Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Веб-орієнтована програмна система замовлення ліків «Аптека Гігеї»
(тема)

Виконав:
студент 4 курсу, групи ПЗПІ-20-1

Петько І. С.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Програмна інженерія
(повна назва освітньої програми)

Керівник ст.викл. кафедри ПІ Олійник О.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти перший (бакалаврський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми Освітньо-професійна
Освітня програма Програмна Інженерія
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Петьку Іллі Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Веб-орієнтована програмна система замовлення ліків «Аптека Гігеї»

Затверджена наказом по університету від 20.05. 2024р. № 471 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 20.06.2024

3. Вихідні дані до роботи Розробити веб-орієнтований застосунок замовлення ліків, який міститиме в собі систему замовлення, підсистему керування ліками, замовленнями та звітами

4. Перелік питань, що потрібно опрацювати в роботі

Вступ, аналіз предметної галузі, формування вимог до програмної системи, архітектура та проектування програмного забезпечення, опис прийнятих програмних рішень, тестування розробленого програмного забезпечення, висновки, додатки.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	20.05.2024	*виконано*
2	Створення специфікації ПЗ	22.05.2024	*виконано*
3	Проектування ПЗ	24.05.2024	*виконано*
4	Розробка ПЗ	28.05.2024	*виконано*
5	Тестування ПЗ	30.05.2024	*виконано*
6	Оформлення пояснювальної записки	05.06.2024	*виконано*
7	Підготовка презентації та доповіді	10.06.2024	*виконано*
8	Попередній захист	14.06.2024	*виконано*
9	Нормоконтроль, рецензування	15.06.2024	*виконано*
10	Здача роботи у електронний архів	18.06.2024	*виконано*
11	Допуск до захисту у зав. кафедри	19.06.2024	*виконано*

Дата видачі завдання 20 травня 2024р.

Студент (ка)_____ Петько І.С.
(підпис)

Керівник роботи _____ ст.викл. кафедри ПІ Олійник О.О.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка до кваліфікаційної роботи бакалавра: 65 стор., 50 рис., 2 табл., 16 джерел.

ВЕБ-ЗАСТОСУНОК, ОНЛАЙН МАГАЗИН, ОНЛАЙН АПТЕКА, ASP.NET CORE, ENTITY FRAMEWORK CORE, MS VISUAL STUDIO 2022, MVC.

Об’єкт розробки – веб-орієнтована система замовлення ліків «Аптека Гігєї» з можливістю пошуку аналогів по різним категоріям.

Мета розробки – створити зручний аналог систем покупки-продажу ліків, де звичайні люди зможуть шукати ліки, а адмінстратори аптеки зможуть додавати свої товари для продажу, змінювати інформації про них, та бачити статистику покупок і рекомендації щодо закупівель нових ліків; розробити веб застосунок для реалізації цих функцій.

Метод рішення – аналіз та моделювання предметної області, UML моделювання, середовище розробки MS Visual Studio 2022, мова програмування C#, із залученням технології ASP.NET Core та entity framework core.

Результатом роботи є веб-орієнтована система продажу ліків «Аптека Гігеї».

WEB APPLICATION, ONLINE STORE, ONLINE PHARMACY, ASP.NET CORE, ENTITY FRAMEWORK CORE, MS VISUAL STUDIO 2022, MVC.

The object of development is a web-oriented drug ordering system "Gygei Pharmacy" with the ability to search for analogues in various categories.

The goal of the development is to create a convenient analogue of drug buying and selling systems, where ordinary people will be able to search for drugs, and pharmacy administrators will be able to add their products for sale, change information about them, and see purchase statistics and recommendations for purchasing new drugs; develop a web application to implement these functions.

Solution method – domain analysis and modeling, UML modeling, MS Visual Studio 2022 development environment, C# programming language, involving ASP.NET Core and entity framework core technology.

The result of the work is a web-oriented drug sales system "Gygei Pharmacy".

Я, Петько Ілля Сергійович, студент гр. ПЗПІ-20-1, здобувач вищої освіти на першому (бакалаврському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Веб-орієнтована програмна система замовлення ліків «Аптека Гігеї»», що буде представлена до екзаменаційної комісії для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElAr KhNURE. Усі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови до допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

Вступ 8
1 Аналіз предметної галузі 9
1.1 Аналіз предметної галузі 9
1.1.1 Дослідження теми «Онлайн аптеки» 9
1.1.2 Дослідження ринку 10
1.2 Виявлення проблем та актуалізація рішень 16
1.3 Постановка задачі 16
2 Формування вимог до програмної системи 18
2.1 Формулювання вимог до веб-застосунку 18
2.2 Формулювання вимог до серверного застосунку 19
2.3 Вимоги до інтерфейсу 20
2.4 Загальні обмеження 20
2.5 Припущення та залежності 20
3 Архітектура та проєктування програмного забезпечення 21
3.1 UML проєктування ПЗ 21
3.2 Проєктування архітектури ПЗ 24
3.3 Проектування бази даних 24
3.4 Діаграма класів 25
4 Опис прийнятих програмних рішень 31
4.1 Опис прийнятих архітектурних та інфраструктурних рішень 31
4.2 Програмні рішення шару презентації 34
4.3 Програмні рішення шару стійкості 35
4.4 Програмні рішення шару бізнес логіки 36
4.5 Програмні рішення шару представлення 39
4.6 Прийняті рішення щодо інтерфейсу системи 40
5 Тестування розробленого програмного забезпечення 48
Висновки 52
Перелік джерел посилання 53
Додаток А 55

Додаток Б..... 56

ВСТУП

Останнім часом на нашу країну обрушується багато викликів: хвороби, карантини, військові вторгнення, і так далі. Всі вони значним чином обмежують мобільність громадян. Це веде до того що громадяни не можуть отримати послуги у звичному вигляді, і призводить до росту потреби в дистанційному наданні послуг.

Електронні магазини мали зростання популярності і до всіх цих жахливих подій. Але після їх початку та розвитку популярність онлайн покупок значно зросла. Ця робота є рефлексією на стан ринку.

Мною було поставлено запитання, що таке я можу зробити корисного в рамках своєї роботи? Відповідь проста – онлайн крамничку. Які групи товарів є первинними для нашого населення в ці важкі часи – їжа та ліки. Вибір було зупинено на ліках. Тому що я особисто зіткнувся з важкою ситуацією пошуку аналогів для звичних ліків. Через кризу на границі звичні препарати не надходили, а пошук аналогів через існуючі онлайн аптеки був жахливою пригодою. Тому і виникла ідея написати власну систему яка не буде мати такого недоліка.

Як символ вирішено обрати богиню Гігею з рук якої ми приймали медицину дуже довго.

Метою роботи є розробка веб орієнтованої системи «аптека Гігеї» яка допоможе користувачам знаходити потрібні їм ліки, обирати їх аналоги якщо вони є у наявності, для адміністратора передбачена статистика замовлень, формування товарних накладних, формування рекомендацій щодо закупівлі товарів, та зручний інтерфейс керування товарами на сайті.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметної галузі

1.1.1 Дослідження теми «Онлайн аптеки»

Купівля ліків займає помітну частину часу в нашому житті та вимагає помітної частки коштів з нашого бюджету. Це відбувалося і в звичайні часи, а в часи пандемії та ті що настали після неї потреба в ліках лише зросла. Причому ріст звичних мереж офлайн продажу ліків уповільнився через карантинні обмеження та логістичні і ризикові обмеження після початку бойових дій. Торгівля онлайн же отримала суттєвого пришвидшення. Це, звісно, стосується не тільки аптек, але і звичайних торгівельних майданчиків.

Розглянемо переваги які надають онлайн-аптеки покупцям.

Зручність і доступність: Онлайн аптеки дозволяють українцям замовляти ліки та медичні товари безпосередньо з будь-якого місця, де є доступ до Інтернету. Це особливо корисно для тих, хто проживає в віддалених районах або має обмежену можливість фізично відвідувати аптеку.

Широкий вибір товарів. Онлайн аптеки можуть пропонувати більший асортимент медичних препаратів, включаючи ті, які можуть бути складно знайти в звичайних аптеках. Це дозволить людям знайти необхідні ліки та медичні товари без зайвих пошуків.

Доступ до інформації і консультацій. Багато онлайн аптек надають інформацію про ліки та їх використання, а також можуть мати онлайн-консультантів або фармацевтів, які надають поради щодо правильного застосування ліків та відповідають на запитання користувачів.

Безпека та якість. Якщо онлайн аптека має відповідні ліцензії та дозволи, вона може забезпечувати якість та безпеку медичних препаратів, що продаються, що є важливим фактором для споживачів.

Цінова конкуренція. З'явлення нових онлайн аптек може призвести до збільшення конкуренції на ринку, що може позитивно позначитися на цінах для споживачів.

Враховуючи ці фактори, створення нових онлайн аптек може стати корисним для українських споживачів, забезпечуючи їм зручний та доступний спосіб отримання медичних товарів і послуг.

Розглянемо які взагалі існують види онлайн аптек. В [1] наводять таку класифікацію:

- компанія що одразу розробляється як інтернет проект;
- веб сайт звичайної аптеки для розширення клієнтської бази;
- сервісна структура на базі декількох аптек для координації процесів замовлення та доставки товару з них;
- інтернет аптека на базі вже існуючої мережі аптек, доставка здійснюється кур'єром з найближчої аптеки.

В цій роботі буде розглянуто створення електронної аптеки першого типу.

В торгівлі ліками, як і в звичайній торгівлі існують певні законодавчі обмеження. Почнемо з визначення інтернет магазину. Закон України «Про електронну комерцію»[2] надає визначення як засобу для представлення або реалізації товару, роботи чи послуги шляхом вчинення електронного правочину (п. 8 ч. 1 ст. 3). Закон України від 17.09.2020 р. № 904[3] «Про внесення змін до статті 19 Закону України «Про лікарські засоби»[4] щодо здійснення електронної роздрібної торгівлі лікарськими засобами», дозволяє дистанційний продаж безрецептурних ліків.

Таким чином створення нової інтернет-аптеки – не протирічить законам України. Звісно, для подальшого комерційного використання знадобиться пройти процес реєстрації та ліцензування.

1.1.2 Дослідження ринку

Проведення аналізу ринку є дуже важливою частиною процесу створення програмного забезпечення. Цей етап дозволяє подивитись на існуючі на ринку аналоги, зрозуміти які функції вважаються обов'язковими, які переваги є у кожного з аналогів, які проблеми можуть зустрітися при розробці. Аналіз

дозволяє виділити пріоритети для бажаних модулів проекту та підвищити якість програмного забезпечення. [5]

Зараз на ринку присутні декілька великих інтернет аптек та дуже багато менш успішних. У них у всіх є свої плюси та мінуси. Розглянемо деякі з них.

Першою розглянемо сервіс доставки ліків LIKI24 (див.рис.1.1).

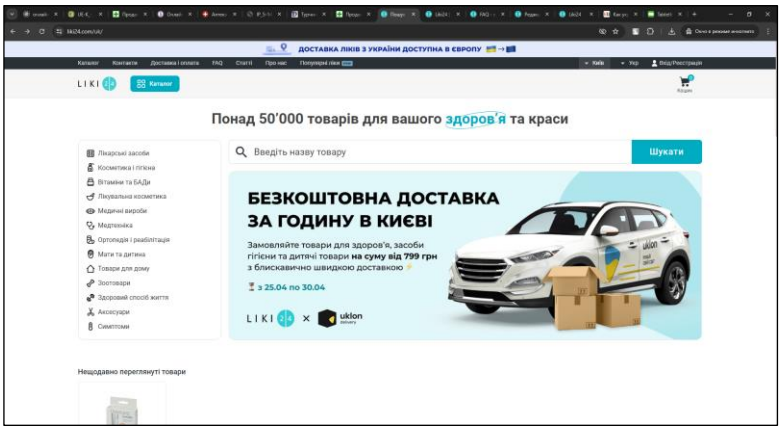


Рисунок 1.1 – Головна сторінка сервісу LIKI24[6]

LIKI24 – система пошуку, бронювання та доставки ліків. Система збирає інформацію про те які препарати є в аптеках та які ціни встановлені на них. В системі відображуються аналоги з ідентичною діючою речовиною. [7]

Для прийому, обробки та доставки заказів у проекта є власна підсистема «Агонь» (див.рис.1.2)

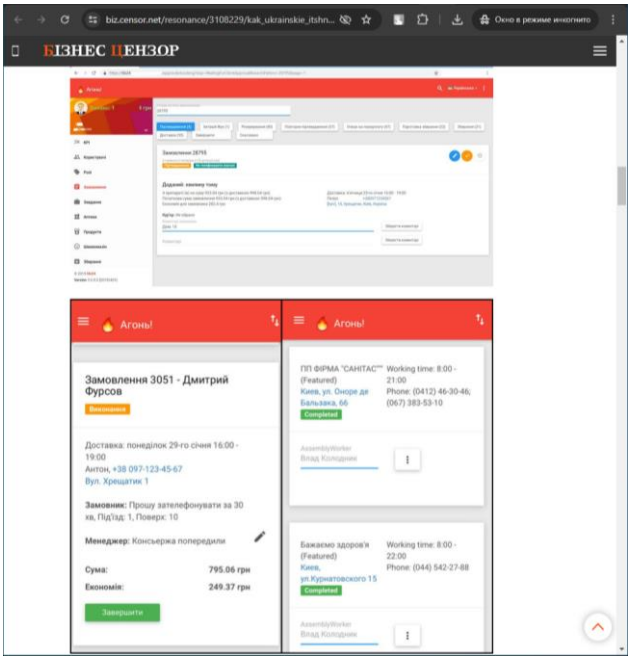


Рисунок 1.2 – Інтерфейс підсистеми доставки «Агонь» [7]

Як стверджується у [7] Ліки24 не стільки інтернет аптека, скільки сервіс посередник між кур'єрами та клієнтами.

Взагалі на самому сервісі є можливість замовити не тільки медичні препарати а і товари інших категорій. Сайт передбачає систему авторизації користувача зі збереженням інформації про нього, статистики замовлень та інше. Присутня локалізація українською та російською мовами.

Щодо самих ліків їх можна шукати за назвою, виконувати пошук по категоріям та симптомам, або брендам. Є розділи з акційними товарами, сезонними товарами.

Кожний товар може бути оцінений користувачами у формі «5 зірок». Також існує можливість залишати коментарі до товарів з профілів зареєстрованих користувачів. До товарів надається електронний варіант інструкції споживача.

З недоліків можна зазначити не дуже зрозумілий механізм пошуку аналогів.

Наступним продуктом для порівняння є Аптека 911 (див.рис.1.3).

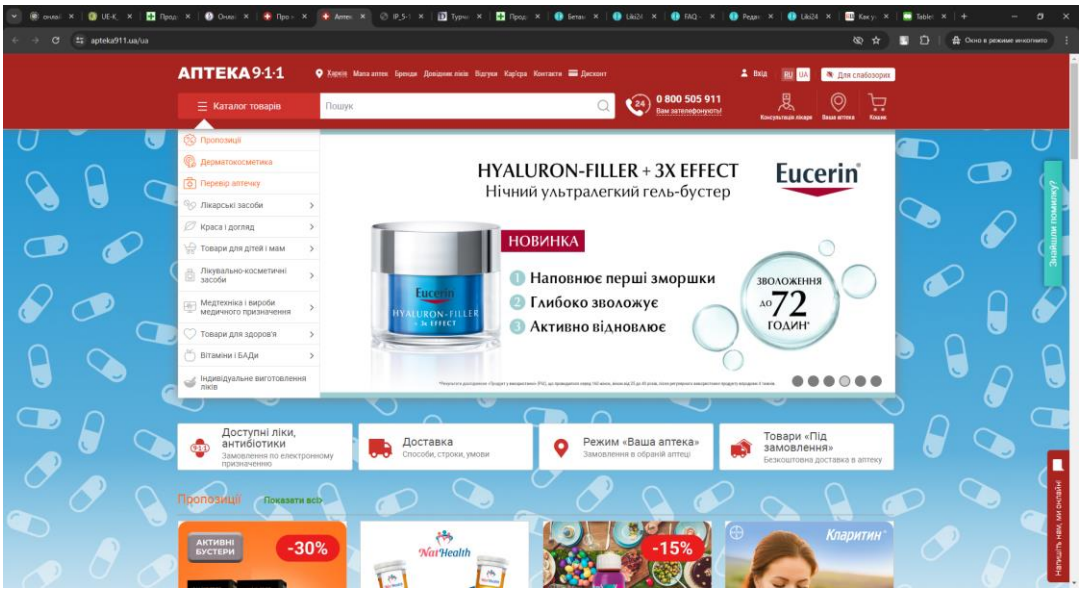


Рисунок 1.3 – Головна сторінка сайту Аптека911 [8]

Самі власники сайту описують свій проект так [8]: ««Аптека 9-1-1» – сучасна та зручна електронна медична інформаційна система. apteka911.ua – перший інноваційний майданчик для бронювання та організації доставки товарів аптечного асортименту. Ми першими запропонували українському споживачеві моментальне on-line бронювання товару в аптеках. Ми першими організували

сервіс віддаленої консультації з сімейним лікарем. Ми першими створили сайт прийому заявок на препарати за державною програмою «Доступні Ліки». Ми першими отримали премію в щорічному конкурсі професіоналів фармацевтичної галузі України «Панацея» в номінації «Кращий сайт».».

При аналізі широко виділяються наступні модулі сайту: розділ зі статтями, розділ консультацій лікарів, мапа аптек, та власне розділ з замовленням та бронюванням ліків. Також на сайті є косметичні товари та медичне обладнання.

Щодо самих ліків передбачено розділення по категоріям, пошук по назві, наявність інструкції виробника, пошук аналогів повних (діюча речовина+дозування), по діючій речовині та по терапевтичному ефекту.

У користувачів є можливість залишити відгук до кожного товару.

В групах товарів розраховується рейтинг по кількості покупок за останні 30 днів. В середині груп передбачені сортування ще й по ціні, а також фільтрація по виробнику.

Товар можна замовити та оплатити, або забронювати в певній аптеці зі списку.

До недоліків можна віднести не дуже дружелюбний інтерфейс. Отримання доступу до деяких можливостей системи не завжди є очивидним.

Третім продуктом розглянемо TABLETKI.UA (див.рис.1.4).

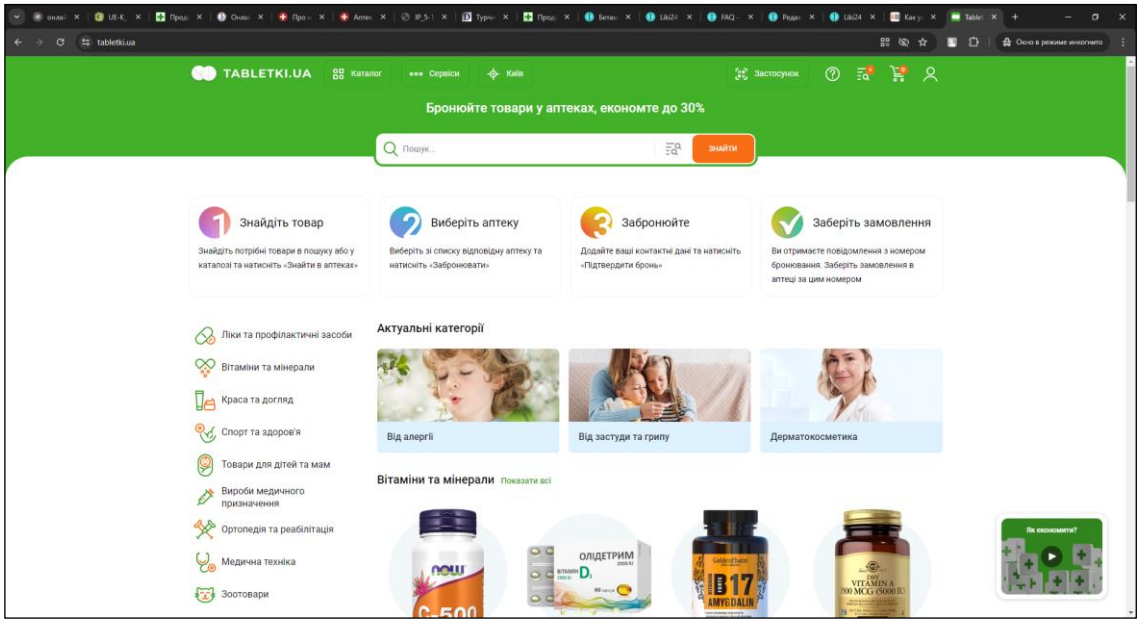


Рисунок 1.4 – Головна сторінка сайту Tabletki.ua [9]

Як пишуть в [10] «Tabletki.ua це онлайн-платформа, що надає велику інформацію про медичні препарати, їх інструкції та відгуки. Сайт допомагає користувачам приймати поінформовані рішення щодо застосування ліків, забезпечуючи доступ до актуальних даних про медикаменти».

Сам сервіс позиціонує себе як засіб для пошуку та бронювання потрібних вам ліків в найближчій від вас аптеці. Сервіс може самостійно визначити ваше місце знаходження, особливо якщо Ви користуєтесь ним з мобільного пристрою та вивести на карту ціни шуканого препарату в найближчих аптеках (див.рис.1.5).

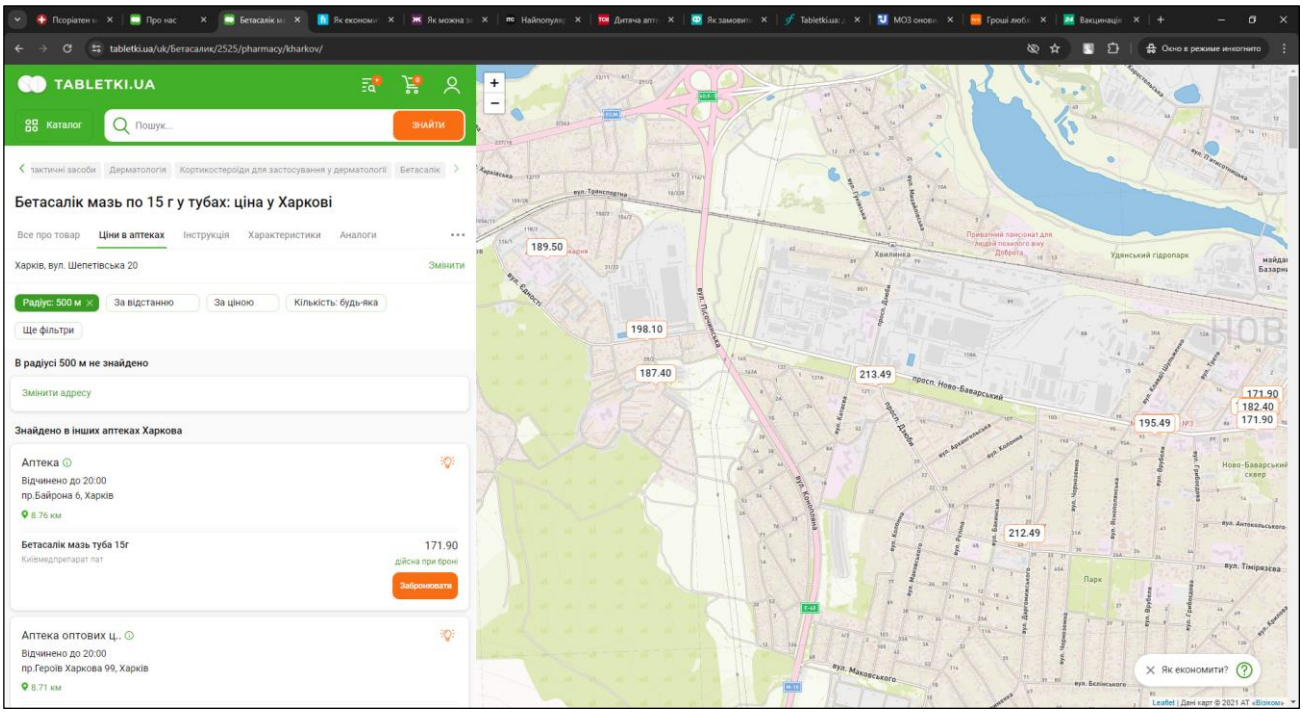


Рисунок 1.5 – Приклад роботи сервісу [9]

Товари для пошуку поділяються на категорії. Представлені не тільки ліки но і супровідні товари, а також зоотовари та товари для спорту.

На веб сайті розроблена дуже гарна система пошуку. Товари розбиті по категоріям, додано дуже багато різних можливостей для фільтрації. Лише категорій фільтрів 16 штук. В кожній з них мінімум по два фільтри. Пошук працює по всім параметрам товару: по назві, бренду, діючій речовині, її дозировці і т.д. Особливістю сервісу є пошук списку товарів. Після формування списку покупок буде надано список аптек де буде вказано які товари з користувацького

списку є в кожній і скільки буде коштувати придбання всіх товарів що є в наявності.

В описі кожного товару є фото, інструкція, опис товару, дуже подробні особливості застосування, пошук аналогів за повним складом чи терапевтичною дією, та статистика коливання середньої ціни серед підключених аптек за останній рік.

Серед недоліків можна виділити відсутність пошуку аналогів по діючій речовині, відсутність рейтингів товарів та можливості коментувати для користувачів (див.табл.1.1).

Таблиця 1.1 – Порівняльна таблиця

	Таблеткі юа	Аптека911	LIKI24
Пошук	+	+	+
Пошук списку	+	-	-
Великий список фільтрів	+	+-	+-
Інструкція виробника	+	+	+
Особливості застосування	+	+	+
Пошук повних аналогів	+	+	-
Пошук аналогів по діючій речовині	-	+	+-
Коментування товарів	-	+	+
Оцінка товарів	-	+	+
Пошук найближчих аптек	+	+-	-
Мапа аптек	+	+	-
Консультація лікаря	-	+	-
Бронювання товару	+	+	-
Придбання товару онлайн	-	+	+
Власна система доставки	-	-	+
Наявність розділу зі статтями	-	+-	+
Інші категорії товарів	-	+-	+

Порівняльна таблиця демонструє, що три найпопулярніші сервісу для придбання ліків онлайн мають схожий функціонал, з певними особливостями для кожного з них. Протягом цієї роботи я спробую повторити ті функції що є у всіх трьох, та, можливо дещо зверху.

1.2 Виявлення проблем та актуалізація рішень

Проведений аналіз продемонстрував що розглянуті системи є дуже популярними на українському ринку. Також, не дивлячись на зовнішню схожість вони доволі суттєво відрізняються за метою роботи та внутрішнім функціоналом.

Якщо ж провести порівняння цих гігантів можна виділити обов'язкові модулі що мають бути імплементовані в розроблюваній системі, а саме: модулі пошуку(по назві, по виробнику, повних аналогів, аналогів по діючій речовині, по виробнику, по аптеці), модулі фільтрації, модуль відображення списку товарів, модуль взаємодії з аптеками, модуль бронювання або замовлення. Також обов'язково потрібно продумати відображення на сайті інструкції виробника та особливостей застосування. Інші аптеки що булі розглянуті, але не ввійшли в цей документ не мали цих частин, можливо саме це разом з менш привабливим інтерфейсом вплинуло на їх меншу популярність.

Основною проблемою видиться збір даних про ліки які будуть відображуватись для покупців. А це значить, що обов'язковим є модуль додавання-редагування товарів зі зручним інтерфейсом.

Також доцільним бачиться додання розділу зі статистикою для покупців та продавців. Статистика для покупців є на одному з трьох ресурсів. Статистика для продавців гарантовано є на одному з трьох, і можливо є ще на одному. Таким чином можна зробити висновок, що її наявність є конкурентною перевагою.

1.3 Постановка задачі

Метою роботи є розроблення вимог та створення веб орієнтованої системи замовлення ліків «Аптека Гігєї».

Система має надавати два інтерфейси користувачів: для покупця та для аптеки. Користувач має отримати можливість обирати товари з сайту та виконувати замовлення, аптека має отримати можливість додавати та редагувати товари та бачити статистику по закупкам у себе.

Серверна частина має оброблювати запити від користувачів та надавати на них відповіді.

Веб-частина повинна відтворювати зручний інтерфейс для відображення відповідей сервера користувачам.

Дані повинні зберігатись в базі даних, навіть у випадках непередбачених відключень.

2 ФОРМУВАННЯ ВИМОГ ДО ПРОГРАМНОЇ СИСТЕМИ

2.1 Формулювання вимог до веб-застосунку

Веб застосунок повинен взаємодіяти з сервером за допомогою протоколу HTTPS[11]. Потрібно забезпечити можливість адаптації до різних браузерів та розмірів екрану. Глобальним завданням цієї частини програмного застосунку є забезпечення зв’язку серверної частини та клієнта між собою, переклад їх запитів та відповідей у зручну для іншої сторони форму.

Веб застосунок повинен поділятись на дві частини: клієнт, адміністратор.

Частина клієнта повинна забезпечувати наступні функції:

- відображення списку товарів;
- пошук та фільтрація товарів;
- можливість реєстрації та авторизації.

Частина аптеки повинна забезпечувати наступні функції:

- механізм додавання та редагування товарів;
- відображення статистики по аптеці по покупкам та витратам;
- відображення статистики по найбільш популярним товарам;
- відображення статистики по найбільш популярним групам товарів;
- відображення поточних розходів та доходів за обраний проміжок часу;
- можливість формування pdf документу-звіту, з відображенням списку необхідних закупівель за останні 30 діб;
- можливість змінювати границі для відображення статистики: на день, 7 днів, 30 днів та весь час роботи аптеки;
- автоматичне форматування списку закупівель, як списку всіх відправлених товарів;
- автоматичне форматування замовлень;
- можливість редагування списку товарів;
- можливість видалення товарів;
- можливість додавання, редагування, видалення списку категорій;

- можливість автоматичного контролю наявності товару на складі перед відправкою замовлення;
- можливість автоматичного зменшення наявного на складі товару при відправленні замовлення.

2.2 Формулювання вимог до серверного застосунку

Сервер – частина програмної системи, що відповідає за виконання обслуговуючих функцій, та надання відповідей на запити клієнтської частини [12]. Сервер саме і забезпечує роботу системи в цілому.

Алгоритм роботи клієнт-серверної архітектури приблизно такий:

- сервер неперервно очікує запиту клієнта;
- клієнт формує запит та відправляє його на сервер;
- сервер оброблює запит та формує відповідь;
- сервер відправляє клієнту отриману відповідь;
- клієнт відображує користувачеві отриману відповідь.

Саме в розроблюваному застосунку сервер повинен виповнювати наступні функції:

- зберігання інформації про товари їх категорії та користувачів;
- забезпечити можливість обробки великої кількості інформації;
- забезпечення захисту інформації від незаконного отримання;
- забезпечити можливість формування наборів даних для отримання статистики;
- забезпечити можливість формування наборів даних для формування звітів;
- забезпечення легального доступу до збереженої інформації.

Для розробки застосунку обрано наступний стек технологій: мова програмування C#, фреймворк .NET, технологія ASP.NET MVC Core, для бази даних – MS SQL server, для відображення деяких деталей інтерфейсу використано JavaScript/

2.3 Вимоги до інтерфейсу

В [14] такі вимоги до інтерфейсу:

- відповідність вимогам користувача;
- відповідність технології;
- зрозумілість та логічність;
- забезпечення високої швидкості роботи користувача;
- забезпечення обробки великої кількості товарів та категорій;
- забезпечення захисту від людських помилок;
- швидке навчання користувача;
- суб'єктивне задоволення користувача.

При розробці цієї системи ми спробуємо притримуватись наданих вимог.

2.4 Загальні обмеження

Розроблювана система має наступні обмеження:

- система залежіть від підключення до інтернету;
- встановлюється мінімальний вік для реєстрації – 16 років, це пов’язано з тим що система працює з грошима та потенційно небезпечними речовинами;
- на одну електронну пошту може бути створено лише один обліковий запис;
- інтерфейс та вміст представлено лише українською мовою.

2.5 Припущення та залежності

Припущення та залежності системи:

- система для своєї роботи вимагає наявності стабільне підключення до Інтернету у сервера та користувача;
- користувач несе відповідальність за правдивість вказаної інформації.

3. АРХІТЕКТУРА ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. UML проектування ПЗ

Під час розробки системо створено декілька UML діаграм. Почнемо з use-case[15] діаграми.

Система передбачає декілька типів користувачів, а саме: неавторизовані користувачі, авторизовані покупці, авторизовані адміністратори системи.

Система надає наступні функції для неавторизованого користувача:

- реєстрація та авторизація;
- перегляд списку доступних ліків;
- перегляд списку категорій ліків;
- пошук та фільтрація;
- замовлення товарів.

Авторизовані покупці мають весь функціонал неавторизованого користувача плюс:

- автоматичне заповнення полів при формуванні замовлення.

Авторизованим адміністраторам системи доступний весь функціонал доступний авторизованому покупцю плюс:

- керування власними товарами (додавання, редагування, видалення);
- відображення статистики (по прибуткам та витратам, популярності товарів, популярності груп товарів);
- управління замовленнями;
- відображення статистики по власним товарам;
- автоматичне формування замовлень від покупців;
- формування та змінювання замовлення в ручному режимі;
- керування списком категорій;
- автоматичне зменшення кількості ліків на складі при відправленні замовлення;

– автоматична перевірка наявності ліків на складі перед відправкою замовлення;
– автоматичне форматування рекомендованого списку закупівель за останні 30 днів, як списку всіх відправлених товарів.

Діаграма use-case для неавторизованого користувача буде виглядати наступним чином (див.рис.3.1):

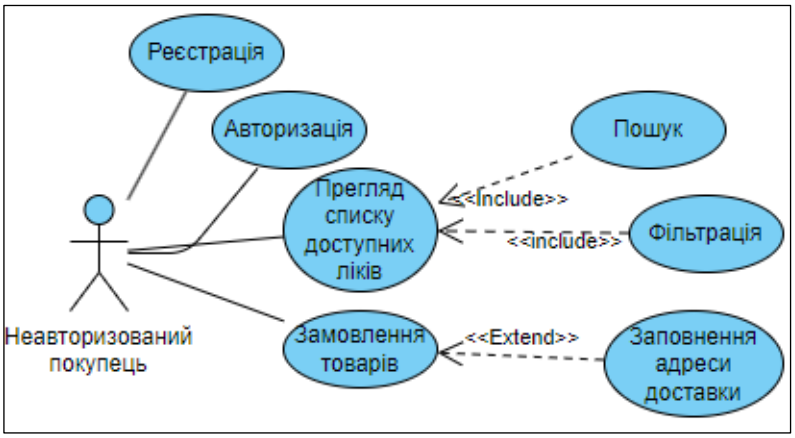


Рисунок 3.1 – Діаграма use-case для неавторизованого користувача

Діаграма use case для авторизованого покупця буде виглядати наступним чином (див.рис.3.2):

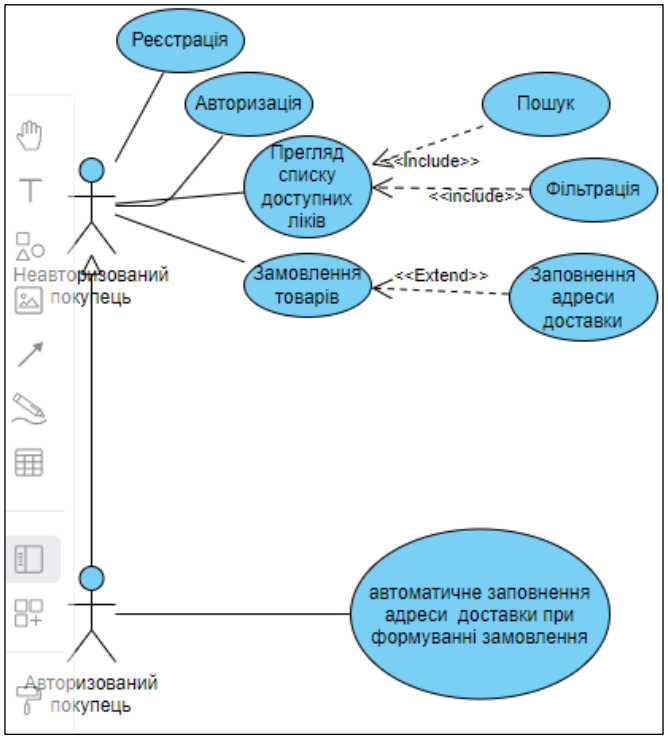


Рисунок 3.2 – Частина діаграми use-case для авторизованого покупця

Діаграма use-case для авторизованого адміністратора буде виглядати наступним чином (див.рис.3.3):

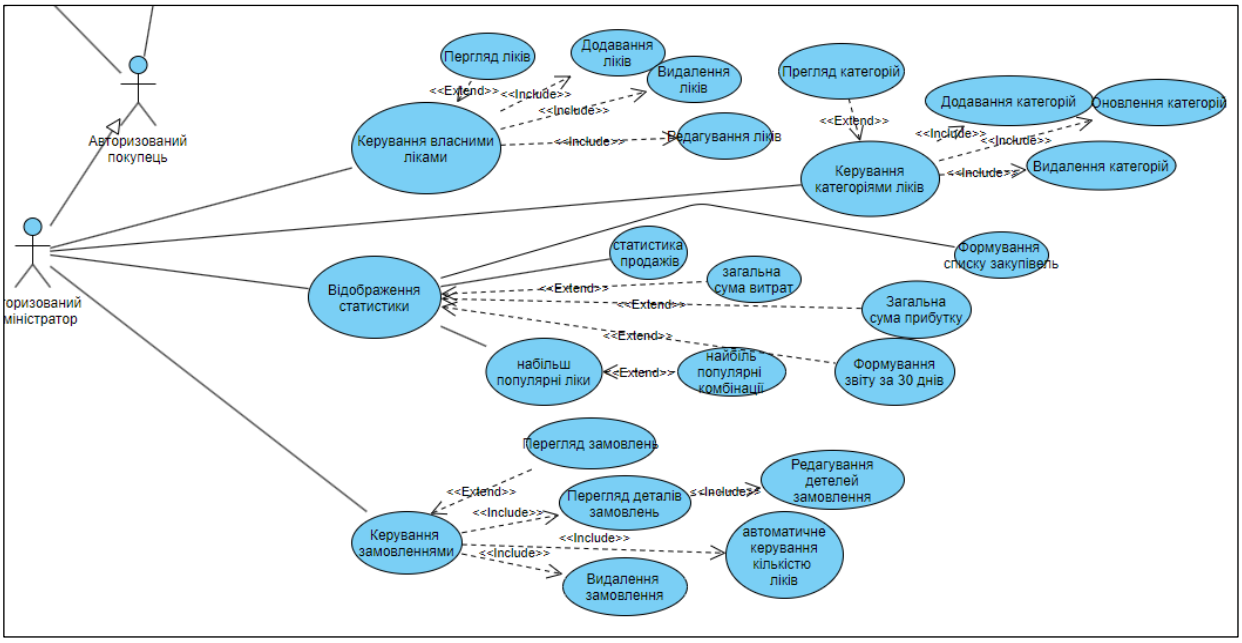


Рисунок 3.3 – Частина діаграми use-case для авторизованого адміністратора

Загальна use case діаграма проекту виглядатиме приблизно так (див.рис.3.4):

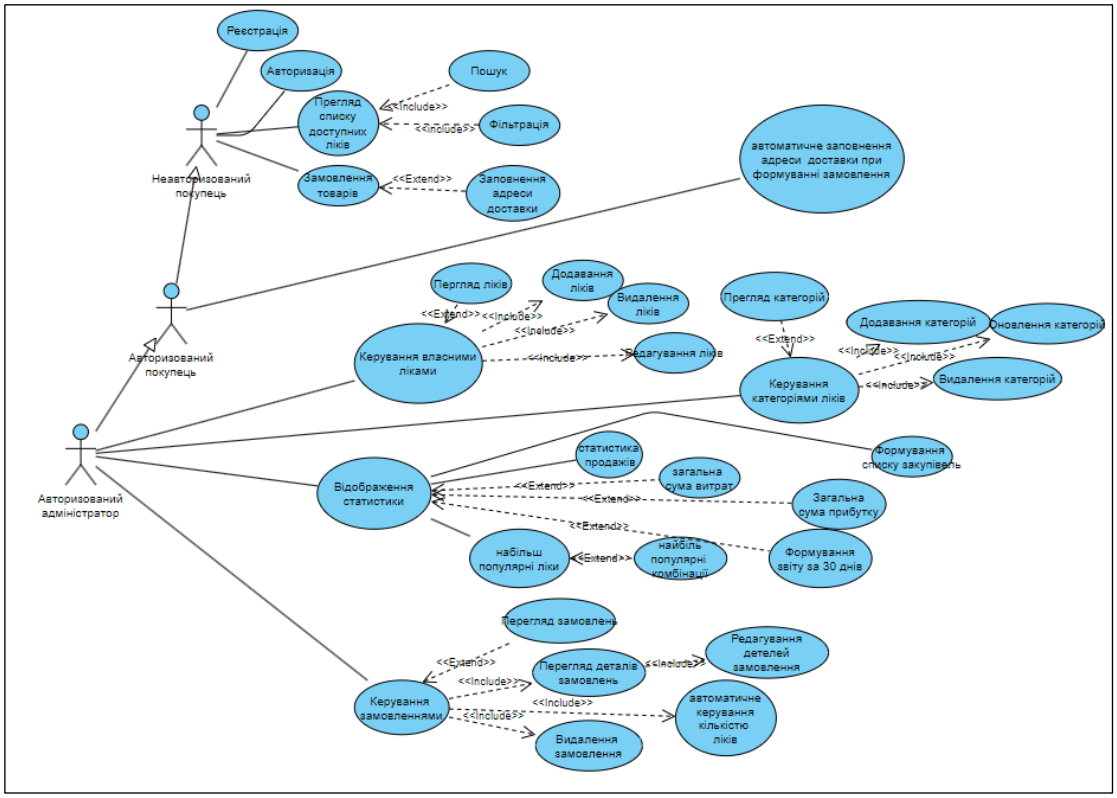


Рисунок 3.4 – Загальна use case діаграма верхнього рівня проекту

3.2 Проєктування архітектури ПЗ

Система буде побудована за допомогою мови C#, фреймворку .NET, архітектурного шаблону MVC, MSSQL (див.рис.3.5).

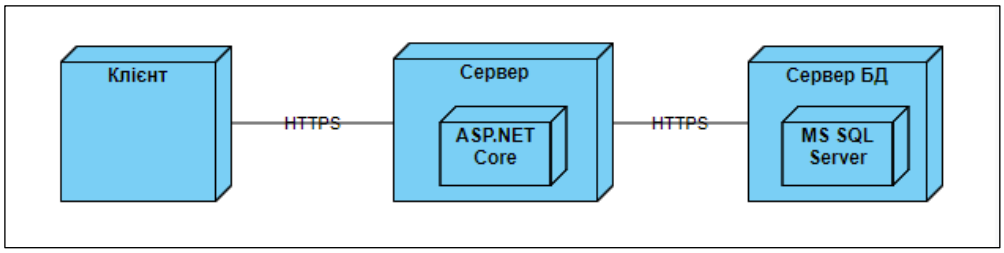


Рисунок 3.5 – Діаграма розгортання

Канали зв’язку планується організувати за протоколом HTTPS.

3.3 Проектування бази даних

Всі дані будуть зберігатись на віддаленому сервері. Для цього використано СУБД Microsoft SQL Server [15]. Усі таблиці приведені до третьої нормальної форми.

База даних містить сім взаємопов’язаних таблиць. На рисунку 3.6 можна бачити ER діаграму системи.

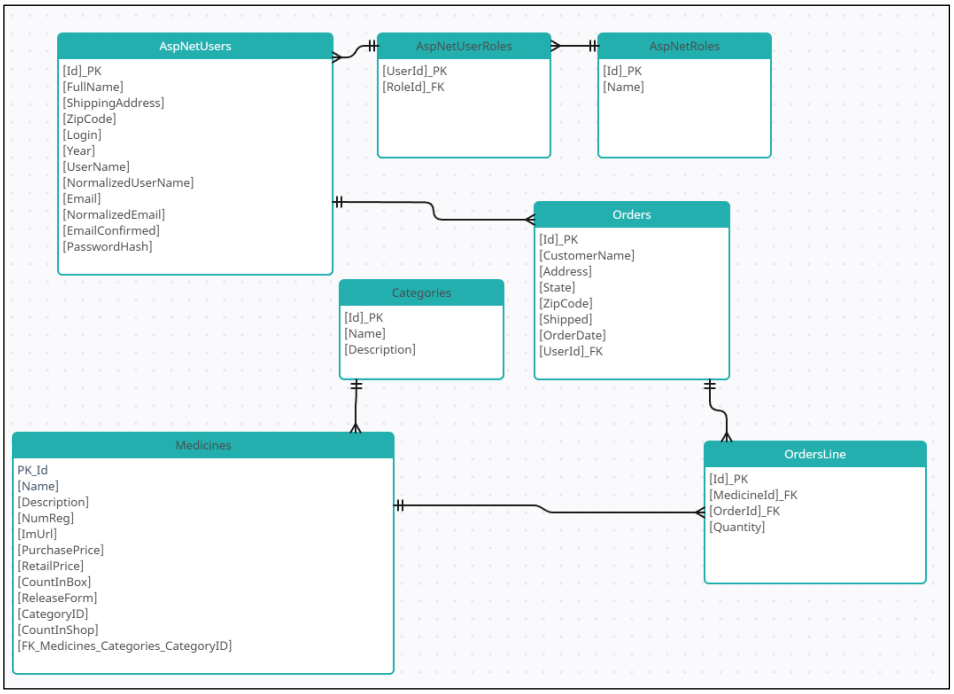


Рисунок 3.6 – ER діаграма системи

База даних складається з наступних таблиць:

– Ліки– зберігає дані про ліки що є в системі;
– Категорія– зберігає дані про категорії;
– Заказ– зберігає дані про закази;
– Рядок_заказу– слугує в’язучою таблицею між Заказами та Ліками;
– Користувач– зберігає дані про користувачів;
– РольКористувача– слугує в’язучою таблицею між Користувачем та Ролями;
– Роль– зберігає дані про доступні ролі користувачів.

3.4 Діаграма класів

На рисунку 3.7 представлена діаграма класів.

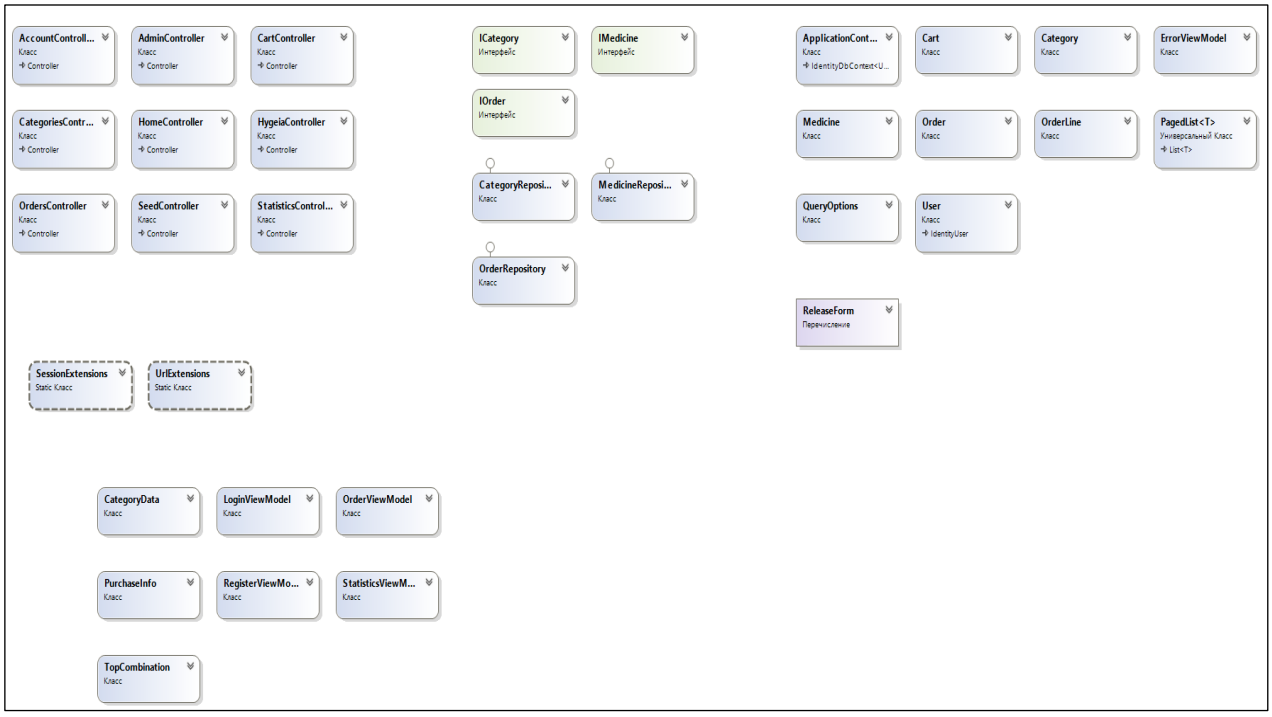


Рисунок 3.7 – Загальна діаграма класів

Розроблена система містить 32 класи та 3 інтерфейси. Ці класи поділені на групи та виконують різні завдання:

Почнемо з групи інфраструктурних класів (див.рис. 3.8):

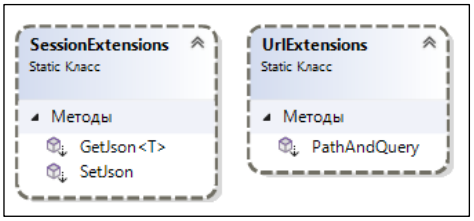


Рисунок 3.8 – Група інфраструктурних класів

Класи SessionExtensions та UrlExtensions використовуються в службових цілях для зберігання поточної сесії та Url попередньої сторінки.

Перейдемо до групи класів моделей (див.рис.3.9):

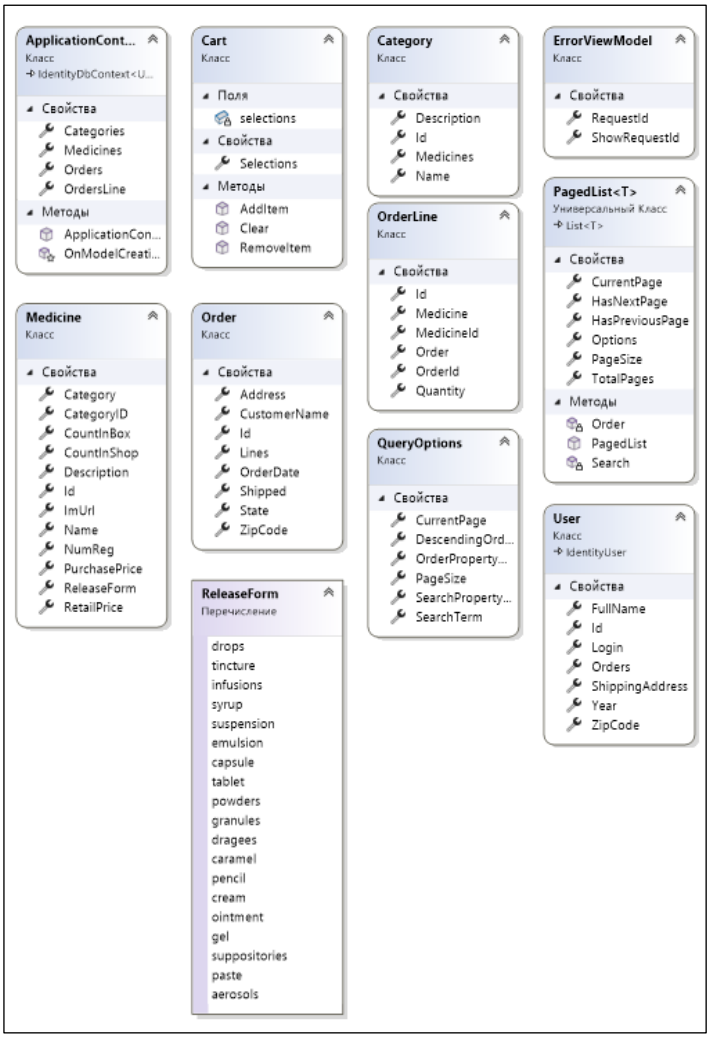


Рисунок 3.9 – Класи групи Моделі

В проекті використано підхід спочатку код. Тому ці створені класи моделі слугують для формування представлення даних в базі даних. Наявні наступни класи:

– PagedList – використовується для організації пагінації;
– QueryOptions –використовується для організації пошуку;
– ApplicationContext – використовується для налаштування контексту додатку;
– Cart – використовується для організації кошику;
– Category – використовується для роботи з категоріями ліків;
– ErrorViewModel – використовується для обробки помилок передачі даних на представлення;
– Medicine – використовується для роботи з ліками;
– Order – використовується для роботи з замовленнями;
– OrderLine – використовується для роботи з кожним рядком замовлення;
– ReleaseForm – перерахування, використовується для завдання форм випуску ліків;
– User – використовується для організації роботи з користувачами;

Далі перейдемо до групи класів контроллерів (див.рис.3.10):

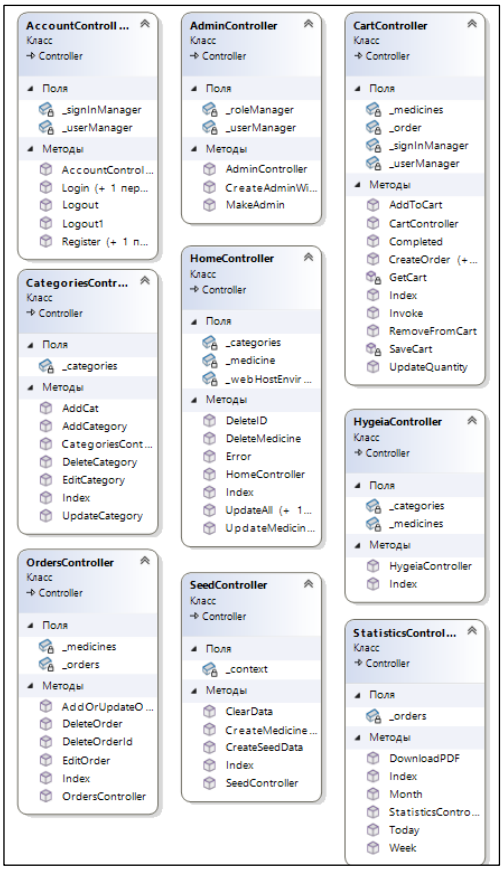


Рисунок 3.10 – Група класів контролерів

Ця група класів використовується для завдання процедур обробки дій користувачів. В цій групі присутні наступні класи:

– AccountController – використовується для керування аккаунтами користувачів, тут виконуються операції регістрації та авторизації;
– AdminController – використовується для надання користувачу повноважень адміністратора;
– CartController- використовується для організації роботи корзини;
– CategoriesController – використовується для організації роботи з категоріями;
– HomeController – використовується для організації роботи авторизованих адміністраторів;
– HygeiaController – використовується для організацій роботи неавторизованих та авторизованих покупців;
– OrdersController – використовується для організації обробки заказів;
– SeedController – використовується для наповнення бази даних тестовою інформацією;
– StatisticsController – використовується для організації роботи зі статистикою для авторизованих адміністраторів.

Перейдемо до групи класів що відповідають за роботу з джерелом інформації. В проекти використано патерн Репозиторій. Для того щоб слідувати йому були створені наступні класи та інтрефейси (див.рис.3.11):

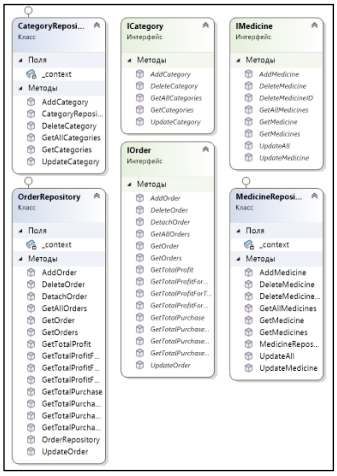


Рисунок 3.11 – Класи використані для організації репозиторію

В цій группі присутні наступні інтерфейси:

– ICategory – відповідає за керування інформацією о категоріях;

– IMedicine – відповідає за керування інформацією о ліках;

– IOrder – відповідає за керування інформацією о замовленнях.

Та класи що реалізують ці інтерфейси:

– CategoryRepository – реалізує інтерфейс ICategory;

– MedicineRepository – реалізує інтерфейс IMedicine;

– OrderRepository – реалізує інтерфейс IOrder.

Наступна група класів слугує для покращення організації передачі даних на сторону відображення(див.рис.3.12):

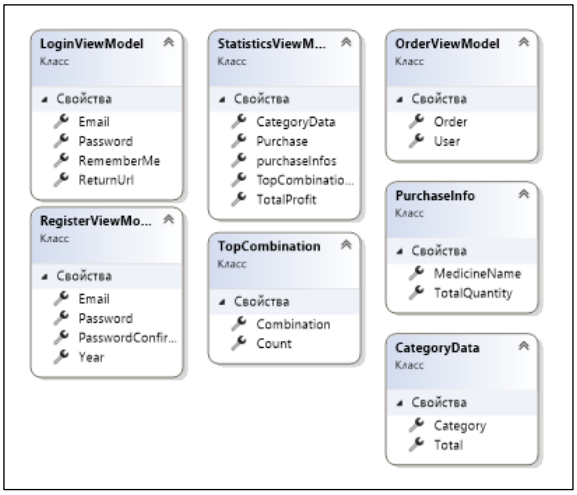


Рисунок 3.12 – група класів ViewModel

В цій групі присутні наступні класи:

– CategoryData – використовується при організації відображення довільної кількості категорій;

– LoginViewModel – використовується для організації сторінки авторизації;

– OrderViewModel – використовується для організації зв’язку між користувачем та замовленням;

– PurchaseInfo – використовується для формування сторінки статистики;

– RegisterViewModel – використовується для організації сторінки регістрації;

– StatisticsViewModel – використовується для формування сторінки статистики;
– TopCombination – використовується для формування сторінки статистики;

Для відображення функціональності покупки ліків було сформовано діаграму активностей (див.рис.3.13):

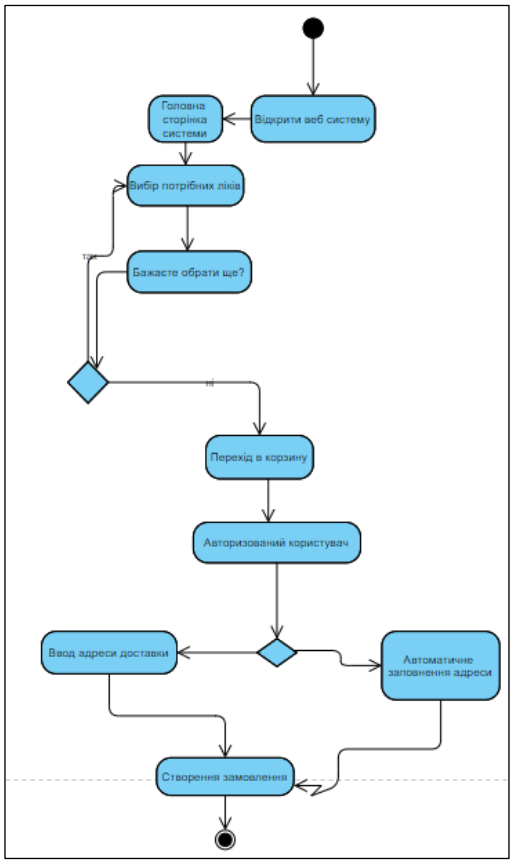


Рисунок 3.13 – Діаграма активностей покупки ліків

На цій діаграмі ми бачимо як відбувається процес придбання ліків. Користувач спочатку входить в систему. Потім обрає потрібні йому ліки, в корзині має можливість змінити кількість кожної позиції замовлення. Після вибору всіх потрібних позицій та їх кількості користувач формує замовлення вносячи інформацію про себе та адресу доставки. Якщо користувач авторизований ці данні підтягуються автоматично, але якщо він бажає – то може їх змінити.

4 ОПИС ПРИЙНЯТИХ ПРОГРАМНИХ РІШЕНЬ

4.1 Опис прийнятих архітектурних та інфраструктурних рішень

Через одиночну розробку програмного забезпечення «аптека Гігєї» було обрано мову C#, фреймворк ASP.NET Core, версію для створення веб додатків. Як середевище розробки використовувалась MS Visual Studio 2022, як СКБД – MS SQL Server.

Для розробки додатку було обрано патерни Модель-Представлення-Контроллер та Репозиторій для керування даними.

Патерн Репозиторій дозволяє доволі швидко змінювати систему, що використовується для керування даними.

Згідно з обраними архітектурними рішеннями була розроблена структура проекту. Її ми можемо побачити на рисунку 4.1

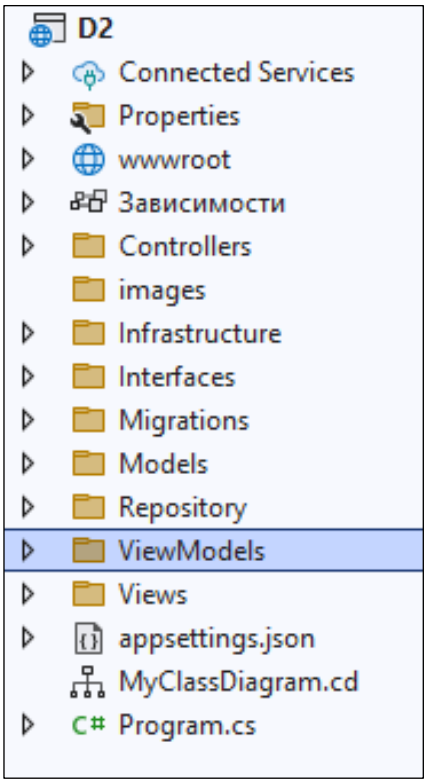


Рисунок 4.1 – Файлова структура проекту

Тут різні папки відповідають за різні шари наших патернів:

– Controllers – папка, місце розташування контролерів, які надають ендпоінти;

– Infrastructure – папка, де розташовані файли класів, що використовуються для відстеження сесії та попереднього URL;
– Interfaces – папка, де розташовані інтерфейси для реалізації патерну Репозиторій;
– Migrations – папка де зберігаються міграції для бази даних;
– Models – папка де зберігаються класи, що описують дані проекту;
– Repository – папка де розташовані реалізації інтерфейсів патерну Репозиторій;
– ViewModels – папка де розташовані допоміжні класи, що використовуються при передачі даних між представленням та контролером;
– Views – папка де зберігаються всі представлення.

Давайте розглянемо декілька прикладів. Почнемо з інтерфейсів. Вони організовані приблизно схожим чином, тому розглянемо інтрефейс репозиторію замовленнь (див.рис.4.2):

```
using D2.Models;
using D2.Models.Pages;

namespace D2.Interfaces
{
    public interface IOrder
    {
        PagedList<Order> GetOrders(QueryOptions options);
        IEnumerable<Order> GetAllOrders();
        Order GetOrder(int id);
        void AddOrder(Order order);
        void UpdateOrder(Order order);
        void DeleteOrder(Order order);
        void DetachOrder(Order order);
        decimal GetTotalProfit();
        decimal GetTotalProfitForToday();
        decimal GetTotalProfitForWeek();
        decimal GetTotalProfitForMonth();
        decimal GetTotalPurchase();
        decimal GetTotalPurchaseForToday();
        decimal GetTotalPurchaseForWeek();
        decimal GetTotalPurchaseForMonth();
    }
}
```

Рисунок 4.2 – Інтерфейс замовлень

Для реалізації цих інтерфейсів використовуються відповідні класи репозиторії. Значним плюсом цього патерну проектування є те, що ми можемо дуже просто змінити використовуваний засіб збереження інформації без суттєвих змін в проекті. Приклад класу репозиторію можна побачити на рисунку 4.3:

```
using D2.Models;
using D2.Interfaces;
using D2.Models.Pages;
namespace D2.Repository
{

    public class CategoryRepository:ICategory
    {
        private ApplicationContext _context;

        public CategoryRepository(ApplicationContext context)
        {
            _context = context;
        }

        public IEnumerable<Category> GetAllCategories()
        {
            return _context.Categories;
        }

        public void AddCategory(Category category)
        {
            _context.Categories.Add(category);
            _context.SaveChanges();
        }

        public void UpdateCategory(Category category)
        {
            _context.Categories.Update(category);
            _context.SaveChanges();
        }

        public void DeleteCategory(Category category)
        {
            _context.Categories.Remove(category);
            _context.SaveChanges();
        }
        public PagedList<Category> GetCategories(QueryOptions options)
        {
            return new PagedList<Category>(_context.Categories, options);
        }
    }
}
```

Рисунок 4.3 – Клас репозиторій що описує взаємодію з даними сутності Категорії

Для налаштування роботи цих сервісів було потрібно зареєструвати їх в файлі Program.cs. Як це виглядало можна побачити на рисунку 4.4:

```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllersWithViews();



builder.Services.AddTransient<IMedicine, MedicineRepository>();
builder.Services.AddTransient<ICategory, CategoryRepository>();
builder.Services.AddTransient<IOrder, OrderRepository>();
builder.Services.AddSingleton<IHttpContextAccessor, HttpContextAccessor>();
builder.Services.AddSingleton<IWebHostEnvironment>(builder.Environment);

IConfigurationRoot _confString = new ConfigurationBuilder().
    SetBasePath(AppDomain.CurrentDomain.BaseDirectory).AddJsonFile("appsettings.json").Build();
builder.Services.AddDbContext<ApplicationContext>(options =>
               options.UseSqlServer(_confString.GetConnectionString("DefaultConnection")));
```

Рисунок 4.4 – реєстрація сервісів зберігання даних

4.2 Програмні рішення шару презентації

Шар презентації реалізується контролерами, які повертають дані користувачу. Приклад контролеру можна побачити на рисунку 4.5.

Рисунок 4.5 – Контролер що відповідає за домашню сторінку адміністратора

Атрибут `[Authorize(Policy = "RequireAdminRole")]` значить що для доступу до функцій цього контролера потрібно бути зареєстрованим адміністратором. На сторінці, що оброблює цій контролер початково відображаються всі ліки що додані до бази даних. Приклад функції цього контролеру, а саме, оновлення інформації про певні ліки можна побачити на рисунку 4.6.

```
[HttpPost]
public async Task<IActionResult> UpdateMedicine(Medicine medicine, IFormFile imageFile)
{
    if (imageFile != null)
    {
        string uploadsFolder = Path.Combine(_webHostEnvironment.WebRootPath, "images");
        string uniqueFileName = Guid.NewGuid().ToString() + "_" + imageFile.FileName;
        string filePath = Path.Combine(uploadsFolder, uniqueFileName);
        using (var fileStream = new FileStream(filePath, FileMode.Create))
        {
            await imageFile.CopyToAsync(fileStream);
        }
        medicine.ImUrl = "/images/" + uniqueFileName;
    }
    else
    {
        // Используем текущее значение ImUrl, если файл не выбран
        medicine.ImUrl = _medicine.GetMedicine(medicine.Id).ImUrl;
    }

    if (medicine.Id == 0)
    {
        _medicine.AddMedicine(medicine);
    }
    else
    {
        _medicine.UpdateMedicine(medicine);
    }
    return RedirectToAction(nameof(Index));
}
```

Рисунок 4.6 – Функція оновлення інформації про певні ліки

4.3 Програмні рішення шару стійкості

Для взаємодії з базою даних було вирішено використовувати підхід ORM з використанням бібліотекою Entity Framework Core та Identity з Entity Framework Core, для керування можливостями налаштування ролей користувачів та процесами авторизації та реєстрації.

Для нормальної роботи цієї бібліотеки потрібно створити класи моделей, що будуть описувати наші таблиці з даними. Приклад такого класу можна побачити на рисунку 4.7.

```
namespace D2.Models
{
    public class Medicine
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }
        public string NumReg { get; set; }
        public string ImUrl { get; set; }
        public decimal PurchasePrice { get; set; }
        public decimal RetailPrice { get; set; }
        public decimal CountInBox {  get; set; }
        public ReleaseForm ReleaseForm { get; set; }
        public int CountInShop {  get; set; }
        public int CategoryID {  get; set; }
        public Category Category { get; set; }

    }
}
```

Рисунок 4.7 – Приклад класу моделі, а саме клас, що описує ліки

Для створення контексту додатку потрібно створити окремий клас, та прописати там певні команди. Цей клас можна побачити на рисунку 4.8:

```
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
namespace D2.Models
{
    public class ApplicationContext: IdentityDbContext<User>
    {
        public ApplicationContext(DbContextOptions<ApplicationContext> context) : base(context)
        {

        }
        public DbSet<Medicine> Medicines { get; set; }
        public DbSet<Category> Categories { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<OrderLine> OrdersLine { get; set; }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            base.OnModelCreating(modelBuilder);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.Name);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.PurchasePrice);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.RetailPrice);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.NumReg);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.CountInShop);
            modelBuilder.Entity<Medicine>().HasIndex(e => e.ReleaseForm);
            modelBuilder.Entity<Category>().HasIndex(e => e.Name);
            modelBuilder.Entity<Category>().HasIndex(e => e.Description);
        }
    }
}
```

Рисунок 4.8 – Налаштування контексту додатку та бази даних

В цьому класі описується підключення всіх таблиць з бази даних, та формування індексів для пришвидшення виконання операцій пошуку, фільтації та сортування.

Також для налаштування нормального підключення до бази даних потрібно внести певні зміни у файл appsettings.json. Зміни що були внесені до цього файлу можна побачити на рисунку 4.9.

```
/json.schemastore.org/appsettings.json
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=HygeiaStore;Trusted_Connection=True;"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Information"
    }
  },
  "AllowedHosts": "*"
}
```

Рисукно 4.9 – Файл конфігурації підключення до бази даних

При наявності такої потреби цей файл може бути переписаний для переходу на окремий сервер бази даних що буде знаходитись у віддаленому місці. Але на поточний момент в цьому немає необхідності.

4.4 Програмні рішення шару бізнес логіки

Там де, це потрібно до моделей додана валідація. Приклад можна побачити на рисунку 4.10

```
using System.ComponentModel.DataAnnotations;

namespace D2.ViewModels
{
    Ссылок: 4
    public class RegisterViewModel
    {
        [Required]
        [Display(Name = "Email")]
        [DataType(DataType.EmailAddress, ErrorMessage = "Не коректний E-Mail")]
        Ссылок: 6
        public string? Email { get; set; }

        [Required]
        [Display(Name = "Рік народження")]
        Ссылок: 4
        public int Year { get; set; }

        [Required]
        [DataType(DataType.Password)]
        [Display(Name = "Пароль")]
        Ссылок: 4
        public string? Password { get; set; }

        [Required]
        [Compare("Password", ErrorMessage = "Паролі не співпадають")]
        [DataType(DataType.Password)]
        [Display(Name = "Підтвердити пароль")]
        Ссылок: 3
        public string? PasswordConfirm { get; set; }
    }
}
```

Рисунок 4.10 – Клас передачі даних при реєстрації з валідацією

Було створено окремі класи моделей для виконання пошуку та фільтації. Приклад такого класу можна побачити на рисунку 4.11:

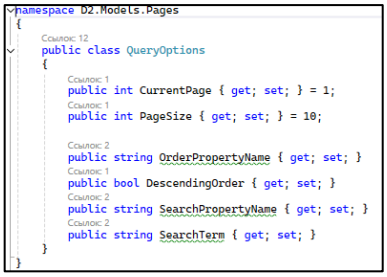


Рисунок 4.11 – Модель запиту для виконання фільтрації та пошуку

Ця ж сама модель використовується для підтримки пагінації. Але для цього інструмента використовується ще й клас, що відповідає за збереження інформації про список всіх сторінок. Частину його коду можна побачити на рисунку 4.12:

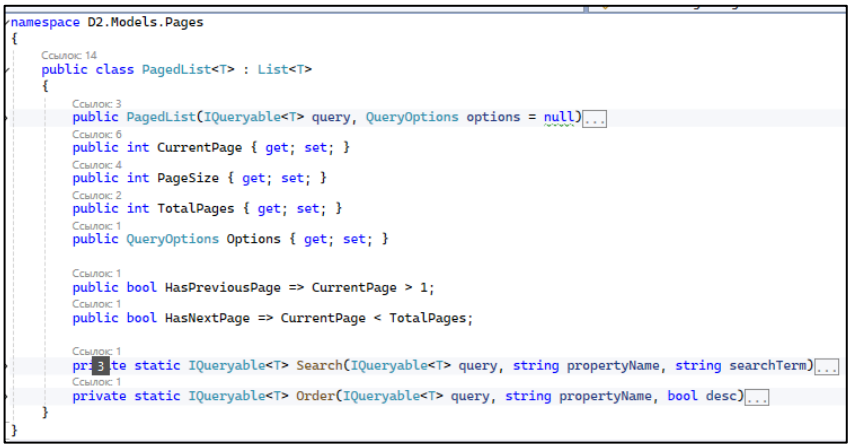


Рисунок 4.12 – Клас що описує список сторінок

Приклад реалізації пошуку можна побачити на рисунку 4.13:

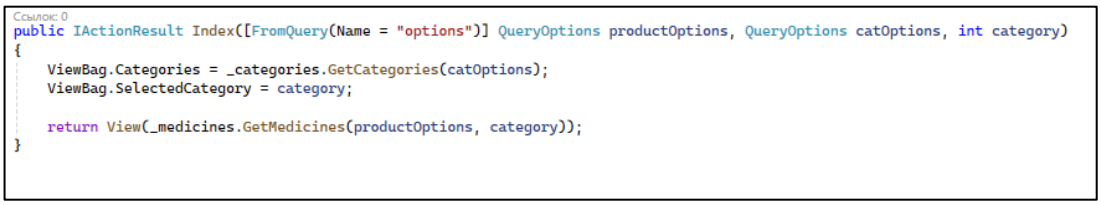
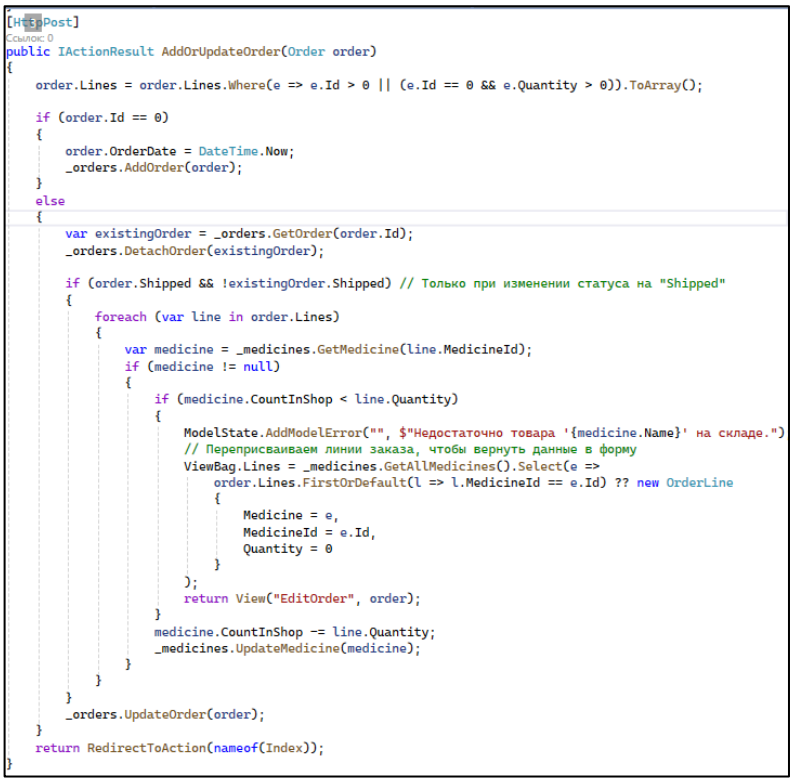


Рисунок 4.13 – Реалізація функції пошуку на контролері

Ще цікавим прикладом рішень шару бізнес логіки є автоматичне зменшення кількості товару на складі при встановлення прапорця відправлений у певному заказі. Приклад функції що це реалізує можна побачити на рисунку 4.14.

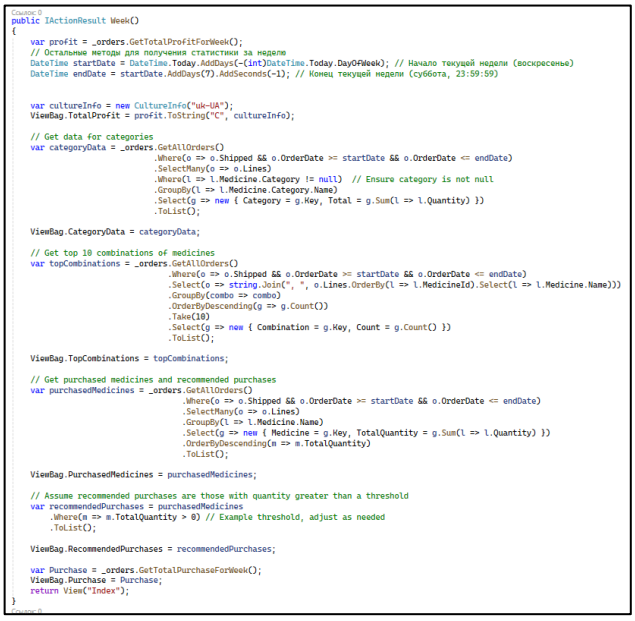


```
[HttpPost]
Ссылок: 0
public IActionResult AddOrUpdateOrder(Order order)
{
    order.Lines = order.Lines.Where(e => e.Id > 0 || (e.Id == 0 && e.Quantity > 0)).ToArray();

    if (order.Id == 0)
    {
        order.OrderDate = DateTime.Now;
        _orders.AddOrder(order);
    }
    else
    {
        var existingOrder = _orders.GetOrder(order.Id);
        _orders.DetachOrder(existingOrder);

        if (order.Shipped && !existingOrder.Shipped) // Только при изменении статуса на "Shipped"
        {
            foreach (var line in order.Lines)
            {
                var medicine = _medicines.GetMedicine(line.MedicineId);
                if (medicine != null)
                {
                    if (medicine.CountInShop < line.Quantity)
                    {
                        ModelState.AddModelError("", $"Недостаточно товара '{medicine.Name}' на складе.");
                        // Переприсваиваем линии заказа, чтобы вернуть данные в форму
                        ViewBag.Lines = _medicines.GetAllMedicines().Select(e =>
                            order.Lines.FirstOrDefault(l => l.MedicineId == e.Id) ?? new OrderLine
                            {
                                Medicine = e,
                                MedicineId = e.Id,
                                Quantity = 0
                            }
                        );
                        return View("EditOrder", order);
                    }
                    medicine.CountInShop -= line.Quantity;
                    _medicines.UpdateMedicine(medicine);
                }
            }
        }
        _orders.UpdateOrder(order);
    }
    return RedirectToAction(nameof(Index));
}
```

Рисунок 4.14 – Код функції оновлення заказу

Ця функція перевіряє чи достатньо ліків на складі, та якщо не достатньо – повертає інформацію про це на шар представлення та припиняє виконання.

Ще одним цікавим місцем нашої програми є формування статистики та рекомендацій. Приклад коду функції контролеру можна побачити на рисунку 4.15

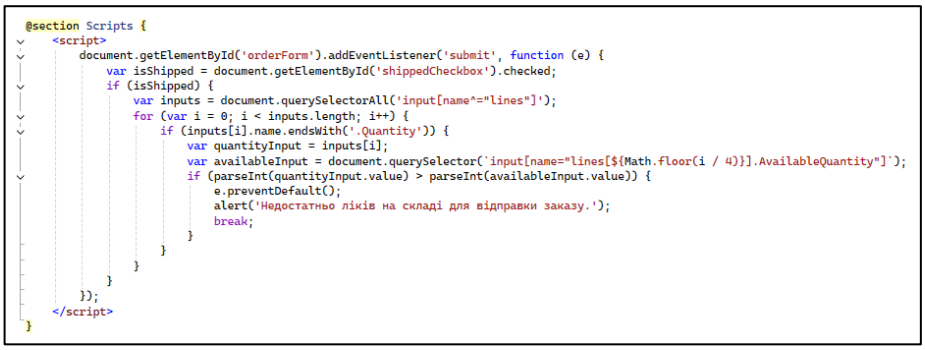


```
Ссылок: 0
public IActionResult Week()
{
    var profit = _orders.GetTotalProfitForWeek();
    // Остальные методы для получения статистики за неделю
    DateTime startDate = DateTime.Today.AddDays(-(int)DateTime.Today.DayOfWeek); // Начало текущей недели (воскресенье)
    DateTime endDate = startDate.AddDays(7).AddSeconds(-1); // Конец текущей недели (суббота, 23:59:59)


    var cultureInfo = new CultureInfo("uk-UA");
    ViewBag.TotalProfit = profit.ToString("C", cultureInfo);

    // Get data for categories
    var categoryData = _orders.GetAllOrders()
                              .Where(o => o.Shipped && o.OrderDate >= startDate && o.OrderDate <= endDate)
                              .SelectMany(o => o.Lines)
                              .Where(l => l.Medicine.Category != null)  // Ensure category is not null
                              .GroupBy(l => l.Medicine.Category.Name)
                              .Select(g => new { Category = g.Key, Total = g.Sum(l => l.Quantity) })
                              .ToList();

    ViewBag.CategoryData = categoryData;

    // Get top 10 combinations of medicines
    var topCombinations = _orders.GetAllOrders()
                                 .Where(o => o.Shipped && o.OrderDate >= startDate && o.OrderDate <= endDate)
                                 .Select(o => string.Join(", ", o.Lines.OrderBy(l => l.MedicineId).Select(l => l.Medicine.Name)))
                                 .GroupBy(combo => combo)
                                 .OrderByDescending(g => g.Count())
                                 .Take(10)
                                 .Select(g => new { Combination = g.Key, Count = g.Count() })
                                 .ToList();

    ViewBag.TopCombinations = topCombinations;

    // Get purchased medicines and recommended purchases
    var purchasedMedicines = _orders.GetAllOrders()
                                    .Where(o => o.Shipped && o.OrderDate >= startDate && o.OrderDate <= endDate)
                                    .SelectMany(o => o.Lines)
                                    .GroupBy(l => l.Medicine.Name)
                                    .Select(g => new { Medicine = g.Key, TotalQuantity = g.Sum(l => l.Quantity) })
                                    .OrderByDescending(m => m.TotalQuantity)
                                    .ToList();

    ViewBag.PurchasedMedicines = purchasedMedicines;

    // Assume recommended purchases are those with quantity greater than a threshold
    var recommendedPurchases = purchasedMedicines
        .Where(m => m.TotalQuantity > 0) // Example threshold, adjust as needed
        .ToList();

    ViewBag.RecommendedPurchases = recommendedPurchases;

    var Purchase = _orders.GetTotalPurchaseForWeek();
    ViewBag.Purchase = Purchase;
    return View("Index");
}
Ссылок: 0
```

Рисунок 4.15 – Функція формування набору даних для відображення статистики за 7 останніх днів.

4.5 Програмні рішення шару представлення

Робота деяких можливостей неможлива без додання додаткового коду, що буде виконуватись на стороні користувача. Прикладом такої функції є перевірка наявності на складі достатньої кількості ліків при встановленні прапорця відправлений. Що відбувається в контролері можна побачити в попередньому рисунку. На рисунку 4.16 представлено код скрипта що виконується на стороні клієнта.



```
@section Scripts {
    <script>
        document.getElementById('orderForm').addEventListener('submit', function (e) {
            var isShipped = document.getElementById('shippedCheckbox').checked;
            if (isShipped) {
                var inputs = document.querySelectorAll('input[name^="lines"]');
                for (var i = 0; i < inputs.length; i++) {
                    if (inputs[i].name.endsWith('.Quantity')) {
                        var quantityInput = inputs[i];
                        var availableInput = document.querySelector(`input[name="lines[${Math.floor(i / 4)}].AvailableQuantity"]`);
                        if (parseInt(quantityInput.value) > parseInt(availableInput.value)) {
                            e.preventDefault();
                            alert('Недостатньо ліків на складі для відправки заказу.');
                            break;
                        }
                    }
                }
            }
        });
    </script>
}
```

Рисунок 4.16 – Код сприпта на сторінці оновлення замовлення.

Другим прикладом наочним прикладом буде вивід інформації про існуючі в системі замовлення в певному порядку. Для цього застосовується код що можна побачити на рисунку 4.17.

```
@{
    ViewData["Title"] = "Всі замовлення";
}
@model IEnumerable<Order>

<h3 class="p-2 bg-primary text-white text-center">Замовлення</h3>
<div class="container-fluid mt-3">
    <div class="row row-cols-1 row-cols-md-2 row-cols-lg-3 g-4">
        @{
            var pendingOrders = Model.Where(order => !order.Shipped).ToList();
            var shippedOrders = Model.Where(order => order.Shipped).ToList();
        }

        @foreach (Order order in pendingOrders)
        {
            var cardClass = "bg-warning bg-opacity-25";
            <div class="col">
                <div class="card @cardClass">
                    <div class="card-body">
                        <h5 class="card-title">Замовлення #@order.Id</h5>
                        <p class="card-text"><strong>Назва:</strong> @order.CustomerName</p>
                        <p class="card-text"><strong>Zip:</strong> @order.ZipCode</p>
                        <p class="card-text"><strong>Всього:</strong> @order.Lines.Sum(e => e.Quantity * e.Medicine.RetailPrice - e.Medicine.PurchasePrice)</p>
                        <p class="card-text"><strong>Статус:</strong> Очікується відправка</p>
                    </div>
                    <div class="card-footer text-center">
                        <a asp-action="EditOrder" asp-route-id="@order.Id" class="btn btn-outline-primary">Редагувати</a>
                        <a asp-action="DeleteOrderId" asp-route-id="@order.Id" class="btn btn-outline-danger" onclick="return confirm('Ви впевнені, що бажаєте видалити це замовлення?');">Видалити</a>
                    </div>
                </div>
            </div>
        }

        @foreach (Order order in shippedOrders)
        {
            var cardClass = "bg-success bg-opacity-25";
            <div class="col">
                <div class="card @cardClass">
                    <div class="card-body">
                        <h5 class="card-title">Замовлення #@order.Id</h5>
                        <p class="card-text"><strong>Назва:</strong> @order.CustomerName</p>
                        <p class="card-text"><strong>Zip:</strong> @order.ZipCode</p>
                        <p class="card-text"><strong>Всього:</strong> @order.Lines.Sum(e => e.Quantity * e.Medicine.RetailPrice - e.Medicine.PurchasePrice)</p>
                        <p class="card-text"><strong>Статус:</strong> Відправлено</p>
                    </div>
                    <div class="card-footer text-center">
                        <a asp-action="EditOrder" asp-route-id="@order.Id" class="btn btn-outline-primary">Редагувати</a>
                        <a asp-action="DeleteOrderId" asp-route-id="@order.Id" class="btn btn-outline-danger" onclick="return confirm('Ви впевнені, що бажаєте видалити це замовлення?');">Видалити</a>
                    </div>
                </div>
            </div>
        }
    </div>
</div>
<div class="text-center mt-3">
    <a asp-action="EditOrder" class="btn btn-primary">Створити</a>
</div>
```

Рисунок 4.17 – Код представлення що відповідає за відображення інформації про існуючі замовлення

На цьому представлені організована кольорова диференціація замовлень на виконані та не виконані, та вивід замовлень у тому порядку, коли спочатку виводяться невиконані замовлення а потім – виконані.

Також місце яке не змогло обійтись без скриптів на представленні – то формування статистики. Приклад скрипту для формування діаграми можна побачити на рисунку 4.18

```
@section Scripts {
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
    <script>
        document.addEventListener('DOMContentLoaded', function () {
            var ctx = document.getElementById('categoryChart').getContext('2d');
            var categoryData = null;

        @if (ViewBag.CategoryData != null)
        {
            <text>
                    categoryData = @Html.Raw(Json.Serialize(ViewBag.CategoryData));
                console.log("Data from ViewBag.CategoryData:", categoryData);
            </text>
        }
        else @if (Model.CategoryData != null)
        {
         var categoryData = @Html.Raw(Json.Serialize(Model.CategoryData));
          //console.log("Data from Model.CategoryData:", categoryData);

        }


                if (categoryData != null) {
            var labels = categoryData.map(item => item.category); // Заменил item.Category на item.category
            var data = categoryData.map(item => item.total); // Заменил item.Total на item.total

            console.log("Labels:", labels);
            console.log("Data:", data);

            var chart = new Chart(ctx, {
                type: 'bar',
                data: {
                    labels: labels,
                    datasets: [{
                        label: 'Кількість покупок',
                        data: data,
                        backgroundColor: 'rgba(75, 192, 192, 0.2)',
                        borderColor: 'rgba(75, 192, 192, 1)',
                        borderWidth: 1
                    }]
                },
                options: {
                    scales: {
                        y: {
                            beginAtZero: true
                        }
                    }
                }
            });
        }
        else {
            console.log("No data available to render the chart.");
        }
        });
    </script>
}
```

Рисунок 4.18 – Скрипт для формування діаграми

Для формування діаграми застосовано бібліотеку chart.js.

4.6 Прийняті рішення щодо інтерфейсу системи

При вході користувач в систему його зустрічає головна сторінка аптеки. Тут зображуються всі доступні ліки. Також є можливість отримати відображення ліків певної категорії та шукати ліки за назвою. Зображення можна побачити на рисунку 4.19.

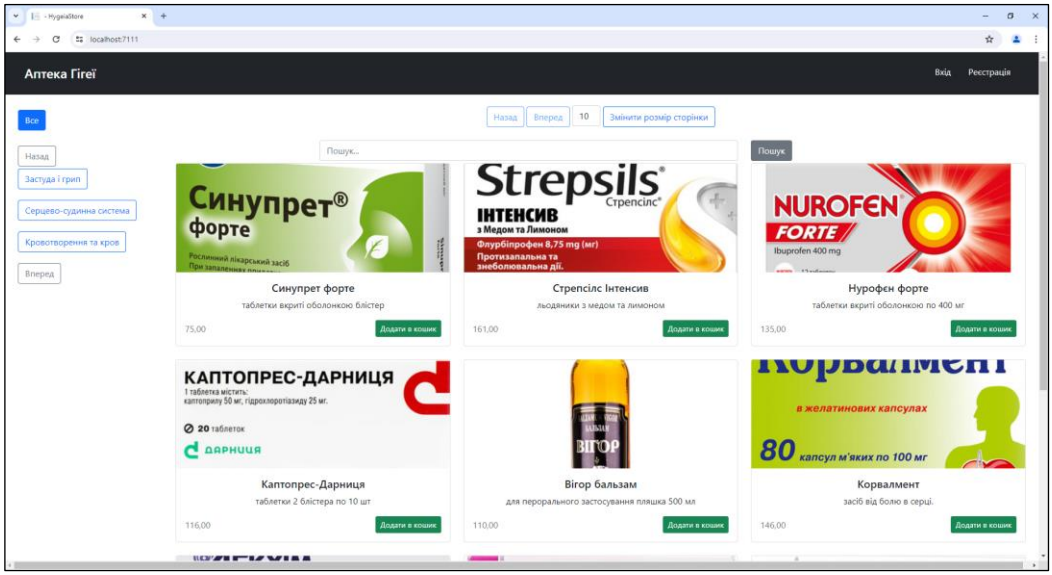


Рисунок 4.19 – Головна сторінка аптеки.

Всі категорії виводяться у списку зліва, без залежності від їх кількості. З гори сторінки є засоби навігації у вигляді налаштування кількості відображуваних на 1 сторінці ліків та власне, переходи між сторінками.

Зверху справа є посилання для входу та авторизації. Вікно авторизації можна побачити на рисунку 4.20.

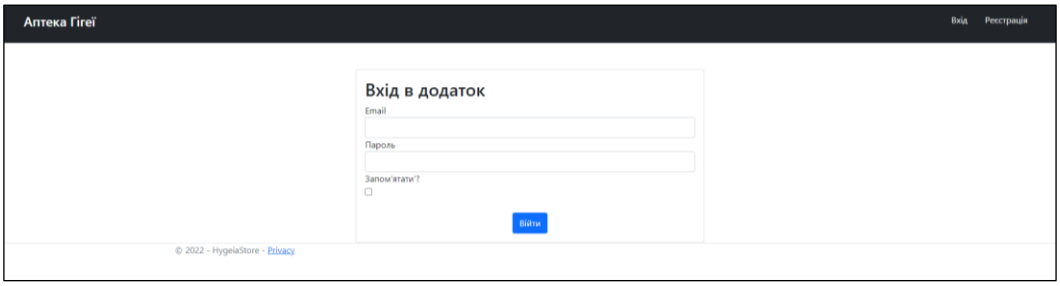


Рисунок 4.20 – Сторінка авторизації

Якщо користувач покладе ліки до кошика, то він зможе отримати доступ до сторінки кошика. Приклад такої сторінки можна побачити на рисунку 4.21

Рисунок 4.21 – Сторінка кошика

В кошику користувачу надається можливість керувати бажаною кількістю замовлюваних ліків та видаляти їх з кошика.

Коли щось додано в кошик інформація про поточну суму покупок виводиться у спеціальному компоненті головної сторінки. Як це виглядає можна побачити на рисунку 4.22.

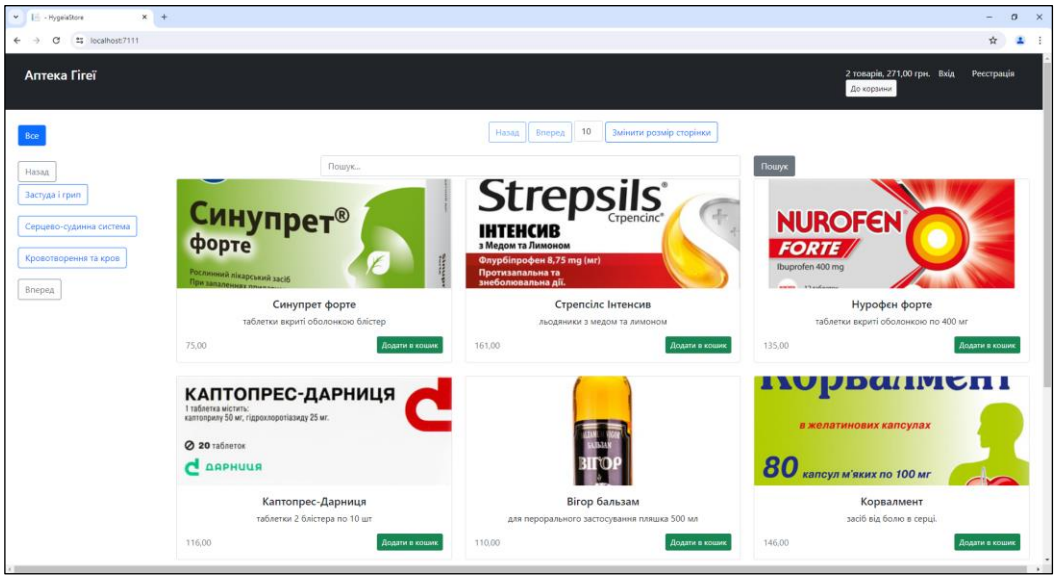


Рисунок 4.22 – Відображення спеціального компонента корзина.

При оформленні замовлення неавторизований користувач має ввести потрібну інформацію про адреси та пошту. Сторінку з оформленням замовлення можна побачити на рисунку 4.23.

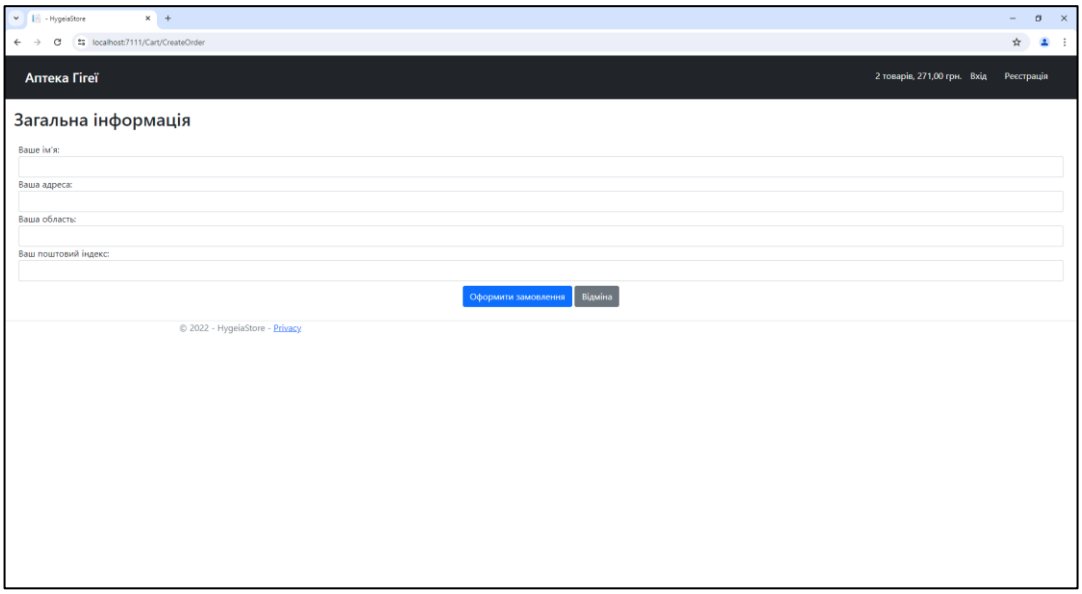


Рисунок 4.23 – Сторінка оформлення замовлення неавторизованого користувача

Якщо ж користувач авторизований – то ця інформація заповнюється автоматично. Приклад можна побачити на рисунку 4.24.

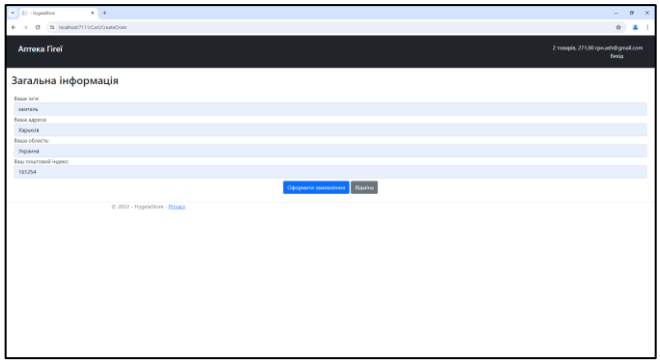


Рисунок 4.24 – Сторінка оформлення замовлення авторизованого користувача

Якщо користувач бажає внести зміни йому достатньо внести зміни на цій сторінці і вони автоматично зберігаються в базі даних.

Після внесення цієї інформації замовлення оформлено та користувач бачить наступну сторінку.

Приклад можна побачити на рисунку 4.25.

Рисунок 4.25 – Сторінка завершення оформлення замовлення

Якщо авторизований користувач має роль адміністратора – то у нього з’являється можливість потрапити до частини адміністратора. Головна сторінка цієї частини виглядає як на рисунку 4.26

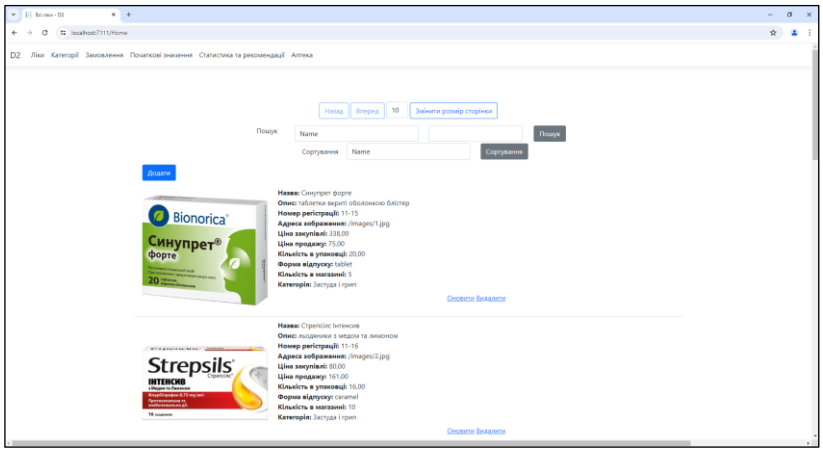


Рисунок 4.26 – Головна сторінка частини адміністратора

На цій сторінці одразу виводиться вся інформація про наявні в базі даних ліки. Надається можливість створювати нові та оновлювати та видаляти старі. Також є інструменти пагінації, пошуку та сортування. Користуючись верхнім меню можна потрапити до модулів керування категоріями, замовленнями, статистики та повернутись до частини аптеки.

Модуль керування категоріями товарів схожий на головний модуль Ліки. Його інтерфейс можна побачити на рисунку 4.27.

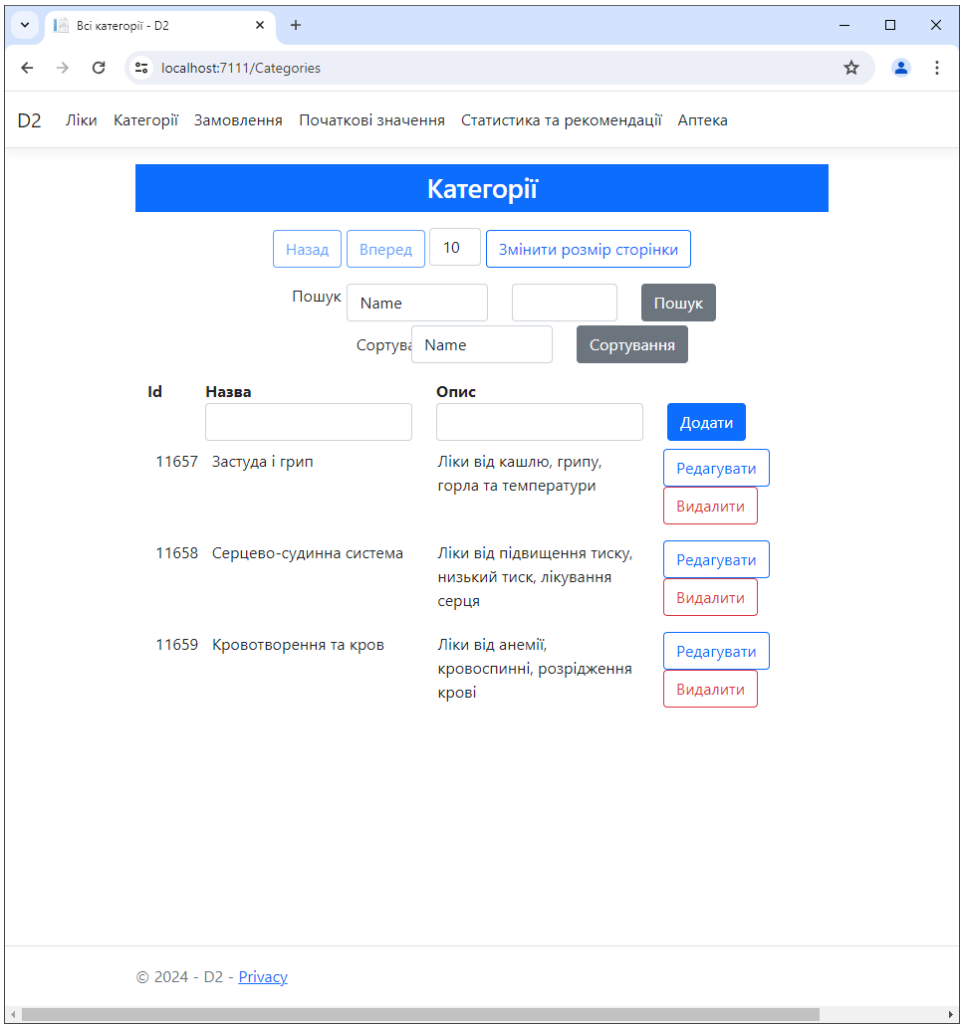


Рисунок 4.27 – Сторінка керування категоріями

На рисунку 4.28 можна побачити інтерфейс модуля керування замовленнями.

Тут спочатку виводяться картки не відправлених замовлень жовтого кольору, а потім вже відправлені зеленого. Також є в наявності можливість редагувати створені замовлення, видаляти їх та при необхідності створювати нові.

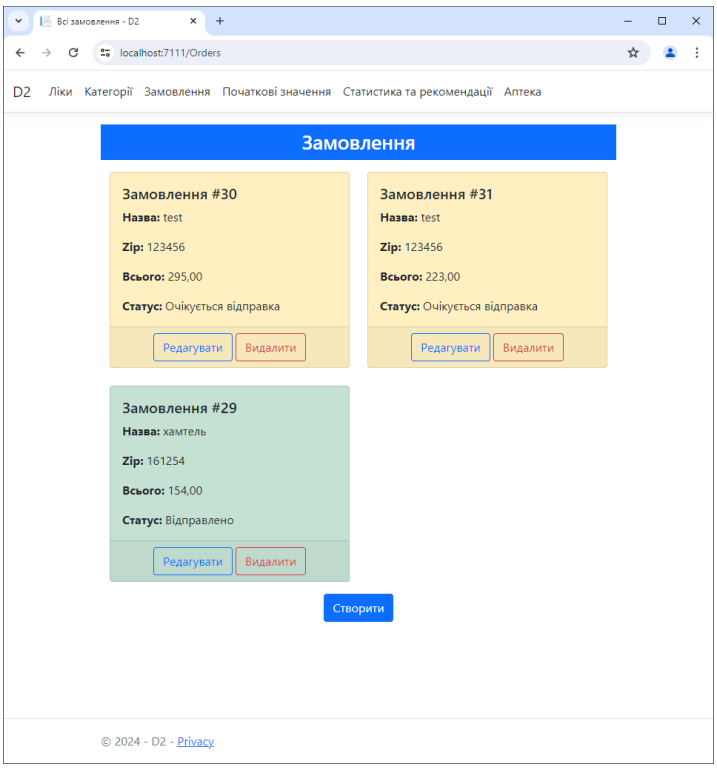


Рисунок 4.28 – Головна сторінка модулю керування замовленнями

Якщо спробувати позначити відправленим замовлення де вказано більше одиниць ніж є на складі буде видано повідомлення про похибку і оновлення не відбудеться. Приклад такої ситуації можна побачити на рисунку 4.29.

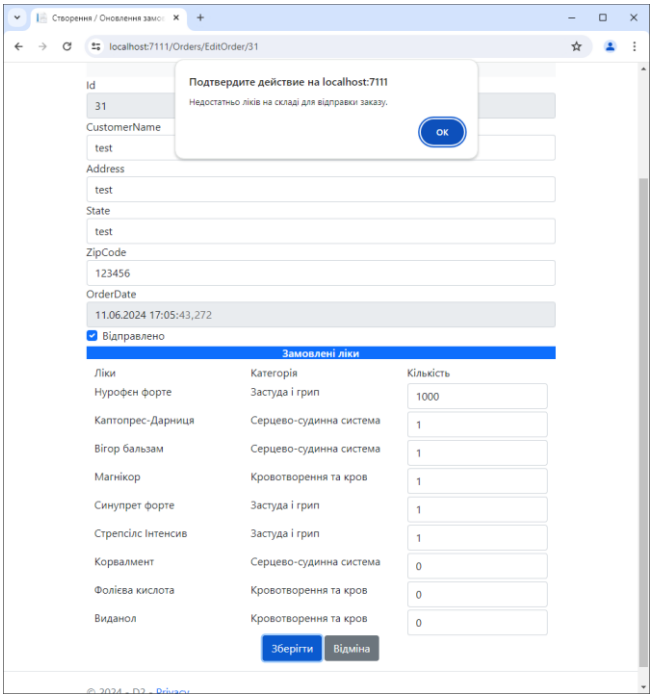


Рисунок 4.29 – Вікно редагування певного замовлення з повідомлення про помилку

Головна сторінка модуля статистики зображена на рисунку 4.30. На цій сторінці можна побачити загальний прибуток, що обраховується як різниця між потраченими грошима на закупівлю проданих ліків та ціною їх продажу, розподіл покупок за категоріями, найбільш часті комбінації ліків в замовленнях, загальні майбутні витрати, що формуються як сума поточних цін закупівлі вже проданих одиниць ліків, та список рекомендованих закупівель що формується аналогічним чином.

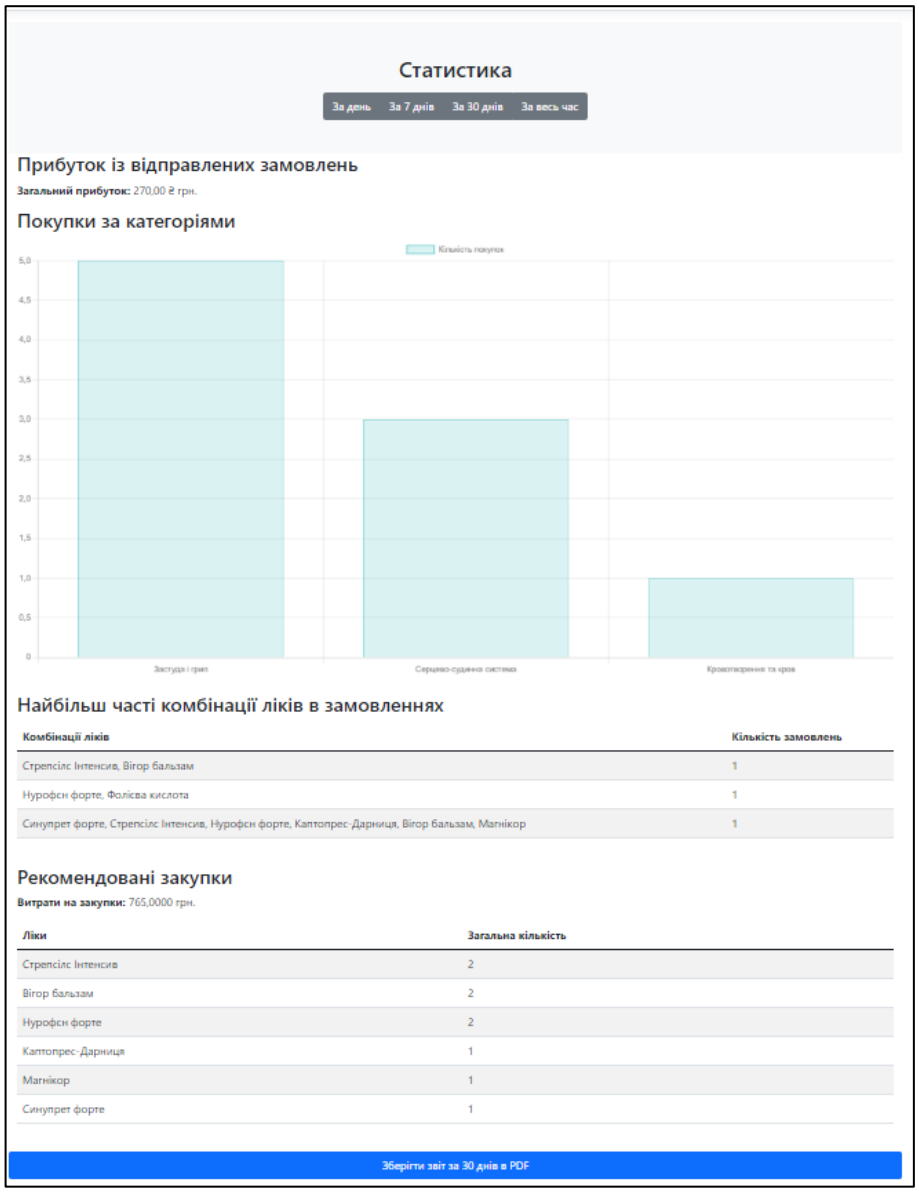


Рисунок 4.30 – Сторінка модулю статистики.

Також передбачена можливість формування звіту за останні 30 днів. Приклад такого звіту зображено на рисунку 4.31.

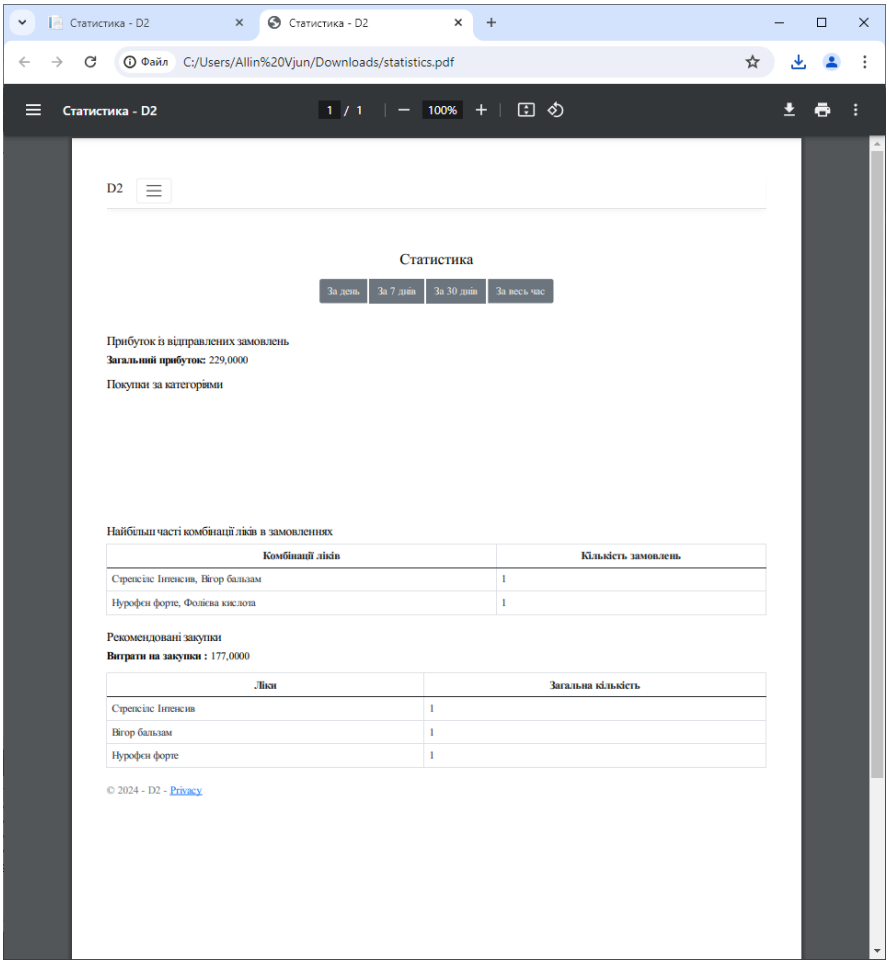


Рисунок 4.31 – Приклад сформованого звіту за 30 днів.

Для формування звітів використовується бібліотека Rotativa.

5 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Для тестування було обрано мануальне тестування. Процес перевірки та пошук помилок відбувався вручну, без застосування додаткових сторонніх тестів.

Також було використано самописний інструмент для наповнення бази даних великою кількістю значень. Цей інструмент застосовувався для перевірки інтрефейсу, чи зможе він відобразити коректно будь яку кількість категорій та ліків. Також він використовувався для тестування швидкості роботи методів пошуку та сортування. На базі його роботи було прийнято рішення про введення пагінації та індексування певних стовпців в базі даних. За роботу цього компоненту відповідає SeedController. Частину його коду можно побачити на рисунку 5.1.

```
namespace D2.Controllers
{
    Ссылок: 1
    public class SeedController:Controller
    {
        private readonly ApplicationContext _context;

        Ссылок: 0
        public SeedController(ApplicationContext context)
        {
            _context=context;
        } 3
        Ссылок: 3
        public IActionResult Index()
        {
            ViewBag.Count = _context.Medicines.Count();
            return View(_context.Medicines.Include(e=> e.Category).OrderBy(e=>e.Id).Take(20));
        }
        [HttpPost]
        Ссылок: 0
        public IActionResult CreateSeedData(int count)
        {
            ClearData();
            if (count > 0)
            {
                _context.Database.SetCommandTimeout(TimeSpan.FromMinutes(10));
                _context.Database.ExecuteSqlRaw("DROP PROCEDURE IF EXISTS CreateSeedData");
                _context.Database.ExecuteSqlRaw($@"
                CREATE PROCEDURE CreateSeedData
                @RowCount decimal
                AS
                BEGIN
                SET NOCOUNT ON
                DECLARE @i INT = 0;
                DECLARE @catId INT;
                DECLARE @CatCount INT = @RowCount / 10;
                DECLARE @pprice DECIMAL(5,2);
                DECLARE @rprice DECIMAL(5,2);
                DECLARE @rform INT = 1;
                DECLARE @cshop INT = 5;
                DECLARE @cbox INT = 20;
                DECLARE @nreg INT = 50;

                BEGIN TRANSACTION
                WHILE @i < @CatCount
                BEGIN
                INSERT INTO Categories (Name,Description)
                VALUES (CONCAT('Category-',@i),
                'Test Data Category');
                SET @catId = SCOPE_IDENTITY();
                DECLARE @j INT = 1;
                WHILE @j <= 10
                BEGIN
                SET @pprice = RAND() * (500-5+1);
                SET @rprice = (RAND() * @pprice) + @pprice;
                INSERT INTO Medicines(Name, Description,CategoryID,PurchasePrice,RetailPrice,CountInBox,CountInShop,NumReg,ReleaseForm,ImUrl)
                VALUES (CONCAT('Medicine' @i '-' @j) CONCAT('Description' @i '-' @j) @catId @pprice @rprice @cbox @cshop @nreg @i 'Medicine')
```

Рисунок 5.1 – Частина коду SeedController

По результатами прийнятих дій швидкість виконання запитів сортування та пошуку зросла приблизно на 20%. У певному прикладі з 93 мс до 71 мс. А час завантаження сторінки з загальними даними став майже непомітним.

Для початку мануального тестування нам потрібно створити план тестування, де зазначили мету(а саме перевірку роботи всіх доступних ззовні функцій) та список тестуємих функцій.

Першими функціями в тому плані були функції додавання, оновлення, видалення ліків. З деякими прикладами розроблених тест-кейсів можна ознайомитись у таблиці 5.1.

Таблиця 5.1 - Деякі приклади розроблених тест-кейсів

ID	Опис	Послідовність роботи	Викор. дані	Результат, що очікується	Результат, який було отримано	Пройдено / не пройдено
1	Створення ліків	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Натиснути кнопку «Додати», ввести дані і натиснути «Зберігти»	Назва= тест1 Опис= тест1 Номер регістрації= 1 Фото=1.jpg Ціна покупки= 10 Ціна продажу= 100 Форма відпуску =краплі Кількість у коробці = 20 Категорія =Застуда та грип	Форма закриється і лікарський засіб з’являєтся у загальному у списку	Форма закрилась і лікарський засіб з’явився в загальному у списку	Пройдено

Продовження таблиці 5.1

ID	Опис	Послідовність роботи	Викор. дані	Результат, що очікується	Результат, який було отримано	Пройдено / не пройдено
2	Оновлення катинки лікарського засобу	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Обрати який лікарський засіб потрібно оновити, натиснути на посилання «Оновити»	Фото = 2.jpg	Сторінка зачиняється та но головній сторінці модуля відображується лікрський засіб зі зміненою картинкою	Сторінка зачинилась. На головній сторінці модулю відобразилась змінена інформація.	Пройдено
3	Видалення лікарського засобу	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Обрати який лікарський засіб потрібно видалити. 4. Натиснути на посилання видалити	Назва = тест1	Сторінка оновлюється та лікарський засіб зникає	Сторінка оновилась та вибраний засіб видалився	Пройдено
4	Зробити пошук за номером регістрації	1. Авторизуватися користувачем маючим правами адміністратора	Номер= 11-1	Після натискання кнопки не з’являться всі	Відобразились лише запрошені ліки	Пройдено

Кінець таблиці 5.1

ID	Опис	Послідовність роботи	Викор. дані	Результат, що очікується	Результат, який було отримано	Пройдено / не пройдено
		2. Перейти в розділ «Ліки» 3. Обрати Номер реєстрації як поле для пошуку. 3. Написати частину номеру та натиснути на кнопку «Пошук»		лікарські засоби що містять таку частину в своєму номері реєстрації		

Під час проведення тестування було усунена помилка що виникала під час оновлення даних про ліки. Якщо не вказувати новий файл – старі дані не встановлювались і виникав системний збій. Помилка була усунута шляхом примусового запису старих даних якщо нові не поступили. Для цього було створено приховане поле на відображенні з записом старої інформації, і данні з нього додатково повертались до контролеру де порівнювались з інформацією з нового поля, а якщо там був null то ми записували інформацію з нового поля.

ВИСНОВКИ

Під час роботи були сформовані вимоги та розроблена перша версія веб-орієнтованої системи замовлення ліків «аптека Гігєї». Було проведено аналіз ринку з порівнянням провідних українських онлайн аптек, їх основних модулів, що можна виділити зі сторони, сформульовано вимоги до розроблюваної системи та виділено обмеження для розроблюваної системи. В ході аналізу було виділено функції що є у всіх розглянутих аналогів, та ті, що виділяють кожен з них. Була сформована постановка задачі.

В результаті проведення аналізу було запропоновано набір модулів для розроблюваної системи. Було спроектовано архітектуру програмної системи. Під час роботи розроблені UML діаграми, що демонструють головні особливості системи та особливості її розгортання. Були розроблені діаграми прецедентів, розгортання, ER діаграма для опису схеми бази даних, діаграма класів та активностей.

Були проаналізовані технології розробки схожих систем.

Для розробки були обрані наступні: середовище розробки MS Visual Studio 2022, мова програмування C#, фреймворки та бібліотеки ASP.NET Core, Entity Framework.

Для майбутніх релізів розглядається розширення модулю покупця, додавання можливості інших аптек використовувати наш сайт.

Таким чином розроблений продукт має буту корисним для користувачів з України.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. В. М. Толочко, Ю. П. Медведєва «СТАТУС ІНТЕРНЕТ-АПТЕК У ЧИННОМУ ЗАКОНОДАВСТВІ УКРАЇНИ», УПРАВЛІННЯ, ЕКОНОМІКА ТА ЗАБЕЗПЕЧЕННЯ ЯКОСТІ В ФАРМАЦІЇ, № 1 (3) 2009

2. Закон України від 03.09.2015 р. «Про електронну комерцію» // ВВР України, 2015 р., № 45, ст. 410.

3. Закон України від 17.09.2020 р. № 904 «Про внесення змін до статті 19 Закону України «Про лікарські засоби» щодо здійснення електронної роздрібної торгівлі лікарськими засобами», що дозволяє дистанційний продаж безрецептурних ліків.

4. ЗАКОН УКРАЇНИ від 04.04.1996 р. № 123/96-ВР «Про лікарські засоби»

5. Ahmed, S., & Christensen, B. T. (2009). An In Situ Study of Analogical Reasoning in Novice and Experienced Design Engineers. Journal of Mechanical Design, 131(11). https://doi.org/10.1115/1.3184693

6. https://liki24.com/uk/

7. https://biz.censor.net/resonance/3108229/kak_ukrainskie_itshniki_sozdali_servis_dostavki_lekarstv_po_nizkim_tsenam

8. https://apteka911.ua/ua

9. https://tabletki.ua/

10. https://www.rbc.ua/rus/styler/30-naypopulyarnishih-program-android-kimi-1700051032.html

11. Naylor D. et al. The cost of the" s" in https //Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies. – 2014. – С. 133-140.

12. Голощапова В. В. Сервер аналізу даних для аналітично-консультативного комплексу на основі технології СКБД MongoDB. – 2020.

13. Голобородько В. М., Опалєв М. Л. Формування дизайн-ергономічних вимог до користувацького інтерфейсу web-сайтів //Вісник Харківської державної академії дизайну і мистецтв. – 2011. – №. 3. – С. 16-19.

14. Bittner K., Spence I. Use case modeling. – Addison-Wesley Professional, 2003

15. Mistry R., Misner S. Introducing Microsoft SQL Server 2014. Microsoft Press, 2014. 144 c.

ДОДАТОК А

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

UNICHECK
by Turnitin

Ім'я користувача:
Олійник Олена Володимирівна каф. ПІ

ID перевірки:
1016350456

Дата перевірки:
12.06.2024 05:13:12 EEST

Тип перевірки:
Doc vs Library

Дата звіту:
12.06.2024 05:43:28 EEST

ID користувача:
100012353

Назва документа: 2024_Б_ПІ_ПЗПІ_Петько_І_С_скорочений

Кількість сторінок: 47 Кількість слів: 5659 Кількість символів: 42497 Розмір файлу: 2.95 MB ID файлу: 1016154128

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

4.31%
Схожість

Найбільша схожість: 0.97% з джерелом з Бібліотеки (ID файлу: 1016122186)

Пошук збігів з Інтернетом не проводився

4.31% Джерела з Бібліотеки 119 Сторінка 49

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

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

Підозріле форматування 24 сторінки

ДОДАТОК Б

Слайди презентації

1

Харківський національний університет радіоелектроніки
Кваліфікаційна робота бакалавра

Веб-орієнтована програмна система замовлення ліків «Аптека Гігеї»

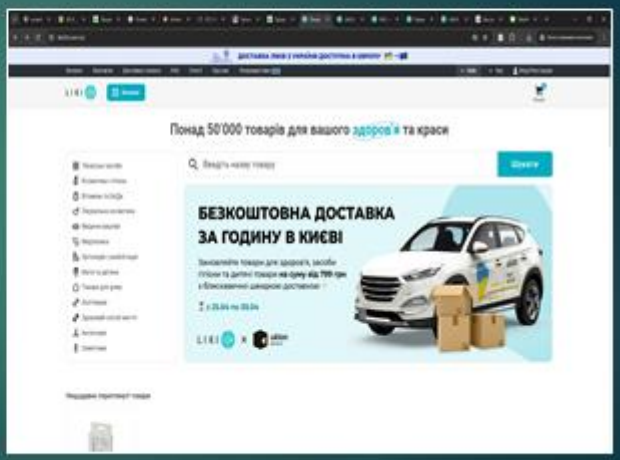
Виконав:
Ст. Гр. Пзпі-20-1
Петько ілля сергійович

Керівник:
Ст. Викл. Кафедри ІП
Олійник О. О.

2

Аналіз аналогів LIKI24

- Ліки24 не стільки інтернет аптека, скільки сервіс посередник між кур'єрами та клієнтами



Аналіз аналогів Аптека 911

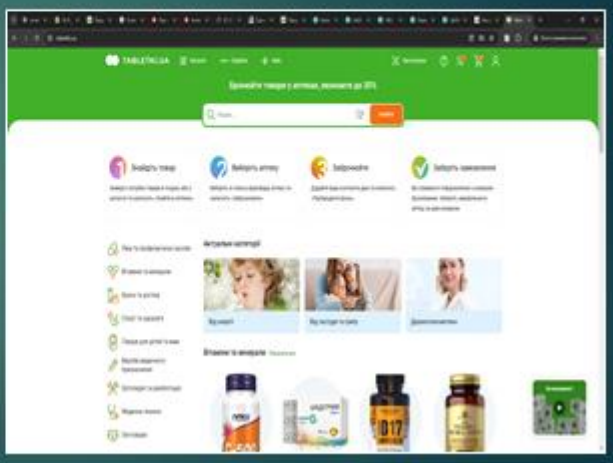
3

- Перший інноваційний майданчик для бронювання та організації доставки товарів аптечного асортименту

Аналіз аналогів TABLETKI.UA

4

- це онлайн-платформа, що надає велику інформацію про медичні препарати, їх інструкції та відгуки



Постановка задачі

5

Мета:

- Створення веб орієнтованої системи замовлення ліків «Аптека Гігєї».

Задачі:

- Система має надавати два інтерфейси користувачів: для покупця та для аптеки.
- Користувач має отримати можливість обирати товари з сайту та виконувати замовлення.
- Аптека має отримати можливість додавати та редагувати товари та категорії, бачити статистику по закупкам, інтерфейс керування замовленнями, інструмент формування звітів за 30 днів.

Формування вимог до програмної системи. Частина клієнта.

6

- відображення списку товарів;
- пошук та фільтрація товарів;
- можливість реєстрації та авторизації.

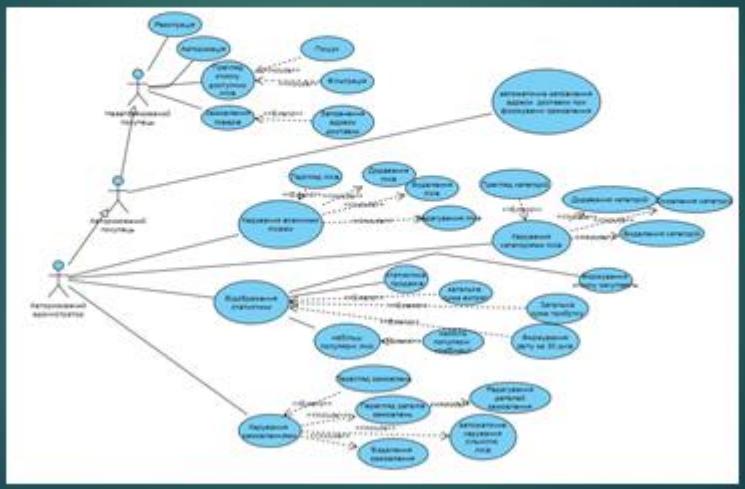
Формування вимог до програмної системи. Частина аптеки.

7

- механізм додавання та редагування товарів;
- відображення статистики по аптеці по покупкам та витратам;
- відображення статистики по найбільш популярним товарам;
- відображення статистики по найбільш популярним групам товарів;
- відображення поточних розходів та доходів за обраний проміжок часу;
- можливість формування pdf документу-звіту, з відображенням списку необхідних закупівель за останні 30 діб;
- можливість змінювати границі для відображення статистики: на день, 7 днів, 30 днів та весь час роботи аптеки;
- автоматичне форматування списку закупівель, як списку всіх відправлених товарів;
- автоматичне форматування замовлень;
- можливість редагування списку товарів;
- можливість видалення товарів;
- можливість додавання, редагування, видалення списку категорій;
- можливість автоматичного контролю наявності товару на складі перед відправкою замовлення;
- можливість автоматичного зменшення наявного на складі товару при відправленні замовлення.

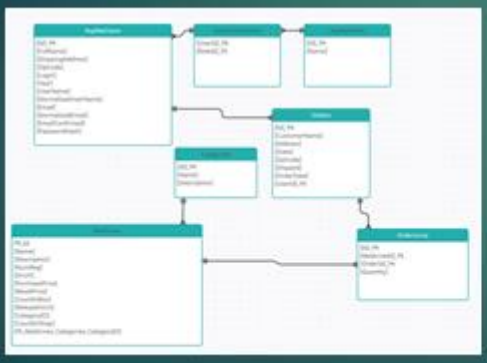
Архітектура та проектування. Use-case

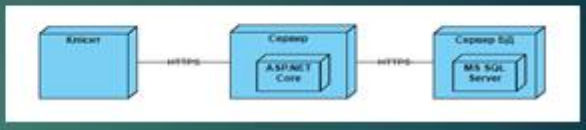
8



Архітектура та проектування. Діаграма розгортання та ER діаграма

9



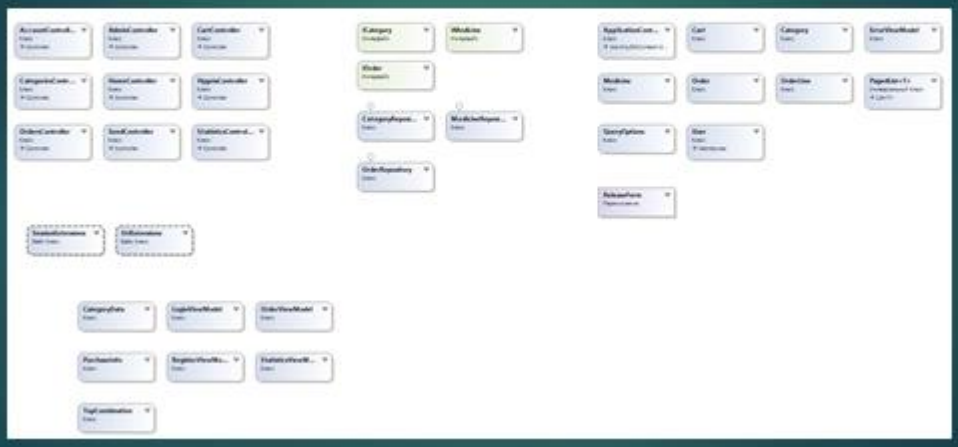


База даних складається з наступних таблиць:

- Ліки– зберігає дані про ліки що є в системі;
- Категорія– зберігає дані про категорії;
- Заказ– зберігає дані про закази;
- Рядок_заказу– слугує в'язучою таблицею між Заказами та Ліками;
- Користувач– зберігає дані про користувачів;
- Роль Користувача– слугує в'язучою таблицею між Користувачем та Ролями;
- Роль– зберігає дані про доступні ролі користувачів

Архітектура та проектування. Загальна діаграма класів.

10



Опис прийнятих програмних рішень

11

- було обрано мову C#, фреймворк ASP.NET Core, версію для створення веб додатків. Як середевище розробки використовувалась MS Visual Studio 2022,
- На слайді також представлені логотипи використаних бібліотек

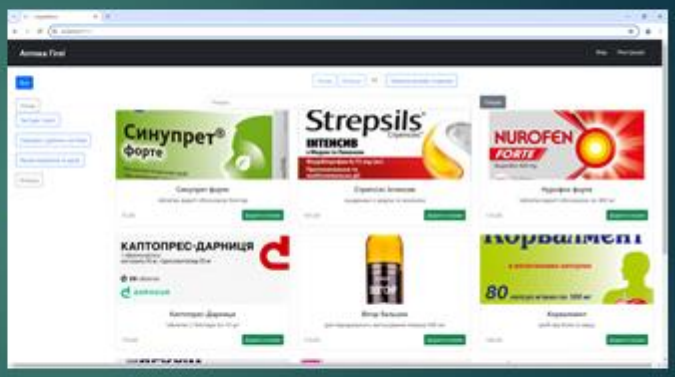
Опис прийнятих програмних рішень для взаємодії з базою даних

12

- Взаємодія виконується за допомогою EntityFrameworkCore;
- Таблиці бази даних описуються за допомогою класів моделей;
- Було використано патерн Репозиторій.

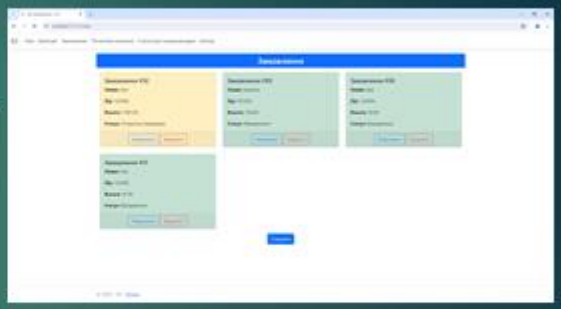
Опис прийнятих програмних рішень. Приклад інтерфейсу клієнтської частини проекту

13



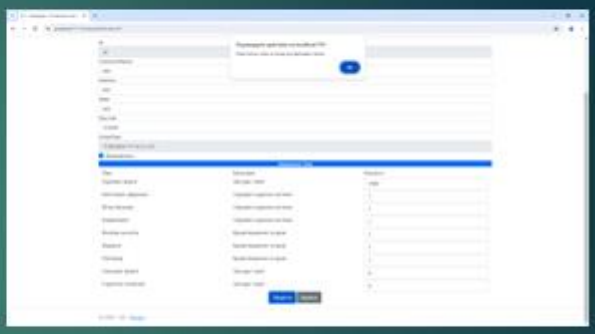
Опис прийнятих програмних рішень. Приклад інтерфейсу модулю керування замовленнями.

14



Опис прийнятих програмних рішень. Приклад повідомлення про недостачу ліків на складі.

15



Опис прийнятих програмних рішень. Приклад статистики.

16

Тестування розробленого програмного забезпечення.

17

- Для тестування було обрано мануальне тестування. Процес перевірки та пошук помилок відбувався вручну, без застосування додаткових сторонніх тестів.
- Також було використано самописний інструмент для наповнення бази даних великою кількістю значень. Цей інструмент застосовувався для перевірки інтрефейсу, чи зможе він відобразити коректно будь яку кількість категорій та ліків. Також він використовувався для тестування швидкості роботи методів пошуку та сортування.

Тестування розробленого програмного забезпечення.

18

ID	Опис	Послідовність роботи	Використані дані	Результат, що очікується	Результат, який було отримано	Пройдено / не пройдено
1	Створення ліки	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Натиснути кнопку «Додати», ввести дані і натиснути «Зберегти»	Назва=тест1 Опис=тест1 Номер реєстрації=1 Фото=1.jpg Ціна покупки=10 Ціна продажу=100 Форма відпуску=краплі Кількість у коробці=20 Категорія=Застуда та грип	Форма закриється і лікарський засіб з'явиться в загальному списку	Форма закрилась і лікарський засіб з'явився в загальному списку	Пройдено

ID	Опис	Послідовність роботи	Використані дані	Результат, що очікується	Результат, який було отримано	Пройдено / не пройдено
2	Оновлення лікарського засобу	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Обрати який лікарський засіб потрібно оновити, натиснути на посилання «Оновити»	Фото = 2.jpg	Сторінка зачиниться та на головній сторінці модуля відобразиться лікарський засіб зі зміненою картинкою	Сторінка зачинилась. На головній сторінці модулю відобразилась змінена інформація.	Пройдено
3	Видалення лікарського засобу	1. Авторизуватися користувачем маючим правами адміністратора 2. Перейти в розділ «Ліки» 3. Обрати який лікарський засіб потрібно видалити. 4. Натиснути на посилання видалити	Назва = тест1	Сторінка оновиться та лікарський засіб зникне	Сторінка оновилась та вибраний засіб видалився	Пройдено
4	Зробити пошук за номером	1. Авторизуватися користувачем	Номер=11-1	Після натискання кнопки	Відобразилась лише запрошена	Пройдено

Висновки

19

Під час роботи були сформовані вимоги та розроблена перша версія веб-орієнтованої системи замовлення ліків «аптека Гігєї».

Готовий продукт буде зручним та корисним для використання малими компаніями.

Наступні релізи:

- Розширити клієнтську частину за рахунок зручного модулю підбору аналогів.
- Додати більше статистики для аптеки.
- Додати можливість підключення інших аптек до мережі.

20

Дякую за увагу!