ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

НА ПРОЄКТНУ-ТЕХНОЛОГІЧНУ ПРАКТИКУ

Студенту групи *К12-19-1*

спеціальності *123* «*Комп’ютерна інженерія*»

*Гусачекну Миколі Сергійовичу*  .

Прізвище, ім’я, по батькові

База практики *ПП «АВІВІ», м. Хмельницький*

Термін практики з 06.06.2022 р. по 26.06.2022 р.

Тема індивідуального завдання  *Розробити веб-сайт з імітацією реального замовлення*

Завдання видав: *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Завдання одержав: \_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис, дата

Хмельницький національний університет

Кафедра комп’ютерної інженерії та програмування

Звіт

з науково-дослідної практики

база практики *ПП «АВІВІ», м. Хмельницький*

                          ЗППКІ. XXXXX. XX. XX

Шифр Номер ІНПС, номер за списком, номер теми в наказі

Галузь знань *12* «*Інформаційні технології*»

Спеціальність *123* «*Комп’ютерна інженерія*»

Студента III курсу, група КІ2–19-1 С. М. Гусаченка

Керівник від кафедри Керівник від бази практики

Павлова О.О. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис, прізвище Підпис, прізвище

Кількість балів \_\_\_\_

Оцінка за шкалою: національною\_\_\_\_\_\_/ЄКТС\_\_\_\_

Члени комісії: \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис Ініціали, прізвище

Хмельницький 2022

ЗМІСТ

[ВСТУП 3](#_Toc107149606)

[1 ЗАГАЛЬНА ХАРАКТЕРИСТИКА БАЗИ ПРАКТИКИ 4](#_Toc107149607)

[1.1 Постановка задачі для проєктування веб-сайту 4](#_Toc107149608)

[1.3 View Component Визначення компонента уявлень 6](#_Toc107149609)

[1.4 Робота з даними у додатках ASP.NET Core 7](#_Toc107149610)

[2 ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ ТА РЕАЛІЗАЦІЯ ПРОЄКТУ 8](#_Toc107149611)

[2.1  Аналіз предметної області і виявлення наявних проблем і завдань 8](#_Toc107149612)

[2.2  Контролери та їх дії 11](#_Toc107149613)

[2.3 Скріншоти сайту 12](#_Toc107149614)

[Висновок 16](#_Toc107149615)

[Список використаних джерел 17](#_Toc107149616)

[Додаток А 18](#_Toc107149617)

[Скріншоти сайту 18](#_Toc107149618)

ВСТУП

Платформа ASP.NET Core представляє технологію від компанії Microsoft, призначену для створення різноманітних веб-додатків: від невеликих веб-сайтів до великих веб-порталів та веб-сервісів.

ASP.NET Core може працювати поверх крос-платформного середовища .NET Core, яке може бути розгорнуто на основних популярних операційних системах: Windows, Mac OS, Linux. І таким чином, за допомогою ASP.NET Core ми можемо створювати крос-платформні програми. І хоча Windows як середовище для розробки та розгортання програми досі переважає, але тепер ми не обмежені тільки цією операційною системою. Тобто ми можемо запускати веб-програми не тільки на ОС Windows, але і на Linux і Mac OS. А для розгортання веб-програми можна використовувати традиційний IIS, або крос-платформний веб-сервер Kestrel.

Як інструментарій розробки ми можемо використовувати останні випуски Visual Studio, починаючи з версії Visual Studio 2015. Крім того, ми можемо створювати додатки в середовищі Visual Studio Code, яке є крос-платформним і може працювати як на Windows, так і на Mac OS X та Linux.

1. ЗАГАЛЬНА ХАРАКТЕРИСТИКА БАЗИ ПРАКТИКИ

## 1.1 Постановка задачі для проєктування веб-сайту

Завдання включає: розробити веб-сайт з імітацією реального замовлення

клієнта згідно запропонованого нижче варіанту технічного завдання. Сайт повинен містити галерею продуктів (картки товарів) для вибору та можливість замовлення товару через форму на сайті (клієнт може залишити свої контактні дані на сайті (ім’я, номер телефону) та вказати бажаний товар, щоб з ним можна було сконтактувати). А також сторінку адміністратора, де можна переглянути інформацію про клієнтів у вигляді таблиці - ім’я, номер телефону та повідомлення.

1.2 Організаційна структура бази практики (ПП «АВІВІ»)

Базою практики виступає приватне підприємство «АВІВІ» м. Хмельницький. Це підприємство було засновано у 2008 році.

Основним видом діяльності є впровадження CRM, розробка сучасного програмного забезпечення від невеличкого сайту чи інтернет-магазину до великого корпоративного порталу.

Основою підприємства є команда ІТ-професіоналів, сертифікованих компанією «Бітрікс».

Сьогодні «АВІВІ» використовує всі сучасні технології і тенденції з метою реалізації найефективніших інструментів для розвитку бізнесу будь-якого масштабу.

Штат підприємства складає понад 50 осіб. Основним напрямком робіт ТОВ «АВІВІ» є допомога клієнтам заробляти більше з використанням сучасних технологій шляхом впровадження сучасних інформаційних технологій

Персонал підприємcтва – висококваліфіковані фахівці у галузі програмування веб-сервісів. Підприємство має 4 офіси, більше 20 країн-клієнтів, та більше 100 успішних імплементованих у бізнес ІТ-проектів.

Компанія «АВІВІ» включає дирекцію, бухгалтерію, відділ кадрів, відділ технічної підтримки, віддів софтверного девелопменту з командами керівників проектів та розробників програмного забезпечення.

Дирекцію складає два співзасновники, яким підпорядковуються усі відділи компанії.

Відділ технічної підтримки нараховує 5 співробітників, серед яких 2 девопси та три системних інженери.

Відділ софтверного девелопменту включає PR-менеджера, 5 керівників проектів та більше 40 розробників програмного забезпечення, які в залежності від замовлення формують необхідну команду розробки ПЗ, та команда з 5 веб-дизайнерів.

## 1.3 View Component Визначення компонента уявлень

View Component (або компонент уявлень) представляє нову концепцію, яка з'явилася в ASP.NET Core MVC, яка багато в чому схожа з частковими уявленнями. При цьому View Component призначений для таких завдань, які, можливо, складно вирішити за допомогою одних часткових уявлень, наприклад створення динамічних меню, хмара тегів, панель логіна, кошик покупок і так далі.

View Component і двох частин: класу на C# і часткового уявлення, яке викликає методи цього. При цьому View Component не може обробляти HTTP-запити, а контент, що генерується ним, включається в код батьківського подання, в якому викликається компонент. Нерідко View Component використовується на майстер-сторінці \_layout.cshtml.

І подібно до контролерів view component повинен представляти клас з модифікатором public. Крім того, клас компонента не повинен бути вкладеним чи абстрактним.

View Component підтримує використання залежностей, тому в коді view component можна отримати залежності з провайдера сервісів. У той самий час комопонент перестав бути частиною життєвого циклу контролера, тому до нього не можна застосовувати фільтри.

Є три способи визначення компонента:

* визначення компонента як звичайного класу (клас POCO)
* успадкування від базового класу ViewComponent
* застосування до класу атрибуту [ViewComponent]

Для роботи з компонентами уявлень створимо новий проект ASP.NET Core на кшталт Web Application (Model-View-Controller).

Tag-хелпер vc:

Також ми можемо вбудовувати комопонент за допомогою тег-хелпера vc у форматі:

<vc:[ім'я\_компонента]></vc:[ім'я\_компонента]>

У нашому випадку ім'я комопонента - "Timer", тому, щоб звернутися до компонента у поданні, можна використовувати наступну розмітку:

<vc:timer></vc:timer>

або

<vc:timer />

Однак перед використанням тега необхідно зареєструвати наш компонент як tag-хелпер. Для цього до файлу \_ViewImports.cshtml до папки Views додамо наступну директиву:

@addTagHelper \*, [Назва\_складання\_проекту]

Назва складання проекту зазвичай збігається з назвою самого проекту. Тобто в моєму випадку проект називається ViewComponents, тому я додаю до файлу \_ViewImports.cshtml рядок

@addTagHelper \*, ViewComponentsApp

## 1.4 Робота з даними у додатках ASP.NET Core

Доступ до даних є одним із найважливіших складових практично будь-якої програми. ASP.NET Core підтримує різні способи доступу до даних, включаючи Entity Framework Core (та Entity Framework 6), і може працювати з будь-якими платформами доступу до даних .NET. Вибір платформи доступу до даних залежить від потреб програми. Абстрагуючи вибір такої платформи від проектів ядра програми та інтерфейсу користувача, а також інкапсулюючи деталі реалізації в інфраструктурі, можна створювати слабко пов'язану програму з розширеними можливостями для тестування.

Якщо ви створюєте нову програму ASP.NET Core для роботи з реляційними даними, рекомендується вибирати Entity Framework Core (EF Core) для доступу до даних. EF Core є об'єктно-реляційним модулем зіставлення (O/RM), який дозволяє розробникам .NET забезпечувати двосторонню збереження об'єктів у взаємодії з джерелом даних. Це усуває необхідність здебільшого коду доступу до даних, який розробникам зазвичай доводиться писати. Як і ASP.NET Core, платформа EF Core була перероблена, щоб забезпечити підтримку модульних кросплатформових додатків. Ви можете додати її до програми у вигляді пакета NuGet, налаштувати при запуску програми та запросити її за допомогою впровадження залежностей у потрібний момент.

Щоб використовувати EF Core з базою даних SQL Server, виконайте таку команду dotnet CLI:

dotnet add package Microsoft.EntityFrameworkCore.SqlServer

Якщо ви створюєте нову програму ASP.NET Core для роботи з реляційними даними, рекомендується вибирати Entity Framework Core (EF Core) для доступу до даних. EF Core є об'єктно-реляційним модулем зіставлення (O/RM), який дозволяє розробникам .NET забезпечувати двосторонню збереження об'єктів у взаємодії з джерелом даних. Це усуває необхідність здебільшого коду доступу до даних, який розробникам зазвичай доводиться писати. Як і ASP.NET Core, платформа EF Core була перероблена, щоб забезпечити підтримку модульних кросплатформових додатків. Ви можете додати її до програми у вигляді пакета NuGet, налаштувати при запуску програми та запросити її за допомогою впровадження залежностей у потрібний момент.

1. ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖУВАНОЇ ПРОБЛЕМИ ТА РЕАЛІЗАЦІЯ ПРОЄКТУ

2.1  Аналіз предметної області і виявлення наявних проблем і завдань

Сайт як засіб передачі інформації та вид комунікативних зв'язків

Головне призначення сайту – це надання інформації. Здавалося б, у світі існує маса інших способів передачі інформації, чому слід застосувати саме цей? Просто він найефективніший. У чому його ефективність:

По-перше, у зв'язку з божевільною швидкістю розвитку мережі Інтернет відбувається втягування все більшої кількості людей у ​​сферу використання Інтернет-технологій та ресурсів. Якщо влаштувати опитування на тему “Чи потрібний вам Інтернет?”, то 62% від всього населення дадуть утвердний відповідь, і багато з них скажуть, що вже не уявляють свого життя без його використання, як без телевізора чи телефону. Тому не дивно, що за допомогою сайту про Вас дізнаються більше потенційних клієнтів, ніж з газети.

По-друге, інформацію, яку ви виклали на сайт, відразу бачать ваші клієнти. Чим швидше ваш клієнт отримає інформацію, тим швидше він ухвалить рішення про те, що йому треба з вами зв'язатися, і тим швидше він це зробить, а значить, Ваш власний сайт – це справді вдалий спосіб заяви про себе.

Сайт як рекламний продукт

Зазвичай реклама сприймається нами як щось негативне. Нескінченні рекламні ролики, які з кожним днем ​​з'являються на екранах наших телевізорів, мало кому приємні. Але насправді, без реклами, сучасне суспільство уявити собі досить важко. І як би критично ми не ставилися до цього, реклама здатна по-справжньому покращити приплив клієнтів і таким чином покращити розвиток діяльності вашого підприємства. Головне правильно підійти до цього питання. А для цього необхідно чітко структурувати інформацію і абсолютно точно знати, що ви хочете донести до інших.

Реклама – це відкрите оповіщення про товар, послуги, яке проводиться за допомогою різних засобів: окремих видань (проспекти, каталоги, плакати, листівки), періодичної преси (статті, оголошення, вкладки), кіно, телебачення, радіо, зовнішньої, прямої поштової реклами .

Сайт – це все перераховане разом. Мета даного напрямку максимально розвинути та підвищити віддачу Інтернет-проекту, збільшити кількість користувачів відповідно потенційних і реальних клієнтів.

Реклама – це “двигун торгівлі”, сайт – двигун вашої фірми, що відповідає за її просування.

А просуванням зазвичай називають комплекс заходів, спрямованих на залучення до сайту уваги аудиторії та, як наслідок, збільшення його відвідуваності та підвищення позиції, займаної сайтом у різних рейтингах. Сайт, поряд з іншими джерелами інформації найбільшим чином відповідає всім вище перерахованим вимогам, крім того, правдоподібність інформації про вашу діяльність залежить тут від вас, адже створюючи сайт, ви самі визначаєте те, що і як на ньому буде розміщено, на відміну від газет.

Для просування сайту може використовуватися безліч способів:

* різні види традиційної реклами;
* електронні розсилки;
* покази банерів;
* розміщення інформації на інших сайтах;
* обмін посиланнями, кнопками, банерами.

За своїми функціями та властивостями сайти бувають: візитками, електронними магазинами, інформаційними сайтами, корпоративними представництвами, порталами, системами управління підприємством.

Сайт візитка – це, як правило, простий сайт, який містить загальну інформацію про компанію та послуги, що надаються. Короткий та компактний, зовсім як ваша візитка.

Інтернет-магазин - сайт, призначений для продажу товарів, послуг через Інтернет. Як правило, містить каталог продукції, прайс-листи, систему замовлень.

Інформаційний сайт – сайт, який містить вичерпну інформацію з певної предметної області. Сайти цього типу зазвичай містять безліч статей різних авторів, а також такі сервіси як: опитування, голосування, розсилки. Здебільшого некомерційного типу.

Корпоративне представництво – такого типу сайти, як правило, автоматизують діяльність компанії. Може нести такі функції як: електронний магазин, систему замовлень, комунікаційні сервіси, електронний обмін документами, online переговори тощо.

Портал – це дуже великий веб-ресурс, який призначений для формування якоїсь спільноти людей із певними інтересами. Портал може об'єднувати безліч різних сервісів, надавати клієнтам можливість купівлі товарів, партнерам – обмін інформацією.

Ще одним вагомим доказом на користь створення сайту є те, що сайт – це сучасний і актуальний засіб надання інформації та заявки про себе. Наявність власного сайту в наш час є правилом гарного тону і запорукою успіху у розвитку Вашого бізнесу.

2.2  Контролери та їх дії

Центральною ланкою в архітектурі ASP.NET Core MVC є контролер. При отриманні запиту система маршрутизації вибирає для обробки запиту необхідний контролер і передає дані запиту. Контролер обробляє ці дані та посилає назад результат обробки.

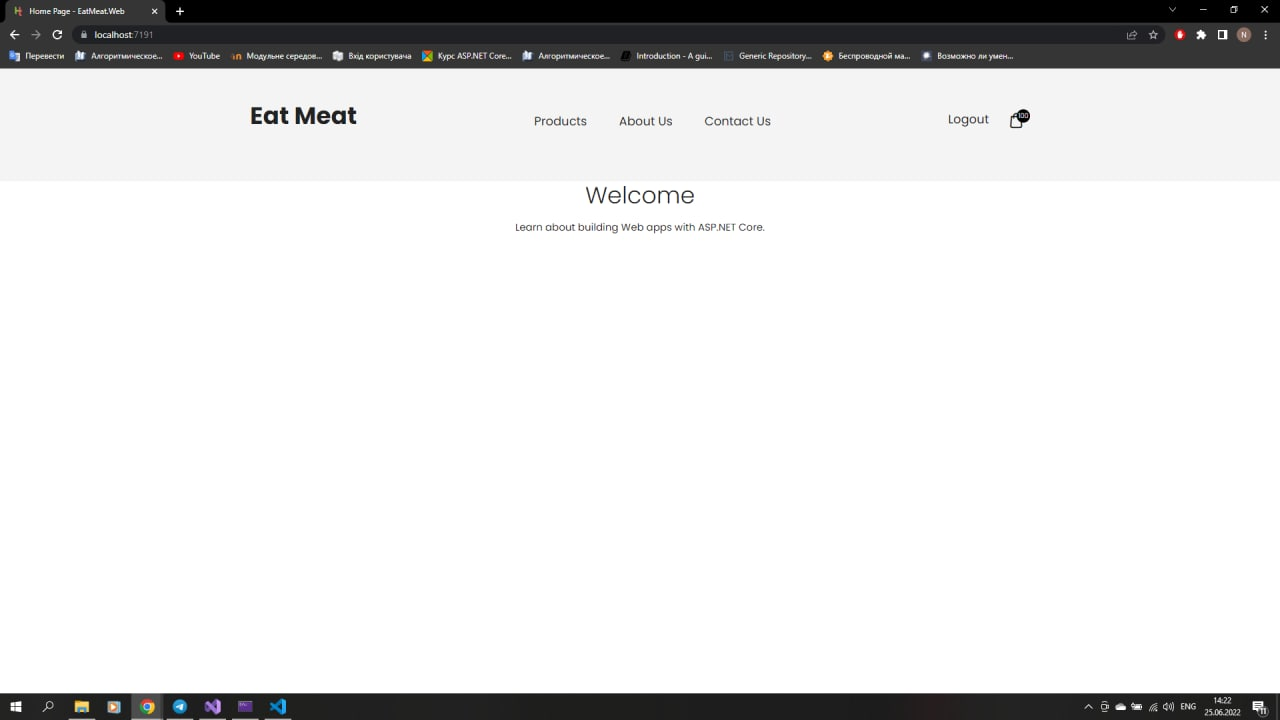
ASP.NET Core MVC контролер представляє звичайний клас мовою C#, який успадковується від абстрактного базового класу Microsoft.AspNetCore.Mvc.Controller.

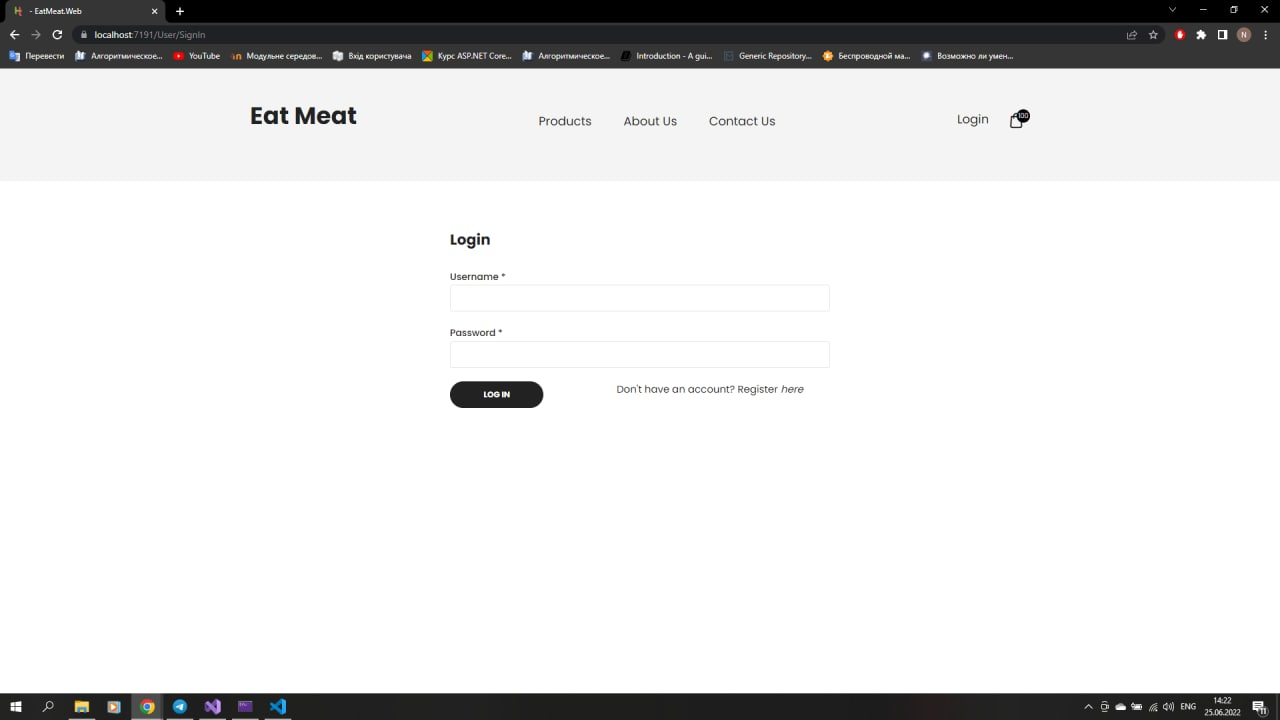
Контролер, як і будь-який клас мовою C#, може мати поля, властивості, методи. За умовчанням, HomeController має чотири методи, які можна назвати діями. Дії контролера - це громадські способи, які можуть зіставлятися із запитами. Наприклад, стандартний контролер містить метод Index – він має модифікатор public і тому може використовуватись для обробки запиту.

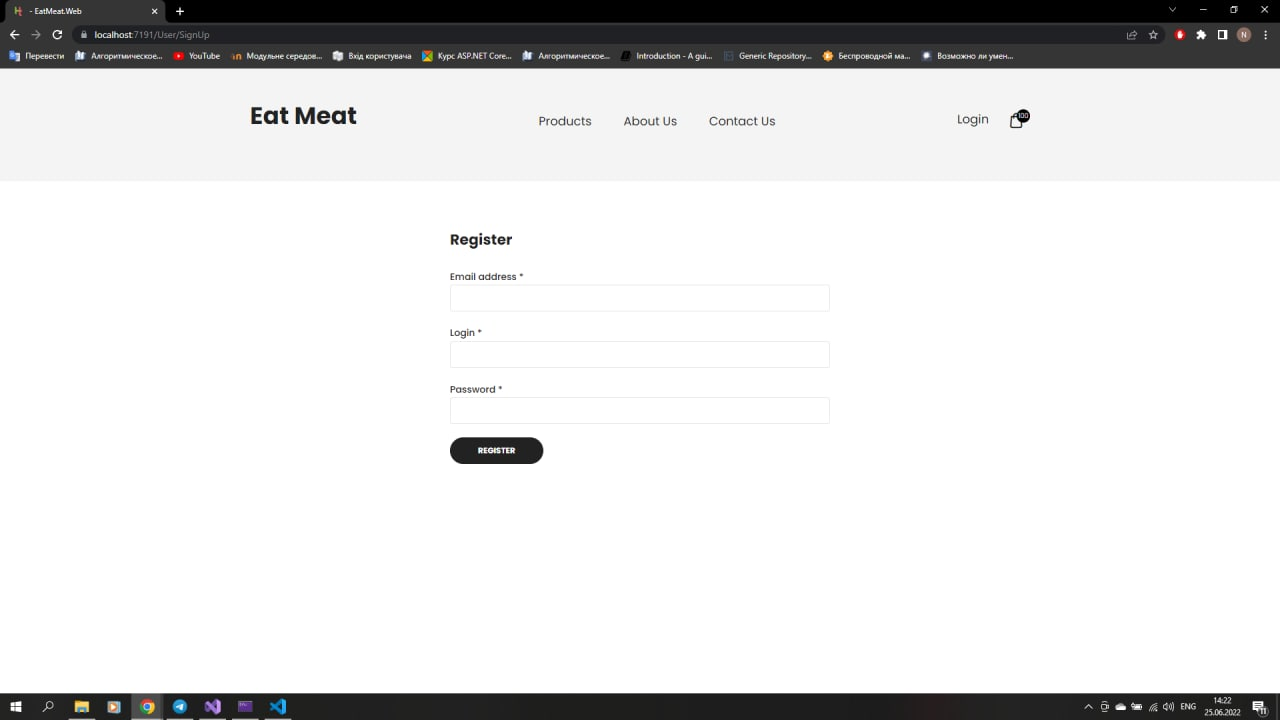
Щоб звернутися контролеру з веб-браузера, нам потрібно в адресному рядку набрати адресу\_сайту/ім'я\_контролера/дія\_контролера. Так, за запитом адреси\_сайту/Home/Index система маршрутизації за замовчуванням викличе метод Index контролера HomeController для обробки вхідного запиту.

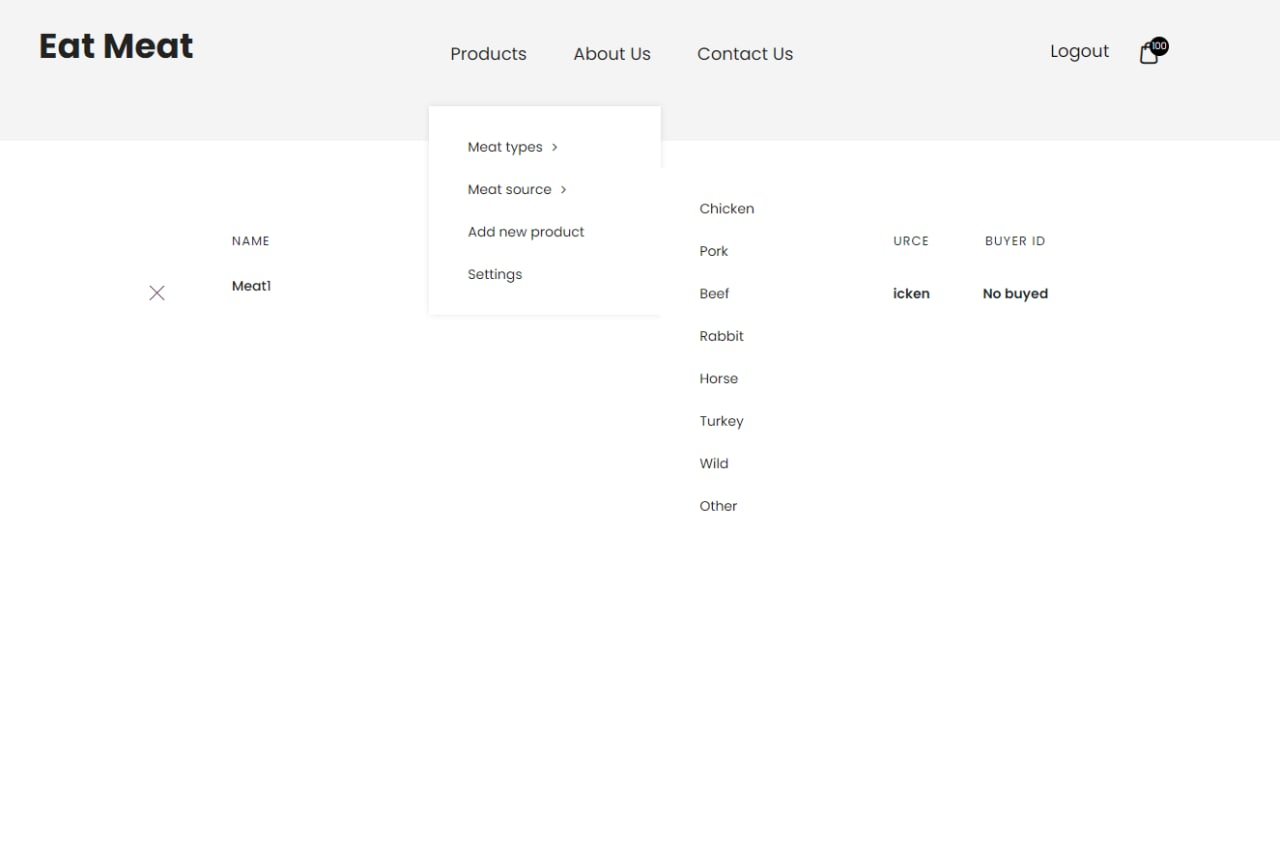
Методи у межах однієї дії можуть обслуговувати різні запити. Для вказівки типу запиту HTTP нам потрібно застосувати до методу один із атрибутів: [HttpGet], [HttpPost], [HttpPut], [HttpDelete] та [HttpHead]. Якщо атрибут явно не вказано, то метод може обробляти всі типи запитів: GET, POST, PUT, DELETE.

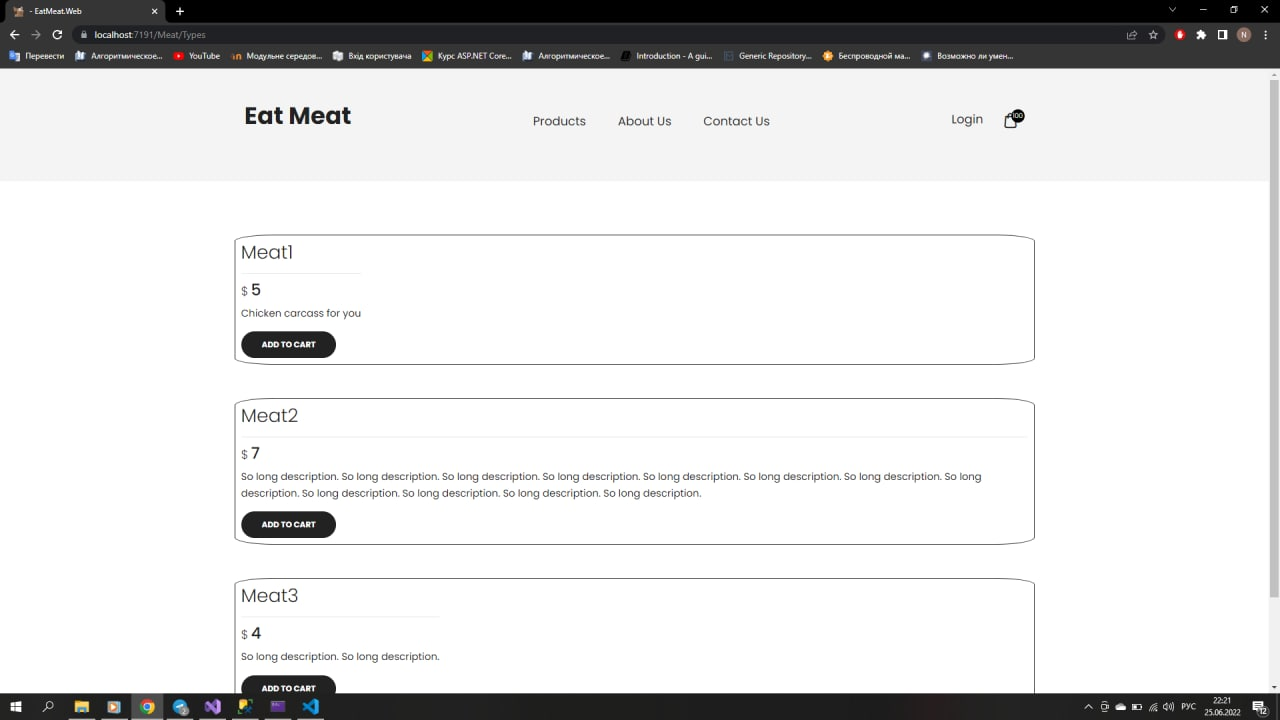
## 2.3 Скріншоти сайту

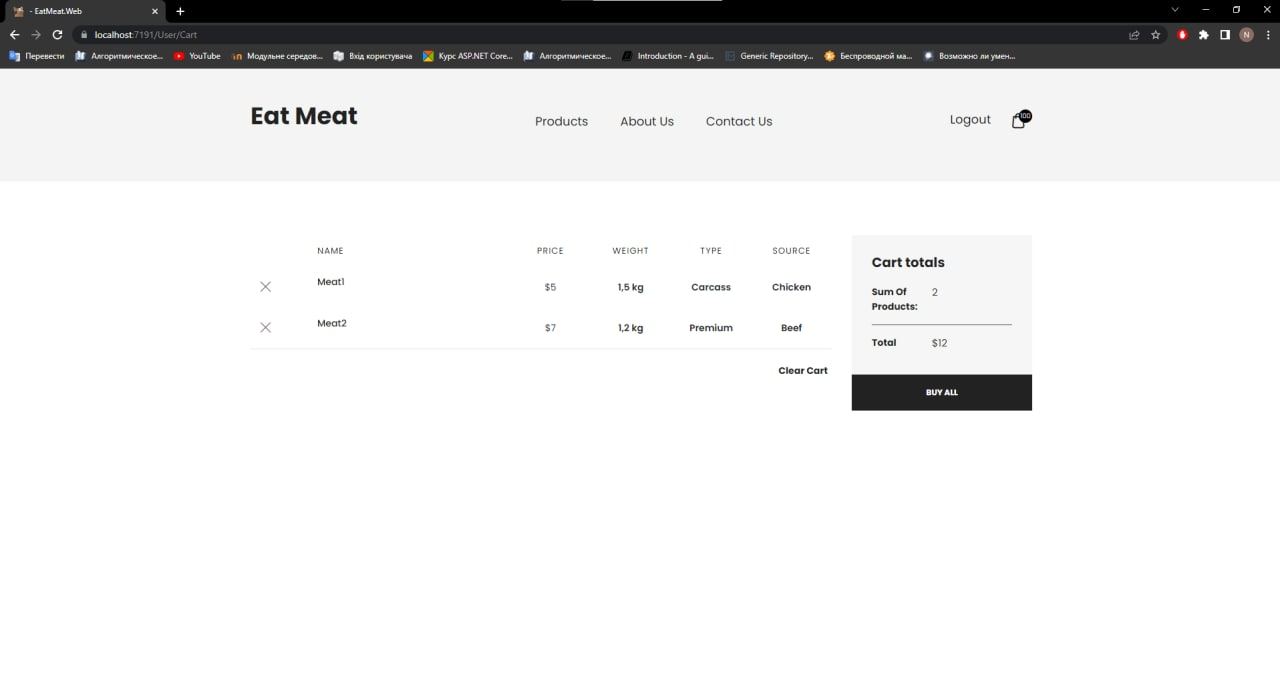


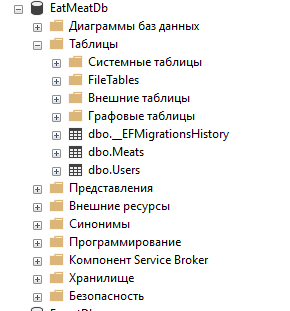












Висновок

Виходячи із даної проєктно-технологічної роботи ASP.NET Core представляє технологію для створення веб-додатків на платформі .NET, яку розвиває компанія Microsoft. Як мови програмування розробки додатків на ASP.NET Core використовуються C# і F#.

Історія ASP.NET фактично почалася з виходом першої версії. . В той же час спочатку ASP.NET була націлена на роботу виключно в Windows на веб-сервері IIS (хоча завдяки проекту Mono програми на ASP.NET можна було запускати і на Linux).

ASP.NET Core працює поверх платформи .NET і, таким чином, дозволяє використовувати весь її функціонал.

Підтримка роботи з більшістю поширених систем баз даних: MS SQL Server, MySQL, Postgres, MongoDB.

Завдяки модульності фреймворку всі необхідні компоненти веб-програми можуть завантажуватись як окремі модулі через пакетний менеджер Nuget.

ASP.NET Core представляє крос-платформний фреймворк, програми на якому можуть бути розгорнуті на всіх основних популярних операційних системах: Windows, Mac OS, Linux. І таким чином, за допомогою ASP.NET Core ми можемо як створювати крос-платформні програми на Windows, на Linux та Mac OS, так і запускати на цих ОС.

## Список використаних джерел

1. ASP .NET Tutorial. Metanit URL: <https://metanit.com/sharp/aspnet5/1.1.php>
2. ASP .NET Tutorial. Metanit URL: <https://metanit.com/sharp/aspnet5/7.6.php>
3. ASP .NET Tutorial. Microsoft URL: <https://docs.microsoft.com/ua-ua/dotnet/architecture/modern-web-apps-azure/work-with-data-in-asp-net-core-apps>
4. ASP .NET Tutorial. Metanit URL: https://metanit.com/sharp/aspnet5/5.1.php

## Додаток А

## Скріншоти сайту

