Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра мультимедійних систем факультету інформатики

«Розробка веб-застосунку на .NET»

Текстова частина до курсової роботи
за спеціальністю «Комп'ютерні науки» 122

Керівник курсової роботи
старший викладач Борозенний С.О.
_____
(підпис)
"_ __ _" _____ 2020 р.

Виконав студент Шевченко В.О.
"_ __ _" _____ 2020 р.

Київ 2020

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра мультимедійних систем факультету інформатики

ЗАТВЕРДЖУЮ
старший викладач
_____ Борозенний С.О.
(підпис)
„____”_____2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на курсову роботу

студентці Шевченко Владиславі

3-го курсу факультету інформатики

ТЕМА: Розробка веб-застосунку на .NET

Вихідні дані:

-
-

Зміст ТЧ до курсової роботи:

Вступ
Анотація
1. Аналіз предметної області
2. Вибір програмного забезпечення
3. Реалізація практичної частини

Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі „___” _____ 2020 р. Керівник _____
(підпис)
Завдання отримав _____
(підпис)

Календарний план виконання курсової роботи

Тема: Розробка веб-застосунку на .NET

Календарний план виконання роботи:

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу	листопад 2019 р.	
2.	Огляд літератури за темою роботи	листопад-грудень 2019 р.	
3.	Вибір програмного забезпечення для роботи	січень 2020 р.	
4.	Розробка веб-застосунку (практичної частини роботи)	лютий-березень 2020 р.	
5.	Написання текстової частини роботи	квітень 2020 р.	
6.	Підготовка слайдів для презентації та доповіді	квітень 2020 р.	
7.	Надання роботи науковому керівнику для перевірки	квітень 2020 р.	
8.	Корегування роботи за результатами попереднього захисту.	квітень 2020 р.	
9.	Остаточне оформлення пояснювальної роботи та слайдів.	квітень-травень 2020 р.	
10.	Захист курсової роботи	18-29 травня 2020 р.	

Студентка Шевченко В.О._____

Керівник Борозенний С.О._____

“_____” _____

Зміст

Вступ

З березня 2020 року в Україні діє карантин через пандемію коронавірусу (COVID-19). До переліку обмежень входить заборона на присутність в аптеці понад однієї людини на 10 кв.м. торгової площі, крім персоналу аптечного закладу. Через це люди вимушені чекати на вулиці у черзі та заходити здебільшого по-одному, адже більше не дозволяє площа торгової зали.

Одним з можливих рішень даної проблеми, а в подальшому оптимізації роботи аптеки, — є створення онлайн-сервісу аптеки, де покупці матимуть можливість подивитися наявність ліків у конкретній аптеці, забронювати їх та забрати у визначений час, до моменту якого працівники закладу вже підготують та упакують ліки, а покупцю залишиться лише забрати ліки та їх оплатити. Це полегшить життя як працівників медичного закладу, так і покупців, бо зменшить черги та ризики зараження вірусом.

Крім цього, для клієнтів буде доступна послуга доставки ліків кур’єром, що особливо актуально для невеликих міст та сіл, в аптеках яких можуть бути відсутні деякі препарати, необхідні для мешканців.

В Україні існують великі сайти для онлайн-бронювання в різних аптеках міста, але всі вони беруть велику комісію з аптек за користування. Тому ними можуть користуватися лише великі мережі аптек зі значним бюджетом.

Ми у свою чергу пропонуємо сервіс з базовим набором необхідних функцій, який буде простим у користуванні для споживачів, в тому числі для людей похилого віку (є однією з найбільших груп ризиків), та для будь-якої аптеки навіть у маленькому місті, яка зможе підключити сервіс та ним користуватися.

Анотація

Курсова робота присвячена створенню веб-застосунку для бронювання лікарських засобів в аптеці з використанням фреймворку ASP.NET Core від компанії Microsoft, архітектурного патерну MVC (Model-View-Controller) та бази даних MySQL.

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Аналіз наявних сервісів

Великі мережі аптек, наприклад як Аптека Низьких цін або Доброго Дня, мають власні онлайн-сайти, на яких можна замовити ліки та доставку кур'єром у великих містах або Новою Поштою. Такі способи стають у нагоді, але не дозволяють забезпечити необхідними ліками людей, які мешкають у маленьких містах та селах і не мають наявних можливостей для отримання ліків.

Ще одним конкурентом для нашого сервісу є такі потужні портали онлайн-бронювання ліків, як tabletli.ua та medbrowse.com, які дозволяють бронювати ліки та купувати їх в аптеках. Проте, цей сервіс покриває лише великі міста та села, а також не дозволяє організувати доставку.

Також існує можливість доставки ліків з аптеки за допомогою сервісу доставки Glovo. На час карантину вони знизили вартість доставки зі 60 грн до 1 грн, але Glovo працює лише в 11 великих містах.

Тому розробка нашого сервісу є актуальною для аптек, які працюють у невеликих містах та не входять до великих мереж.

1.2. Постановка задачі

Мета розробки — створення онлайн-сервісу з бронювання лікарських засобів для аптек, щоб оптимізувати процес продажу товарів в умовах пандемії через коронавірус, зменшити час, витрачений на обслуговування одного клієнта, розробити систему бонусів для покупців та знизити можливі ризики захворювань.

Переваги онлайн-сервісу:

- Веб-застосунок працюватиме цілодобово, тому покупець матиме можливість дізнатися про наявність та забронювати ліки у зручний час, а потім за кілька хвилин забрати їх в аптеці;
- Створення бази постійних клієнтів, які реєструватимуться на сайті та у

подальшому отримуватимуть знижки під час покупки ліків;

- Продаж рекламних площ онлайн-сервісу, як додаткове джерело доходу для маленького підприємства;
- Підтримка послуги кур’єрської доставки, що забезпечить збільшення кількості покупців, а також дозволить підвищити впізнаваність бренду для можливого подальшого розширення;
- Зменшення ризику зараження вірусами завдяки доставці кур’єром або зменшенню часу купівлі, бо провізор в аптеці заздалегідь підготує необхідні ліки;
- Після закінчення епідемії онлайн-сервіс дозволить зменшити черги в аптеці, що призведе до збільшення кількості покупців та більшого прибутку.

1.3. Що має бути доступно у веб-застосунку

Веб-застосунок має містити кілька інтерфейсів: для менеджера аптеки, зареєстрованого покупця та просто відвідувача сайту. Всі вони мають власний доступний набір функціоналу. Доступ до інтерфейсу визначається за допомогою введення логіна та пароля.

Менеджер аптеки має можливість переглядати всі доступні, шукати ліки за категоріями, додавати нові надходження зі складів, дивитися історію замовлень покупців.

Зареєстрований покупець може переглядати всі доступні ліки, шукати ліки за категоріями, бронювати ліки, замовити доставку кур’єром.

Інтерфейс відвідувача найпростіший. Він має можливість переглядати всі доступні ліки, шукати ліки за категоріями та йому доступна функція реєстрації на сайті. Послуга бронювання ліків доступна лише зареєстрованим клієнтам.

Сторінка з інформацією про ліки має містити: назву препарату, короткий опис, категорію препарату, кількість ліків в упакуванні, кількість діючої речовини, виробника та вартість за одне упакування. Також кожний лікарський засіб має супроводжуватися фотографією упакування.

Загалом веб-застосунок має бути оформлений у простому стилі, щоб бути зрозумілим для людей похилого віку та не відволікати клієнтів безпосередньо від бронювання.

2. ВИБІР ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. ASP.NET

Для розробки веб-застосунку була обрана платформа ASP.NET Core з використанням патерну MVC та базою даних MySQL.

Платформу ASP.NET створила компанія Microsoft у 2002 році для розробки веб-застосунків за допомогою IIS (Internet Information Services). Крім ASP.NET, для створення веб-застосунків використовують такі технології, як Node.JS, Spring. Проте для розробки онлайн-сервісу ми обрали ASP.NET за такі основні переваги[1]:

- простіша підтримка порівняно з PHP Framework;
- краща інтеграція з сервісами Microsoft;
- застосунки, розроблені за допомогою ASP.NET, можна відкривати у будь-яких браузерах на всіх пристроях (смартфони, планшети, ноутбуки, ПК);
- підтримка UNICODE, що дозволяє використовувати більшість національних мов;
- можливість підключення та використання великої кількості готових сервісів;
- використання типових шаблонів автентифікації та авторизації юзера;
- більш дешева вартість.

Оскільки ми використовуємо платформу ASP.NET, то основною мовою для написання застосунків на ній є C#.

- C# підтримує об’єктно-орієнтовану парадигму;
- C# має схожий інтерфейс на мови C/C++, що дозволяє легший перехід з C/C++ на C#;
- На C# реалізовано безліч фреймоврків, які входять у платформу .NET;
- В C# не є таким масштабним використання указників, адресної арифметики, розіменування, які за своїми властивостями є

небезпечними, бо мають прямий доступ до пам’яті.

ASP.NET Core MVC працює на основі вбудованого шаблону MVC (model-view-controller). Він дозволяє керувати формою веб-застосунку та взаємодіяти з компонентами, які містяться у застосунку.

Вперше модель MVC (model - view - controller) була придумана у 1970-х роках компанією Xerox для організації ранніх додатків з графічним інтерфейсом.

Шаблон MVC уявно ділить веб-застосунок на три частини:

1. Контролер реалізується у вигляді класу, що є проміжним етапом обробки та подання даних між користувачем та безпосередньо даними (база даних або програма). Дані, які вводить користувач, обробляються контролером, та повертаються у вигляді результату (view).
2. View — графічна частина застосунку, яку бачить користувач та з якою взаємодіє, зазвичай як html-сторінка.
3. Модель — клас, який реалізує методи для представлення та обробки даних [1].

Завдяки використанню паттерну MVC, структура проєкту розподіляється на окремі компоненти. Розділення сприяє тому, що окремий компонент виконує виключно свої операції. Таким чином, тестування та відладка (debugging) відбуваються не цілого проєкту, а окремих його частин. Це покращує реалізацію кожної частини програми та підвищує швидкість знаходження помилок при їхній наявності [2].

Наприклад, якщо ми хочемо протестувати базу даних, нам не потрібно тестувати весь проєкт, а достатньо протестувати лише контролер. Або при тестуванні frontend-частини проєкту буде тестуватися лише view.

Загалом, ASP.NET Core побудована на основі незалежних компонентів з характеристиками, які чітко визначені. Зручність даної платформи полягає в тому, що всі головні компоненти можна замінювати на інші компоненти, які були реалізовані власноруч.

Зокрема, ASP.NET Core MVC пропонує кілька варіантів:

- Створення підкласу стандартної реалізації, щоб мати можливість коригування;
- Заміна компонента власною реалізацією інтерфейсу;
- Використання реалізації компонента у стандартному вигляді [1].

Як середовище програмування була використана Visual Studio 2019 від компанії Microsoft, бо це є функціональним IDE якраз для розробки застосунків на платформі .NET.

2.2. База даних

Для зберігання даних обрали даних MySQL, яка наразі є однією з найвідоміших та найпопулярніших завдяки своїй швидкості та розповсюдженням під безкоштовною ліцензією. Вона підтримує стандарт мови SQL, складні структури даних та великий спектр вбудованих типів даних. До того ж одним з основних призначень використання цієї бази даних є якраз створення веб-сторінок [4].

2.3. Entity Framework

Також для роботи з даними під час розробки використовували технологію Entity Framework, яка працює з поняттям сутності та дозволяє працювати з об’єктами, а не таблицями. Для використання Entity Framework з MySQL ми повинні встановити бібліотеку MySQL.Data.EntityFramework

Інфраструктура Entity Framework Core являє собою обгортку, яка зв’язує базу даних з об’єктно-орієнтованою парадигмою, створюючи віртуальну об’єктну базу даних. Вона була створена корпорацією Microsoft для розробників на платформі Net.Core, щоб дозволити їм зберігати дані у застосунках в реляційних базах даних.

Основним завданням Entity Framework Core є зберігання об’єктів у базі даних, доступу до них та забезпечення ніби зв’язку між базою даних та застосунком ASP.NET Core [3].

Entity Framework використовує реляційну базу даних, у якій дані представляють у вигляді таблиць з рядками. Запити до реляційної бази даних пишуться на мові SQL. Проте, сервери баз даних використовують діалекти

мови SQL, які відрізняються одна від одної при звертанні до даних. Таким чином, Entity Framework дозволяє нам перетворити об’єкти у прийнятну форму для зберігання у таблицях баз даних та реалізовувати виконання запитів до цих даних [3].

Щоб сервери могли працювати з різними даними, Entity Framework Core покладається на версію сервера, який реалізовує запити до баз даних. Щоб отримати об’єкт, .NET Entity Framework Core реалізує SQL-команду, яка створює запит до серверу бази даних на отримання даних, які представляють собою об’єкт, поля якого заповнюють поля об’єкта .Net.

Для роботи з колекціями Entity Framework Core використовує вбудовану у платформу .Net мову LINQ, яка реалізовує маніпуляції з колекціями об’єктів .Net.

Перевагами використання Entity Framework Core є [3]:

- відкритий sourse-code;
- можливість безпосереднього створення бази даних без прямого написання коду, тобто з наявних сутностей класів Entity Framework Core самостійно створює базу даних за їх кодом;
- Entity Framework Core не має прив’язки та не залежить від релізів фреймворку .NET;
- можливість працювати з будь-якою реляційною базою даних та чинним провайдером Entity Framework;
- можливість генерації команд мови SQL із мови LINQ в сутності;
- безпосередня підтримка параметризованих запитів;
- можливість виконувати маніпуляції з командами: вставка, видалення, оновлення;
- підтримка роботи не лише із вбудованими даними, а також із програмованими класами;
- можливість безпечної компіляції завдяки перевірці синтаксичних помилок і безпеки типів;
- підтримка роботу з процедурами.

2.4. Razor Pages

Razor Pages у паттерні MVC являє собою view або іншими словами типову візуалізацію інформації з бази даних для користувача. Razor Pages реалізована як стандартна html-розмітка з використанням стилів (CSS, bootstrap) та відображенням інформації з моделі [5].

Відображення інформації на сторінці завдяки розмітці та backend-класу з'явилося відносно нещодавно. Таким чином, це нововведення нагадує використання WebForms, але без підтримки збереження стану.

Очевидно, що підхід такого рішення (відображення змісту завдяки розмітці та backend-класу), має перевагу в тому, що зникає необхідність у прошарку моделі сторінки (модель даних). Як результат, внутрішня частина реалізації механізму зображення сторінки являє собою як контролер, так і модель. Тобто, підтримує класичну парадигму об'єктно-орієнтованого програмування, а саме: інкапсуляцію даних і методів роботи з ними в одному об'єкті [5].

Врешті-решт, модель, що відображає сторінку — це просто клас. Тому відсутні будь-які причини, чому цим класом не може бути частина шаблону MVC (model-view-controller), а саме — контролер.

У результаті Razor Pages являє собою краще рішення для написання клієнт-серверних застосувань, які реалізовуються за допомогою паттерна MVC. Тому що технологія Razor Pages створює та використовує традиційне та цілком інтуїтивно зрозуміле поняття "сторінки", а не прийняті в паттерні MVC (model-view-controller) моделі та контролери, які можуть бути розкидані по всьому проєкту, що ускладнює відладку, тестування та підтримку візуальної частини проєкту.

Однією з характерних особливостей Razor Pages є те, що сторінка представляє собою блочну систему, тобто будь-які дані, що візуалізуються користувачу, реалізовані в окремому файлі з розширенням .cshtml. Таким чином, навігація відтворюваних даних стає зрозумілішою і краще піддається підтримці [5].

Також характерною рисою Razor Pages є те, що код, написаний мовою C#, можна реалізувати безпосередньо у файлі розмітки Razor Pages, тобто у файлі з розширенням .cshtml. Завдяки цій особливості полегшується візуалізація даних, які представлені у вигляді колекцій об'єктів, що обробляються моделями та зберігаються в базі даних.

3. Реалізація практичної частини

3.1. Структура проєкту

У загальному проєкт, написаний на фреймворку Asp.NET Core MVC (model-view-controller) складається з:

- двох основних файлів, які мають назву Program.cs та Startup.cs;
- різноманітних папок та підключених сервісів.

Для кращого розуміння розглянемо їх детальніше:

Базова структура:

- Program.cs — точка входу в програму (проєкт), написана на фреймворку Asp.NET Core MVC (model-view-controller);
- Startup.cs — файл використовується для того, щоб підключати сервіси для коректної роботи, написані на фреймворку. Також цей файл використовується для налаштування коректної програми сторінками розробленого сайту.

Папки:

- Wwwroot слугує для підключення каскадної таблиці стилів (CSS), а також для зберігання візуальних зображень, посилання на які зберігаються в база даних проєкту;
- Controllers — папка, яка зберігає усі контролери проєкту, тобто .cs класи, що використовуються для повернення візуальної частини проєкту (view);
- Data — папка, яка містить основну частину класів, що взаємодіють з базою даних, моделі, чиї сутності зберігаються в базі даних, а також інтерфейси моделей;
- Migrations — папка, яка зберігає міграції при оновленні бази даних, а також інформацію про саму базу даних;
- ViewModels — папка, яка зберігає класи для візуалізації інформації, взятої з абзиданих про моделі, які там зберігаються;
- Views — папка, яка зберігає в собі файли, а також інші папки, які класифікують розбиття візуальної частини проєкту.Файли, що

представлені у цій папці фактично являються файлами Razor Pages з розширенням .cshtml, тобто дана папка уособлює собою частину шаблону MVC, а саме view або візуальну частину.

Оскільки Razor Pages має блочну систему, то кожен підблок загального блоку містить в окремій папці, яка в свою чергу міститься у папці View, тобто папка View містить, окрім загальних шаблонів сторінок, безліч папок.

3.2. Розробка бази даних

База даних онлайн-аптеки складається з восьми таблиць.

3.2.1. Реалізація бази даних

Таблиця Order містить контактні дані покупця, надані під час замовлення а також безпосередньо інформацію про замовлення.

Поле	Тип	Опис
id	int	Унікальний ідентифікатор в базі даних
name	varchar(30)	Ім’я покупця
surname	varchar(30)	Прізвище покупця
orderTime	Datetime2(7)	Час, коли зроблено замовлення
isByCourier	bit	Чи потрібна доставка кур’єром
phone	varchar(20)	Номер телефону покупця
orderNumber	int	Номер замовлення
UserId	varchar(450)	ID користувача

Таблиця Drug містить інформацію про лікарські засоби.

Поле	Тип	Опис
id	int	Унікальний ідентифікатор в базі даних
description	varchar(MAX)	Опис товару
producer	varchar(MAX)	Виробник товару
price	int	Ціна товару
name	varchar(MAX)	Назва товару

doze	varchar(MAX)	Кількість діючої речовини
volume	varchar(MAX)	Кількість ліків в одному упакуванні
img	varchar(MAX)	Посилання на зображення ліків
available	bit	Чи доступний товар
isFavorite	bit	Чи найпопулярніший
categoryID	bit	ID категорії товару

Таблиця Category містить інформацію про категорії товарів.

Поле	Тип	Опис
id	int	Унікальний ідентифікатор в базі даних
name	varchar(MAX)	Назва категорії
description	varchar(MAX)	Опис категорії

Таблиця Item містить інформацію про товари, які додаються у корзину.

Поле	Тип	Опис
id	int	Унікальний ідентифікатор в базі даних
drugID	int	ID ліків
price	int	Ціна ліків
shopCartID	varchar(MAX)	ID корзини

Таблиця OrderDetail містить інформацію про замовлення.

Поле	Тип	Опис
id	int	Унікальний ідентифікатор в базі даних
orderID	int	ID замовлення
drugID	int	ID ліків
price	bigint	Ціна ліків

Таблиця Asp.Net.Users містить інформацію про користувача.

Поле	Тип	Опис
Id	varchar(450)	Унікальний ідентифікатор в базі даних
UserName	varchar(256)	Ім’я користувача
PasswordHash	varchar(MAX)	Кешований пароль користувача
PhoneNumber	varchar(MAX)	Номер телефону користувача

Таблиця Asp.Net.Roles містить інформацію про права доступу користувачів.

Поле	Тип	Опис
Id	varchar(450)	Унікальний ідентифікатор в базі даних
Name	varchar(256)	Назва ролі

Таблиця Asp.Net.User.Roles містить інформацію про те, яку роль займає користувач.

Поле	Тип	Опис
UserId	varchar(450)	ID користувача
RoleId	varchar(450)	ID ролі

3.3. Функціональність сайту

3.3.1. Головна сторінка

Сайт онлайн-аптеки має головну сторінку, на якій відображаються найпопулярніші товари. За допомогою циклу обробки колекції foreach, ми відображаємо об’єкт типу Drug, колекція якого повертається з методу favDrugs, у файлі Razor Pages AllDrugs.cshtml (рисунок 3.3.1.1).

```
@model HomeViewModel

<div class="d-flex justify-content-center"
     style="color:aliceblue;margin-top:60px">
<h2>Найпопулярніші товари</h2>
</div>
<div class="d-flex justify-content-center">
    <div style="display:flex; flex-wrap:wrap;">
        @{
            foreach (Medicines drug in Model.favDrugs)
            {
                @Html.Partial("AllDrugs", drug);
            }
        }
    </div>
</div>
```

Рис. 3.3.1.1

Метод favDrugs належить класу HomeViewModel і реалізована за допомогою вбудованих селекторів та модифікаторів. У класі HomeController, який є частиною паттерну MVC, а саме контролером, реалізований метод Index, що повертає тип даних ViewResult (тобто файл з розширенням .html).

У середині методу Index методу favDrugs присвоюється значення з методу getFavDrugs, визначення якого знаходиться в інтерфейсі IMedicines(рисунок 3.3.1.2)., а реалізація — у класі MedicineRepository (рисунок 3.3.1.3).

```
public interface IMedicines
    {
        IEnumerable<Medicines> drugs { get; }
        IEnumerable<Medicines> getFavDrugs { get; }
        Medicines getDrug(int drugId);
    }
```

Рис. 3.3.1.2

```
public class MedicineRepository : IMedicines
   {
       private readonly AppDBContent appDBContent;

       public MedicineRepository(AppDBContent app)
       {
           this.appDBContent = app;
       }
       public IEnumerable<Medicines> drugs =>
         appDBContent.Drug.Include(c => c.Category);

       public IEnumerable<Medicines> getFavDrugs =>
         appDBContent.Drug.Where(p => p.isFavorite).Include(c => c.Category);

       public Medicines getDrug(int drugId) =>
         appDBContent.Drug.FirstOrDefault(p => p.id == drugId);

   }
```

Рис. 3.3.1.2

3.3.2. Сторінка виведення товарів за категоріями

Сторінка виведення категорій товарів обробляється контролером CategoriesController, який повертає зовнішній вигляд (view). При натисканні на іконку категорії (додаток інтерфейсу з категоріями) відбувається переадресація до контролеру Medicines. Дана сторінка реалізує вибір товарів за категоріями, а також можливість вивести всі товари (рисунок 3.3.2.1).

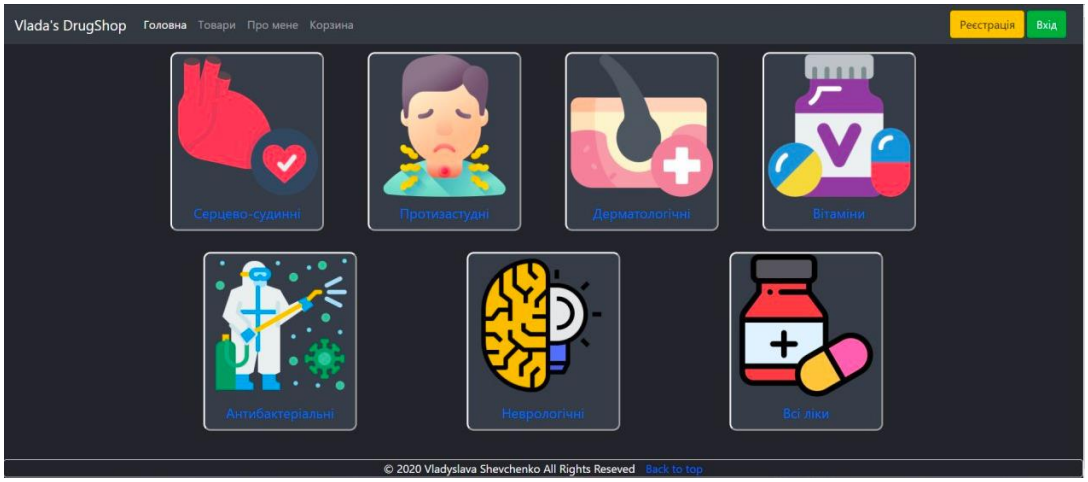


Рис. 3.3.2.1

3.3.3. Корзина

Корзина надає можливість створювати бронювання товару. Корзина реалізовується завдяки контролеру ShopCartController, який здійснює створення корзини та відображає зовнішний вигляд корзини та товарів у ній (рисунок 3.3.3.1).

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;

namespace Shop
{
    public class Program
    {
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                });
    }
}
```

Рис. 3.3.3.1

При натисканні кнопки «в корзину» відбувається виклик методу AddToCart контролера ShopCart. Цей метод викликає метод AddToCart (рисунок 3.3.3.2), який приймає в якості параметру об’єкт типу Medicine і передає його екземпляру бази даних, який викликає метод Add, класу ShopCart.

```
public void AddToCart(Medicines dr)
        {
            appDBContent.Item.Add(new ShopCartItem
            {
                ShopCartID = ShopCartId,
                drug = dr,
                price = dr.price
            });

            appDBContent.SaveChanges();
        }
```

Рис. 3.3.3.2

У Додатку А можна побачити корзину, у яку зареєстрований користувач додав три товари.

3.3.4. Створення замовлення

При натисканні кнопки «замовити» відбувається виклик контролеру OrderController, який у свою чергу викликає постметод CheckOut (рисунок 3.3.4.1). Цей метод перенаправляє нас на форму для введення даних про користувача та вибору способу доставки.

```
[HttpPost]
public IActionResult CheckOut(Order order)
{
    shopCart.listShopItem = shopCart.getShopItems();

    if(shopCart.listShopItem.Count ==0)
    {
        ModelState.AddModelError("", "У вас повинні бути товари");
    }

    if(ModelState.IsValid)
    {
        allOrders.createOrder(order,num);
        num++;
        if (order.isByCourier)
        {

            return RedirectToAction("CompleteByCourier");
        }
        else
        {
            return RedirectToAction("CompleteBySelf");
        }
    }

    return View(order);
}
```

Рис. 3.3.4.1

Залежно від вибору способи доставки відбуватиметься переадресація на різні методи, які у свою чергу повертають різне view (Додаток Б)

3.3.5. Реєстрація користувачів

Реєстрацією, авторизацією та виходом з профілю користувачів керує контролер AccountController.

При натисканні кнопки «Реєстрація» відкривається форма (рисунок 3.3.5.1), у якій користувач вводить дані про себе (логін користувача, номер телефону користувача, пароль, який вигадує користувач самостійно, та підтвердження паролю (додаток В).

```
@model RegisterViewModel

@{
    ViewBag.Title = "User Registration";
}

<h1>User registration</h1>

<div class="row" style="color:aqua">
    <div class="col-md-12">
        <form method="post">
            <div asp-validation-summary="All" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="name">Логін</label>
                <input asp-for="name" class="form-control" />
                <span asp-validation-for="name" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="phoneNumber">Номер телефону</label>
                <input asp-for="phoneNumber" class="form-control" />
                <span asp-validation-for="phoneNumber" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password">Пароль</label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ConfirmPassword">Підтвердіть пароль</label>
                <input asp-for="ConfirmPassword" class="form-control" />
                <span asp-validation-for="ConfirmPassword" class="text-danger">
</span>
            </div>
            <button type="submit" class="btn btn-primary">Реєстрація</button>
        </form>
    </div>
</div>
```

Рис. 3.3.5.1

Перш за все, всі поля є обов’язковими, тобто, якщо користувач не введе якісь дані, то система не дозволить йому зареєструватися без введення цих даних (рисунок 3.3.5.2). Поле номеру телефону може містити виключно цифри.

Водночас, пароль, який вводить користувач, має містити щонайменше шість символів, один з яких має бути символьним (наприклад, *, /, %, #). Крім цього, обов’язково у паролі повинна бути присутня принаймні одна літера маленького регістру та одна літера великого регістру. Також забороняється введення символів кирилицею, дозволена виключно латиниця.

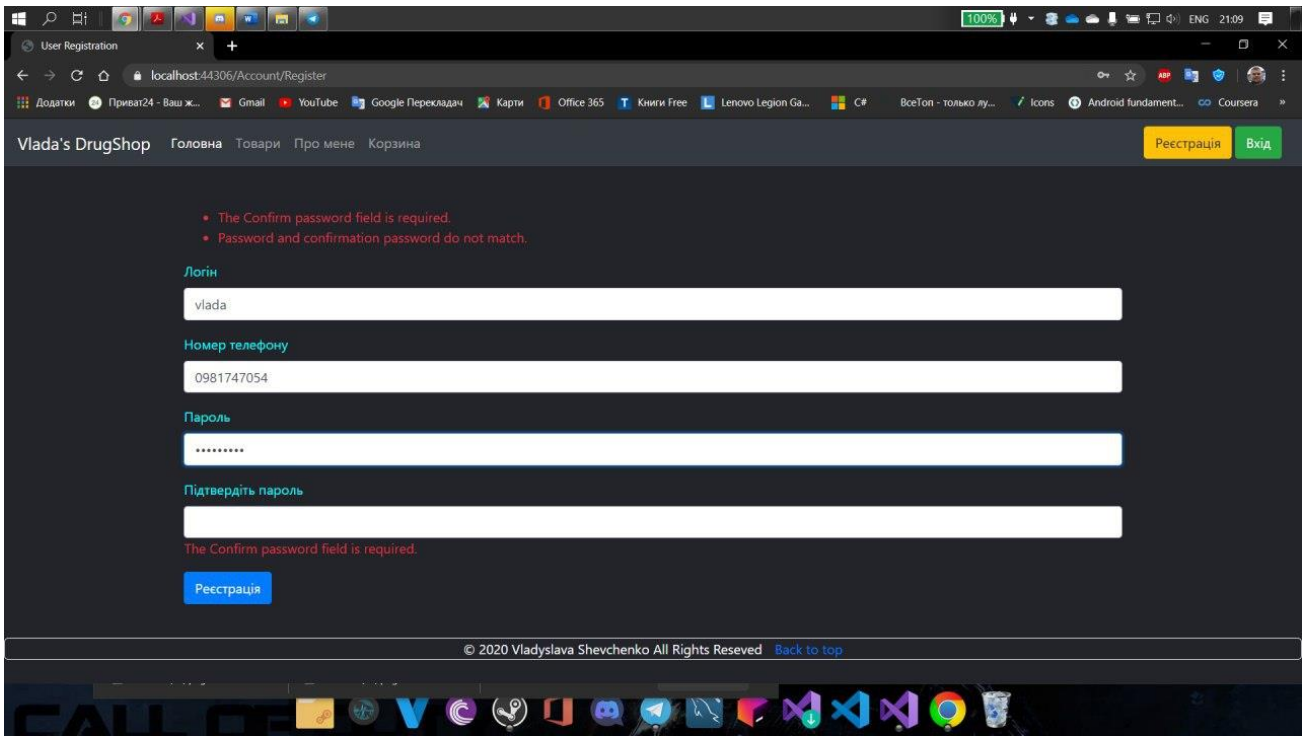


Рис. 3.3.5.2

Далі натиснувши кнопку «зареєструватися» викликається постметод Register (рисунок 3.3.5.3), який створює екземпляр користувача типу IdentityUser та додає його у базу даних. При успішній реєстрації відбувається переадресація користувача на головну сторінку.

```
[HttpPost]
        public async Task<IActionResult> Register(RegisterViewModel model)
        {
            if (ModelState.IsValid)
            {
                var user = new IdentityUser { UserName = model.name, PhoneNumber =
model.phoneNumber, PhoneNumberConfirmed = true };
                var result = await userManager.CreateAsync(user, model.Password);
                if (result.Succeeded)
                {
                    await signInManager.SignInAsync(user, isPersistent: false);
                    return RedirectToAction("Index", "Home");
                }

                foreach (var error in result.Errors)
                {
                    ModelState.AddModelError("", error.Description);
                }
            }
            return View(model);
        }
```

Рис. 3.3.5.3

3.3.6. Вхід до акаунта користувача

Під час входу викликається контролер AccountController (рисунок 3.3.6.1), проте на цей раз з контролера викликається метод Login. Цей метод відкриває форму, в якій користувач повинен ввести логін, пароль та вибрати функцію «запам’ятати мене» (це означає, що при закритті браузера вихід з акаунта здійснюватися не буде) або не вибрати. Всі поля у формі авторизації так само мають валідацію, як і всі поля у формі реєстрації.

При натисканні кнопки «вхід» викликається контролер AccountController, а саме постметод Login.

```
[HttpPost]
        public async Task<IActionResult> Login(LoginViewModel model)
        {
            if (ModelState.IsValid)
            {

                var result = await signInManager.PasswordSignInAsync(model.Login,
model.Password, model.RememberMe, false);
                if (model.Login=="admin" && model.Password=="1234admin")
                {
                    return RedirectToAction("Index", "Home");
                }
                else
                {
                    if (result.Succeeded)
                    {
                        return RedirectToAction("Index", "Home");
                    }
                    else
                    {
                        ModelState.AddModelError(string.Empty, "Неправильний логін
або пароль");
                    }
                }
            }
            return View(model);
        }
```

Рис. 3.3.6.1

Щоб користувач, який зайшов у свій акаунт, мав можливість вийти зі свого акаунту, потрібно натиснути кнопку «вихід». Коли користувач знаходиться у стані входу, то кнопки «реєстрація» та «вхід» змінюються на кнопку «вихід» (рисунок 3.3.6.2).

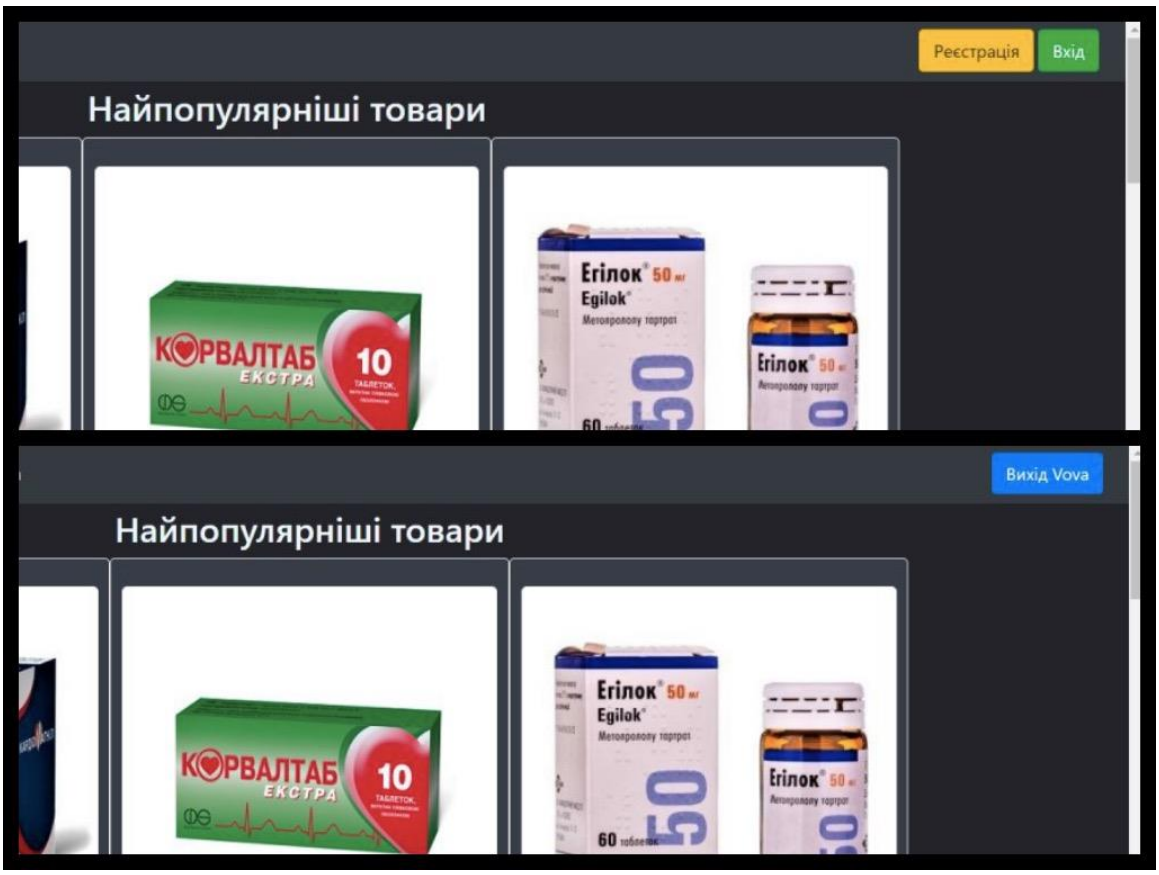


Рис. 3.3.6.2

При натисканні на кнопку «вихід» викликається метод LogOut (рисунок 3.3.6.3) з контролеру AccountController і відбувається перенаправлення на головну сторінку.

```
[HttpPost]
        public async Task<IActionResult> LogOut()
        {
            await signInManager.SignOutAsync();
            return RedirectToAction("Index", "Home");
        }
```

Рис. 3.3.6.2

3.3.7. Права доступу

Залежно від того, чи користувач зареєстрований, він може мати різні ролі та права доступа. Звичайний користувач сайту може переглядати всі доступні

ліки, шукати ліки за категоріями та йому доступна функція реєстрації на сайті. Послуга бронювання ліків доступна лише зареєстрованим клієнтам.

Зареєстрований покупець може переглядати всі доступні ліки, шукати ліки за категоріями, додавати ліки у корзину, замовити доставку кур'єром. Менеджер аптеки має можливість переглядати всі доступні, шукати ліки за категоріями, додавати нові надходження зі складів та дивитися історію замовлень покупців.

3.3.8. Додавання нового товару

При додаванні нового товару менеджер натискає на кнопку «додати товар». При натисканні на кнопку відбувається виклик постметоду AdminAddDrug (рисунок 3.3.8.1) контролера AdministrationController, який створює екземпляра класу Medicine та додає їх у базу.

```
[HttpPost]
        public IActionResult AdminAddDrug(Medicines drug)
        {
            if(ModelState.IsValid)
            {
                var addDrug = new Medicines
                {
                    available = drug.available,
                    categoryID = drug.categoryID,
                    description = drug.description,
                    doze = drug.doze,
                    img = drug.img,
                    isFavorite = drug.isFavorite,
                    name = drug.name,
                    price = drug.price,
                    producer = drug.producer,
                    volume = drug.volume
                };
                db.Add<Medicines>(addDrug);
                db.SaveChanges();
                return RedirectToAction("Index", "Home");
            }
            else
            {
                return View();
            }
        }
```

Рис. 3.3.8.1

Сторінка з інформацією (рисунок 3.3.8.2) про ліки має містити: назву препарату, короткий опис, категорію препарату, кількість ліків в упакуванні, кількість діючої речовини, виробника та вартість за одне упакування. Також кожний лікарський засіб має супроводжуватися фотографією упакування.

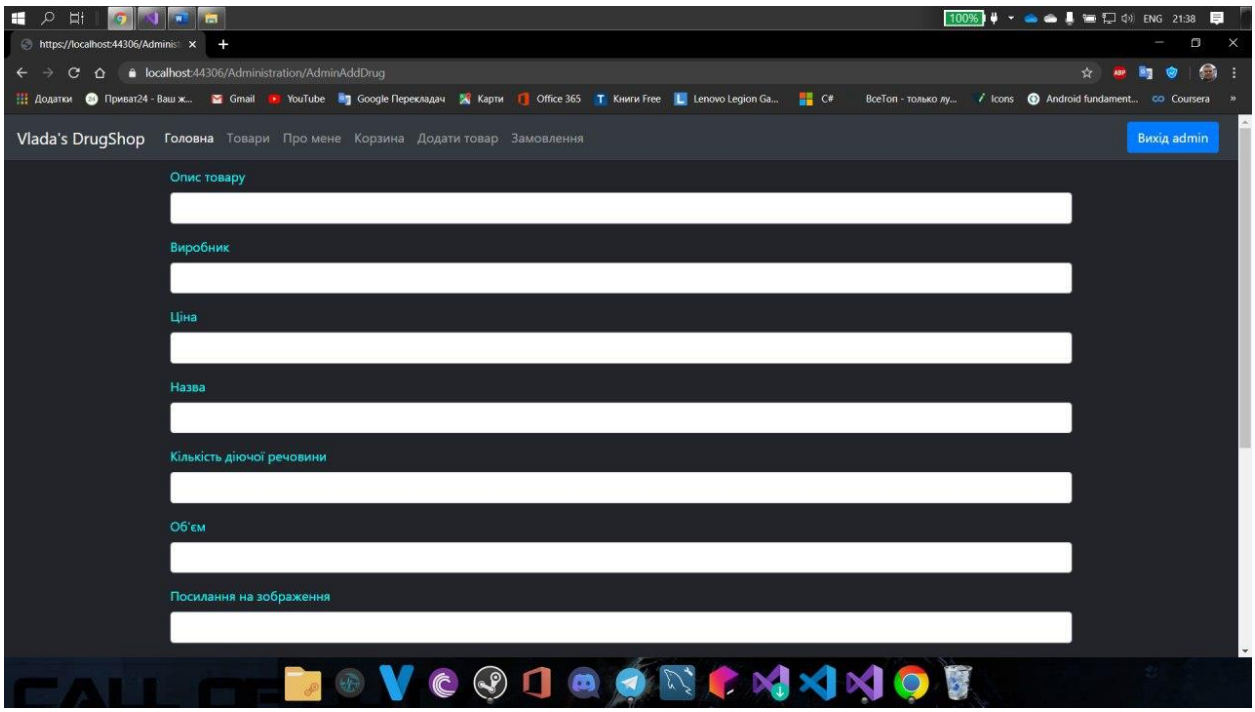


Рис. 3.3.8.2

3.3.9. Перегляд замовлення менеджером

При натисканні кнопки «замовлення» (рисунок 3.3.9.1) менеджером, викликається контролер AllOrdersController, а саме метод Index. Цей метод виводить з бази даних інформацію по замовлення та про користувача, який здійснив його.

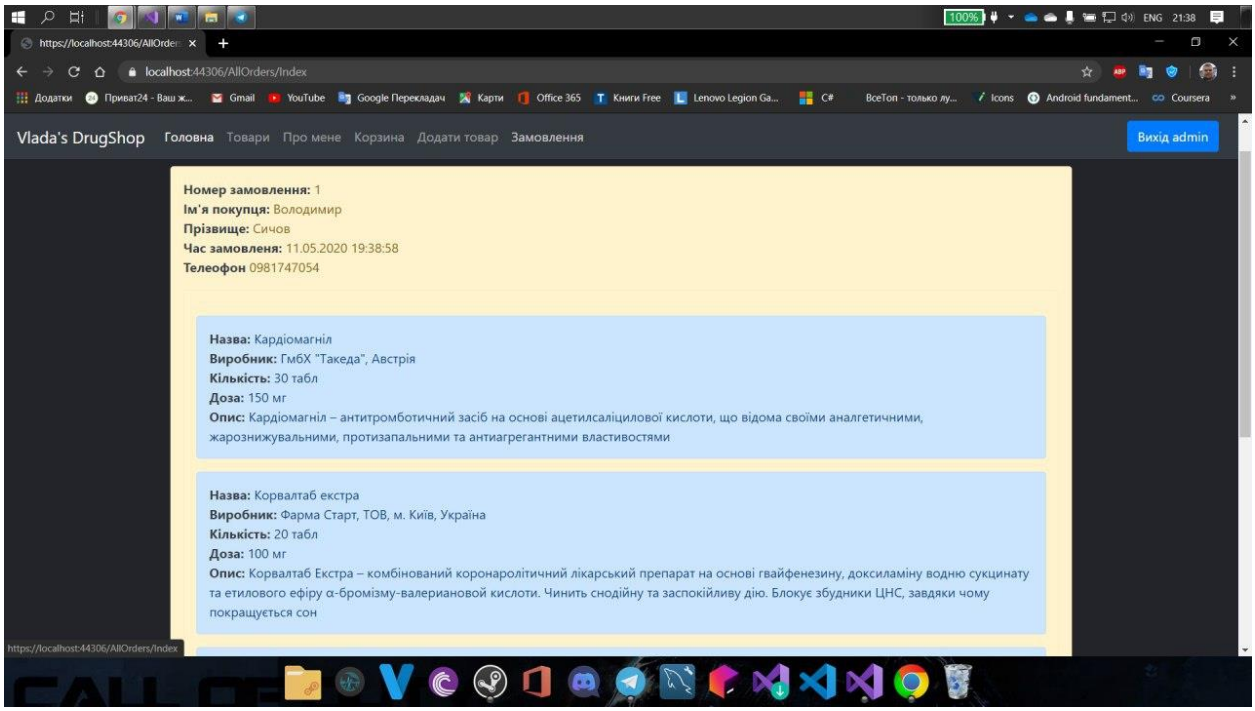


Рис. 3.3.8.2

Висновки

У рамках курсової роботи я познайомилася з платформою ASP.NET Core та багатьма новими технологіями. Однією з найважливіших технологій, які я використовувала, є підхід до розробки програмного забезпечення MVC (model-view-controller).

Крім цього, була використана нова для мене технологія роботи з базами даних — Entity Framework. У процесі розробки графічного інтерфейсу онлайн-сайту було розглянуто технологію Razor Pages та її особливість, а саме ASP-Helper-Tag.

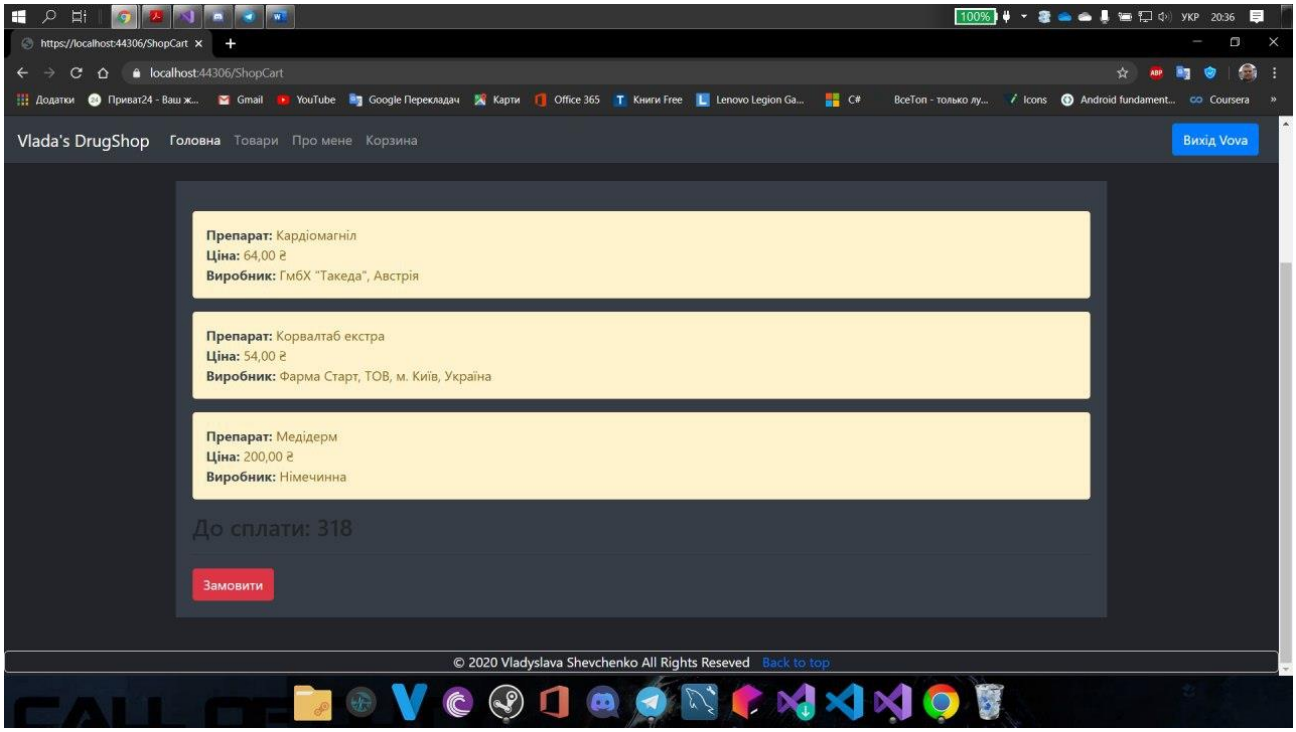
За результатами практичної частини ми отримали онлайн-сервіс аптеки, який підтримує бронювання лікарських засобів для покупців, оптимізує процес роботи для працівників, а також дозволяє аптеці збільшити кількість покупців завдяки зручному сервісу.

На мою думку, у подальшому я зможу покращити свої вміння в написанні веб-застосунків з використанням фреймворку ASP.NET Core та мови програмування C#. Продовжуючи здобувати нові вміння, сервіс онлайн-аптеки можна буде покращити, додавши новий функціонал, наприклад, пошук за назвою.

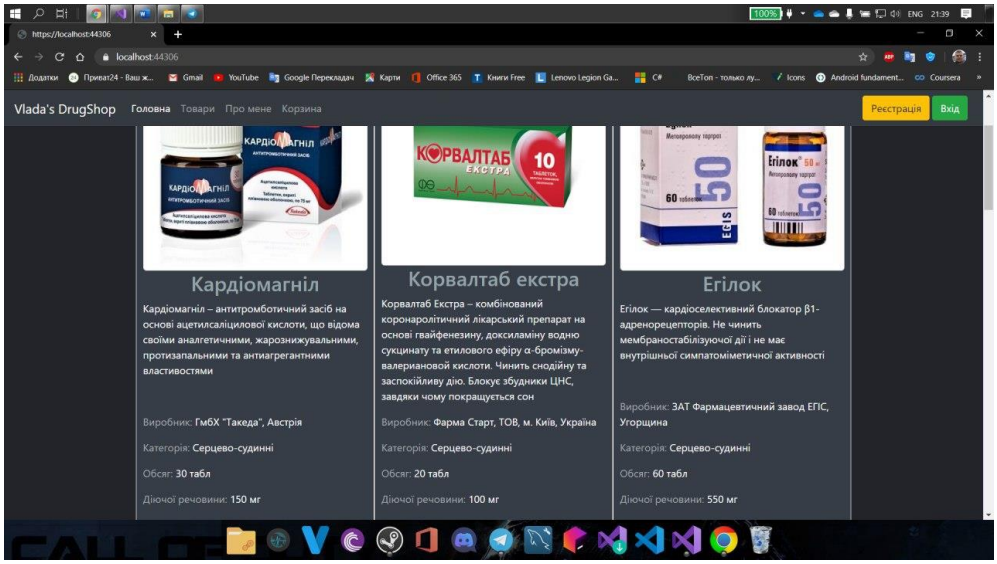
Список використаних джерел

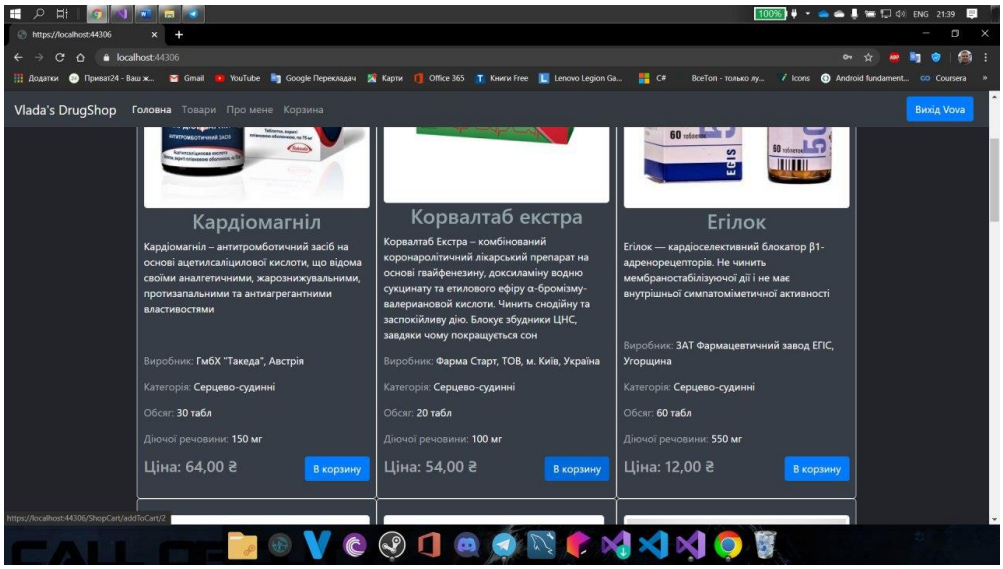
1. Pro ASP.NET Core MVC 2. Автор А. Фрімен. Рік видання – 2017.
2. ASP.NET Core in Action. Автор Е. Лок. Рік видання – 2018.
3. Pro Entity Framework Core 2 for ASP.NET Core MVC. Автор А. Фрімен. Рік видання – 2018.
4. MySQL Stored Procedure Programming. Автор Г. Гаррісон та С. Фейєрштейн. Рік видання — 2006.
5. ASP.NET Core 2.0 MVC & Razor Pages for Beginners: How to Build a Website. Автор Д. Фагерберг. Рік видання — 2017.

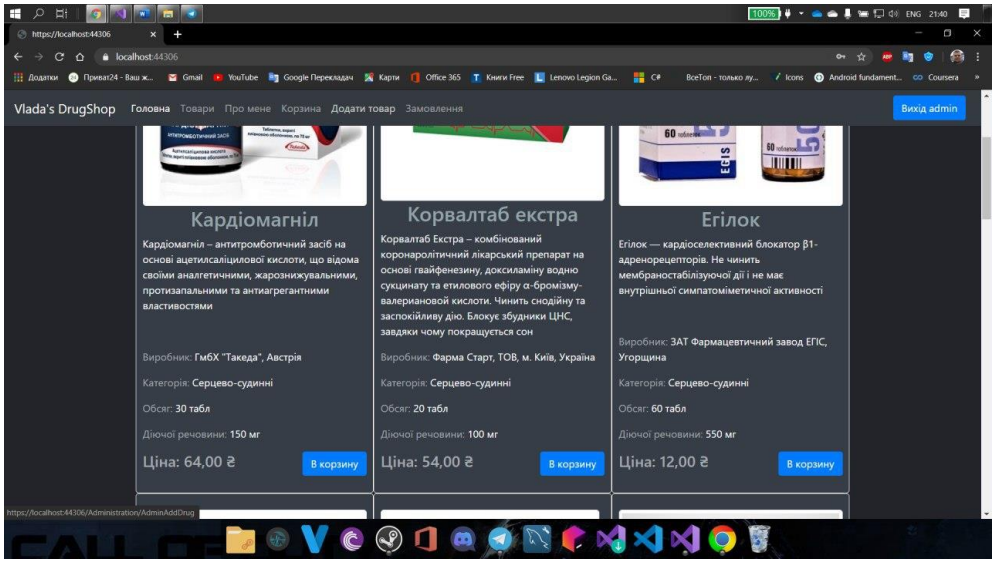
Додаток А


https://localhost:44306/ShopCart
localhost:44306/ShopCart
Vlada's DrugShop
Головна
Товари
Про мене
Корзина
Вихід Vova
Препарат: Кардіомагніл
Ціна: 64,00 ₴
Виробник: Гмбх "Такеда", Австрія
Препарат: Корвалтаб екстра
Ціна: 54,00 ₴
Виробник: Фарма Старт, ТОВ, м. Київ, Україна
Препарат: Медідерм
Ціна: 200,00 ₴
Виробник: Німечинна
До сплати: 318
Замовити
© 2020 Vladyslava Shevchenko All Rights Reseved
Back to top

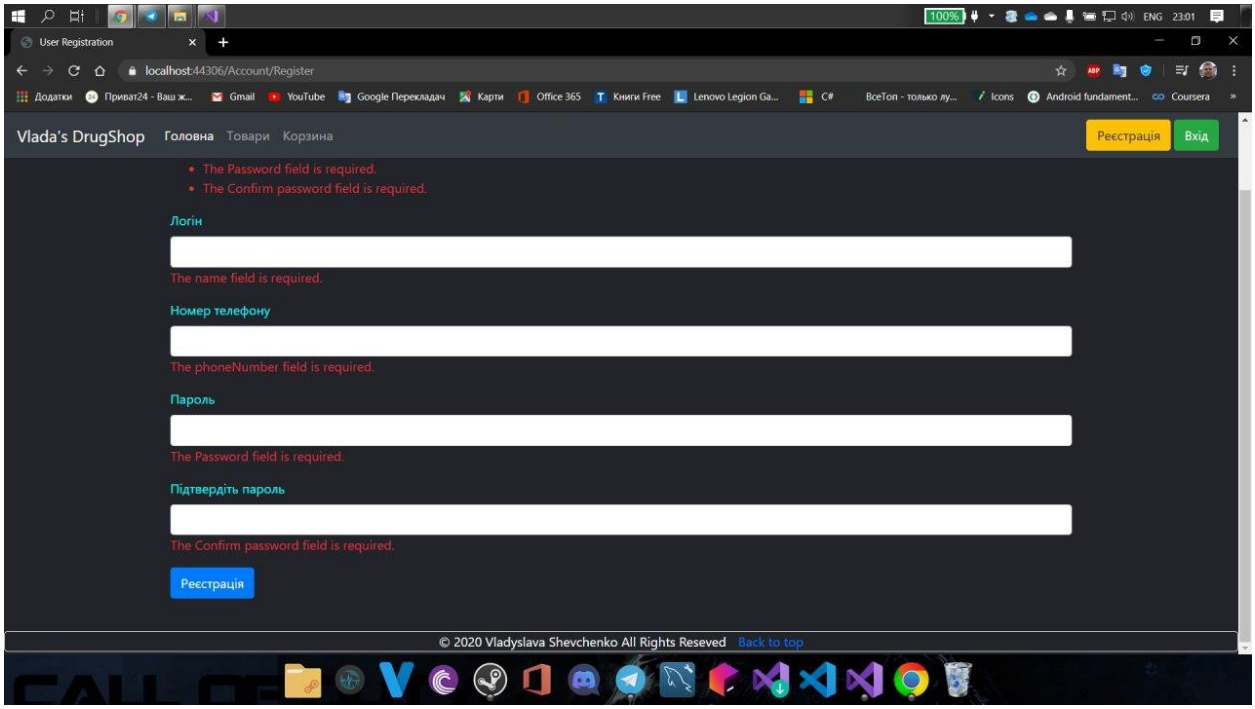
Додаток Б


Vlada's DrugShop
Головна
Товари
Про мене
Корзина
Реєстрація
Вхід
Кардіомагніл
Кардіомагніл – антитромботичний засіб на основі ацетилсаліцилової кислоти, що відома своїми аналгетичними, жарознижувальними, протизапальними та антиагрегантними властивостями
Виробник: ГмбХ "Такеда", Австрія
Категорія: Серцево-судинні
Обсяг: 30 табл
Діючої речовини: 150 мг
Корвалтаб екстра
Корвалтаб Екстра – комбінований коронаролітичний лікарський препарат на основі гвайфенезину, доксиламіну водню сукцинату та етилового ефіру α-бромізму-валеріанової кислоти. Чинить снодійну та заспокійливу дію. Блокує збудники ЦНС, завдяки чому покращується сон
Виробник: Фарма Старт, ТОВ, м. Київ, Україна
Категорія: Серцево-судинні
Обсяг: 20 табл
Діючої речовини: 100 мг
Егілок
Егілок — кардіоселективний блокатор β1-адренорецепторів. Не чинить мембраностабілізуючої дії і не має внутрішньої симпатоміметичної активності
Виробник: ЗАТ Фармацевтичний завод ЕПІС, Угорщина
Категорія: Серцево-судинні
Обсяг: 60 табл
Діючої речовини: 550 мг


Vlada's DrugShop
Головна
Товари
Про мене
Корзина
Вихід Vova
Кардіомагніл
Кардіомагніл – антитромботичний засіб на основі ацетилсаліцилової кислоти, що відома своїми аналгетичними, жарознижувальними, протизапальними та антиагрегантними властивостями
Виробник: ГмбХ "Такеда", Австрія
Категорія: Серцево-судинні
Обсяг: 30 табл
Діючої речовини: 150 мг
Ціна: 64,00 ₴
В корзину
Корвалтаб екстра
Корвалтаб Екстра – комбінований коронаролітичний лікарський препарат на основі гвайфенезину, доксиламіну водню сукцинату та етилового ефіру α-бромізму-валеріанової кислоти. Чинить снодійну та заспокійливу дію. Блокує збудники ЦНС, завдяки чому покращується сон
Виробник: Фарма Старт, ТОВ, м. Київ, Україна
Категорія: Серцево-судинні
Обсяг: 20 табл
Діючої речовини: 100 мг
Ціна: 54,00 ₴
В корзину
Егілок
Егілок — кардіоселективний блокатор β1-адренорецепторів. Не чинить мембраностабілізуючої дії і не має внутрішньої симпатоміметичної активності
Виробник: ЗАТ Фармацевтичний завод ЕПІС, Угорщина
Категорія: Серцево-судинні
Обсяг: 60 табл
Діючої речовини: 550 мг
Ціна: 12,00 ₴
В корзину


Vlada's DrugShop
Головна
Товари
Про мене
Корзина
Додати товар
Замовлення
Вихід admin
Кардіомагніл
Кардіомагніл – антитромботичний засіб на основі ацетилсаліцилової кислоти, що відома своїми аналгетичними, жарознижувальними, протизапальними та антиагрегантними властивостями
Виробник: ГмбХ "Такеда", Австрія
Категорія: Серцево-судинні
Обсяг: 30 табл
Діючої речовини: 150 мг
Ціна: 64,00 ₴
В корзину
Корвалтаб екстра
Корвалтаб Екстра – комбінований коронаролітичний лікарський препарат на основі гвайфенезину, доксиламіну водню сукцинату та етилового ефіру α-бромізму-валеріанової кислоти. Чинить снодійну та заспокійливу дію. Блокує збудники ЦНС, завдяки чому покращується сон
Виробник: Фарма Старт, ТОВ, м. Київ, Україна
Категорія: Серцево-судинні
Обсяг: 20 табл
Діючої речовини: 100 мг
Ціна: 54,00 ₴
В корзину
Егілок
Егілок — кардіоселективний блокатор β1-адренорецепторів. Не чинить мембраностабілізуючої дії і не має внутрішньої симпатоміметичної активності
Виробник: ЗАТ Фармацевтичний завод ЕПІС, Угорщина
Категорія: Серцево-судинні
Обсяг: 60 табл
Діючої речовини: 550 мг
Ціна: 12,00 ₴
В корзину

Додаток В


User Registration
localhost:44306/Account/Register
Vlada's DrugShop
Головна
Товари
Корзина
Реєстрація
Вхід
The Password field is required.
The Confirm password field is required.
Логін
The name field is required.
Номер телефону
The phoneNumber field is required.
Пароль
The Password field is required.
Підтвердіть пароль
The Confirm password field is required.
Реєстрація
© 2020 Vladyslava Shevchenko All Rights Reseved
Back to top