name = name;

Items.Add(Id, (T) this);

}

public Base(string name, Guid key)

{

Id = key;

Name = name;

Items.Add(Id, (T) this);

}

public static List<T> Indexer => Items.Values.ToList();

[DataMember]

public Guid Id { get; set; }

[DataMember]

public string Name { get; set; }

public bool Load()

{

if (!Directory.Exists("Saves"))

return false;

var file = typeof (T).Name + ".xml";

var path = Directory.GetCurrentDirectory();

if (!File.Exists(path + @"\Saves\" + file))

return false;

using (var xmlReader = XmlReader.Create(path + @"\Saves\" + file))

{

var dataContractSerializer = new DataContractSerializer(typeof (Dictionary<Guid, T>));

Items = dataContractSerializer.ReadObject(xmlReader) as Dictionary<Guid, T>;

}

return true;

}

public void Save()

{

var file = typeof (T).Name + ".xml";

var settings = new XmlWriterSettings {Indent = true};

var path = Directory.GetCurrentDirectory();

if (!Directory.Exists(path + @"\" + "Saves"))

Directory.CreateDirectory(path + @"\" + "Saves");

path += @"\" + "Saves";

using (var xmlWriter = XmlWriter.Create(path + @"\" + file, settings))

{

var dataContractSerializer = new DataContractSerializer(typeof (Dictionary<Guid, T>));

dataContractSerializer.WriteObject(xmlWriter, Items);

xmlWriter.Close();

}

}

public override string ToString()

{

return Name;

}

}

}

Лістинг А.2. – Клас «Client» (файл Client.cs)

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

namespace MedicaleShop.Core.Classes

{

// Клиенты

[DataContract]

public class Client : Base<Client>, IFileManager

{

public Client(string name, string phone, int discount = 0) : base(name)

{

Phote = phone;

Discount = discount;

}

public Client()

{

}

[DataMember]

public string Phote { get; set; } // Номер телефону

[DataMember]

public int Discount { get; set; } // Знижка

[DataMember]

public bool IsAdmin { get; set; }

[DataMember]

public string Login { get; set; }

[DataMember]

public string Password { get; set; }

public List<Order> Orders => Order.Items.Values.Where(value => value.Client == this).ToList();

public override string ToString()

{

return base.ToString() + $" ({Login}), тел. " + Phote + ", " +

(Discount > 0 ? $"знижка - {Discount}%" : "знижки немає");

}

}

}

Лістинг А.3. – Клас «Medicin» (файл Medicin.cs)

using System.Runtime.Serialization;

namespace MedicaleShop.Core.Classes

{

// Медикаменти

[DataContract]

public class Medicin : Base<Medicin>, IFileManager

{

public Medicin(string name) : base(name)

{

}

public Medicin()

{

}

[DataMember]

public string Producer { get; set; } // Виробник

[DataMember]

public string UnitActiveSubstance { get; set; } // Одиниця виміру діючої речовини

[DataMember]

public int CountOfActiveSubstance { get; set; } // Кількість діючої речовини

[DataMember]

public int Packing { get; set; } // Кількість в упаковці

[DataMember]

public bool NeedRecipe { get; set; } // Рецепт

[DataMember]

public decimal Price { get; set; } // Ціна

[DataMember]

public string Appointment { get; set; } // Призначення (короткий опис призначення)

[DataMember]

public string Instruction { get; set; } // Інструкція (посилання на файл з інструкцією)

public override string ToString()

{

return base.ToString() + " [шт. " + Packing + ", ціна: " + Price + "]";

}

}

}

Лістинг А.4. – Клас «Order» (файл Order.cs)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

namespace MedicaleShop.Core.Classes

{

[DataContract]

public enum OderStatus

{

[EnumMember] Formed, // Сформований

[EnumMember] Confirmed, // підтверджений

[EnumMember] ToProvide, // в забезпеченні

[EnumMember] InGetPoint, // в точці отримання

[EnumMember] Received, // отриманий

[EnumMember] Deflected // відхилений

}

// Замовлення

[DataContract]

public class Order : Base<Order>, IFileManager

{

[DataMember] private Guid? \_clientId; // ІД Клієнта

[DataMember] private Guid? \_pharmacyId; // ІД аптеки доставки

[DataMember] public int OrderId;

public Order(string name = "Замовлення №{0}") : base(string.Format(name, SecondId))

{

OrderId = SecondId;

CreateTime = DateTime.Now;

Status = OderStatus.Formed;

}

public static int SecondId => Items.Values.Select(order => order.OrderId).Concat(new[] {0}).Max() + 1;

public Client Client

{

get { return \_clientId != null ? Client.Items[\_clientId.Value] : null; }

set { \_clientId = value?.Id; }

}

public Pharmacy Pharmacy

{

get { return \_pharmacyId == null ? null : Pharmacy.Items[\_pharmacyId.Value]; }

set { \_pharmacyId = value?.Id; }

}

[DataMember]

public DateTime CreateTime { get; set; } // Дата створення

[DataMember]

public DateTime CloseOrderTime { get; set; } // Дата закриття замовлення

[DataMember]

public OderStatus Status { get; set; } // Статус замовлення

[DataMember]

public int Discount { get; set; } // Дисконт

[DataMember]

public decimal Price { get; set; } // Сума заказу без знижки

[DataMember]

public string ReasonRejection { get; set; } // Причина відхилення замовлення

[DataMember]

public bool NeedRecipe { get; set; } // Потрібен рецепт для отримання

public Dictionary<Medicin, int> Medicines

{

get

{

var temp = new Dictionary<Medicin, int>();

foreach (var value in Purchase.Items.Values)

{

if (value.Order == this)

{

if (!temp.ContainsKey(value.Medicin))

temp.Add(value.Medicin, value.Count);

}

}

return temp;

}

}

public void ChangeStatus(OderStatus status)

{

Status = status;

if (status == OderStatus.Received || status == OderStatus.Deflected)

CloseOrderTime = DateTime.Now;

}

}

}

# ДОДАТОК Б

# Діаграма класів

Pharmacy

1

∞

∞

1

Order

Client

Medicin

1

1

Purchase

∞

∞

# ДОДАТОК В

# Діаграма наслідування

