Міністерство освіти і науки України
Гусятинський фаховий коледж
Тернопільського національного технічного університету
імені Івана Пулюя

Спеціальність 121 Інженерія програмного забезпечення

Курс III Група П-31 Семестр 6

ЗВІТ

з навчальної практики

(Розробка програм на C# та технологія .NET)

Тема: «Розробка сайту для кондитерського магазину засобами ASP.NET MVC5.»

Виконав: Євген ГУТА

Викладач: Роман ЧАПЛІНСЬКИЙ

Гусятин, 2025

ЗМІСТ

ВСТУП

В сучасному світі стрімкий розвиток технологій посприяв на рівень створення нових технологій. Це дозволило втілювати різні різні ідеї та збільшило потенційні можливості реалізації складніших проєктів. Одною з таких технологій є онайн сайти для продажу певних товарів. Зручно та доступно замовляти покупку, можливість переглядати асортимент з функціями пошуку та сортування за вибраними параметрами, і не тільки.

Під час практики було створено чат сайт , який міг би надавати людям можливість перегляду предметів з списку товарів, додавати їх до корзини та редагувати збережене у самому кошику для покупок. Найосновнішою рисою сайту є його простота у використанні та функціонал.

Проєкт був розроблений в середовищі програмування Visual Studio 2017 на платформі ASP.NET, що дозволило створити багатофункціональний сайт для зручних покупок онлайн.

1 ПОСТАНОВКА ТА ЗМІСТ ЗАВДАННЯ

1.1 Створення технічного завдання на розробку проєкту

На етапі проєктування, перш за все, слід визначити призначення майбутнього сайту. Тому доцільно встановити тематичність його сторінок, проаналізувати задачі проєкту, потреби користувачів та сформулювати функціональність, структуру та вміст сторінок сайту, описати кожен блок функціональності та деталізувати його до рівня окремих елементів. Визначити найскладніші та найважливіші сторінки та сформувати для них список функціональності й відсортувати його по пріоритетом.

На основі проведеного аналізу методів та засобів розробки додатку, було сформовано технічне завдання на його розробку. Створити додаток для магазину кондитерської в якому повинна міститися наступна інформація:

- дані про товари;
- можливість регістрації для нових користувачів;
- вхід в особистий профіль;
- перегляд товарів у кошику;
- можливість редагування вмісту кошика;
- змога переглядати, додавати та редагувати нові дані про продукцію, покупців та користувачів з профілю адміністратора.

1.2 Опис варіантів використання

Розроблений програмний засіб складається з основних файлів та бібліотек, які описують архітектуру та структуру вебсайту.

Опис основних файлів наведений в таблиці 1.1

Таблиця 1.1 – Опис файлів проєкту

№ п/п	Назва файлу	Короткий опис
1	2	3
1.	AccountController.cs	Файл обробки інформації входу або реєстрації в акаунт
2.	AdminController.cs	Файл обробки інформації до якої має доступ адміністратор
3.	_Layaout.cshtml	Файл для створення шаблону сайту
4.	Index.cshtml	Головна сторінка додатку для сайту
5.	HomeController.cs	Файл обробки і виведення інформації про товари
6.	BuyerController.cs	Файл обробки і виведення інформації про покупців

Для відображення майбутньої архітектури програмного забезпечення та відображення взаємодії кінцевого користувача з сайтом. Визначення акторів та варіанти використання створюється за допомогою use-case діаграми.

Для роботи з вебсайтом використовуємо такі спеціалізації:

– Користувач (Покупець);

– Адміністратор;

Користувач вебсайту має можливість переглянути список товарів, та додати товар у кошик. Як і Покупець, Адміністратор має ті самі права, але додатково з’являється можливість введення нової інформації про товар або редагування вже існуючого (див. Рис.1.1).

Рисунок 1.1 – Діаграма варіантів користування вебсайтом.

В результаті роботи над першим розділом було сформоване технічне завдання, що дало змогу визначити усі функції, які повинні були бути реалізовані у вебсайті.

2 ОПИС РЕАЛІЗАЦІЇ ЗАВДАННЯ

2.1 Основні етапи створення проекту

Вебсайт чату реалізований за допомогою ASP.NET MVC5 та jQuery.

ASP.NET – це частина технології. NET, яка використовується для написання потужних клієнт-серверних інтернет додатків. Вона дозволяє створювати динамічні сторінки HTM, ASP.NET виникла в результаті об’єднання старішої технології ASP (активні серверні сторінки) і NET Framework. Вона містить безліч готових елементів управління, використовуючи які можна швидко створювати інтерактивні web-сайти. Загалом, можливості ASP.NET обмежені тільки вашою уявою [1], [2], [5].

Платформа ASP.NET MVC представляє собою фреймворк для створення сайтів і веб-додатків за допомогою реалізації паттерна MVC.

Концепція паттерна (шаблону) MVC (model - view - controller) передбачає поділ додатка на три компоненти:

Контролер (Controller) представляє клас, що забезпечує зв’язок між користувачем і системою, представленням і сховищем даних. Він отримує дані, що вводяться користувачем і обробляє їх. І в залежності від результатів обробки відправляє користувачеві певний висновок, наприклад, у вигляді представлення.

Представлення (View) – це власне візуальна частина або призначений для користувача інтерфейс програми. Як правило, html-сторінка, яку користувач бачить, зайшовши на сайт.

Модель (Model) представляє клас, що описує логіку використовуваних даних.

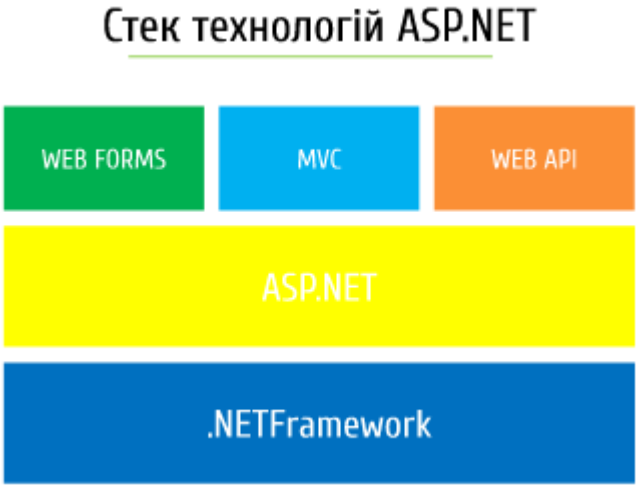


Рисунок 2.1 – Стек технологій

jQuery є популярною бібліотекою JavaScript, яка значно спрощує розробку динамічних веб-сторінок завдяки зручному API для маніпуляцій з DOM, обробки подій та виконання анімацій. Використання jQuery в ASP.NET надає можливість створювати більш інтерактивні та зручні для користувача веб-додатки, поєднуючи переваги обох технологій [3].

jQuery була створена Джоном Резіґом у 2006 році і швидко здобула популярність серед веб-розробників завдяки своїй простоті та ефективності. Основні можливості jQuery включають:

- Просте та ефективне маніпулювання DOM.
- Інтуїтивна обробка подій.
- Анімації та ефекти.
- Підтримка крос-браузерності.

Інтеграція jQuery з ASP.NET дозволяє розробникам використовувати можливості обох технологій для створення ефективних та інтерактивних веб-додатків. Тезнологію також можна ефективно використовувати в ASP.NET WebForms для покращення користувацького досвіду. Наприклад, можна використовувати jQuery для валідації форм перед відправкою даних на сервер.

2.2 Засоби реалізації проекту

Для розробки проєкту було використано платформу ASP.NET MVC5 з використанням jQuery.

Створений вебсайт, можна буде використовувати для вибору та покупки комп’ютерних комплектуючих, що реалізовано технологіями HTML, CSS, C#[4]. Основним шаблоном для веб-застосунку служить _Layout.cshtml, що відображає вигляд та елементи взаємодії сайту.

На головній сторінці для користувача є змога переглянути товари з контерської та додати їх до кошику, перйти до вкладки Кошик, де можна переглянути додані предмети та редагувати дані, котрі є в кошику. Для адміністратора є змога додавати, редагувати та видаляти нові товари, покупців та наявних користувачів у системі.

3 ОПИС ВИХІДНИХ ЕКРАННИХ ФОРМ

Під час завантаження вебсайту відривається головна сторінка, на якій з’являються: таблиця з списком продукції, навігаційна панель сайту, де є змога переходити між вкладками та кнопка входу в особистий акаунт користувача та реєстрації (див. Рис. 3.1).

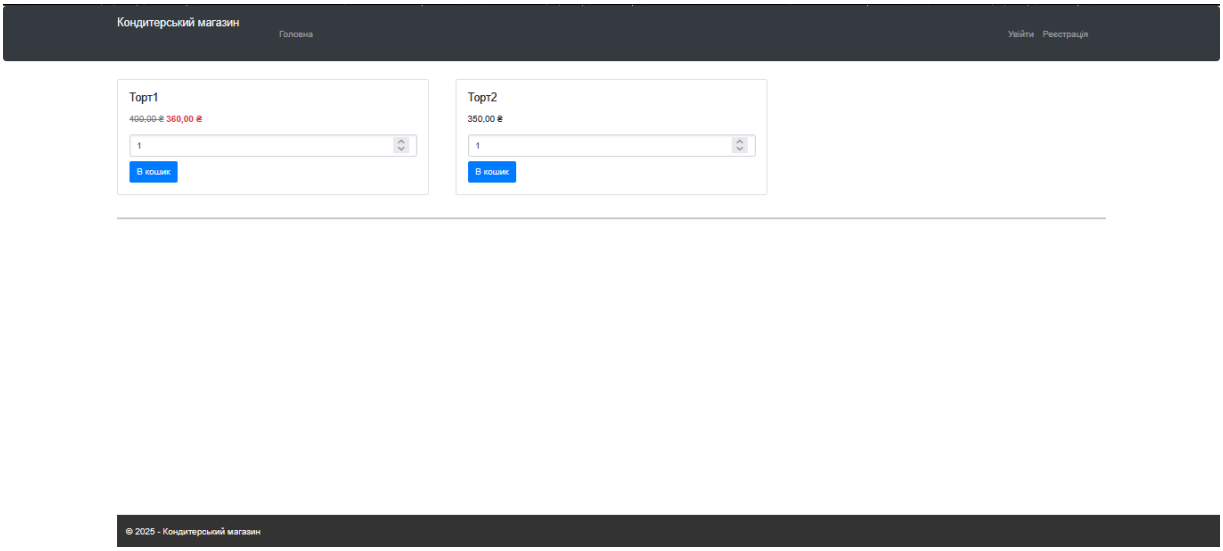


Рисунок 3.1 – Вікно головної сторінки вебсайту.

Для адміністратора вигляд початкової сторінки буде наступний (див. рис. 3.2).

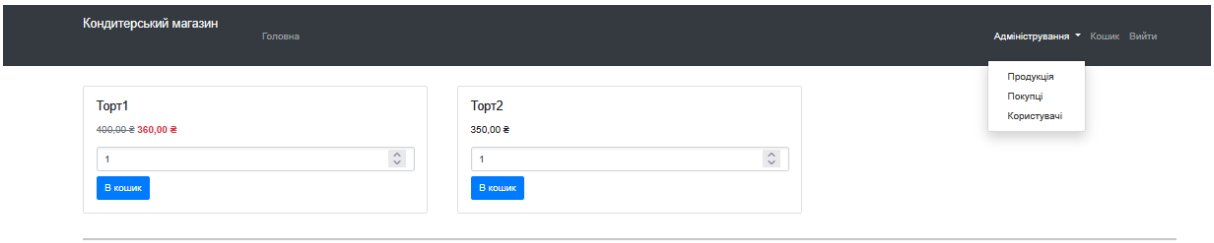


Рисунок 3.2 – Головна сторінка вебсайту.

Якщо користувач не зареєстрований і намагатиметься зайти у кошик, то відбудеться перехід у вкладку входу (див. рис. 3.3).

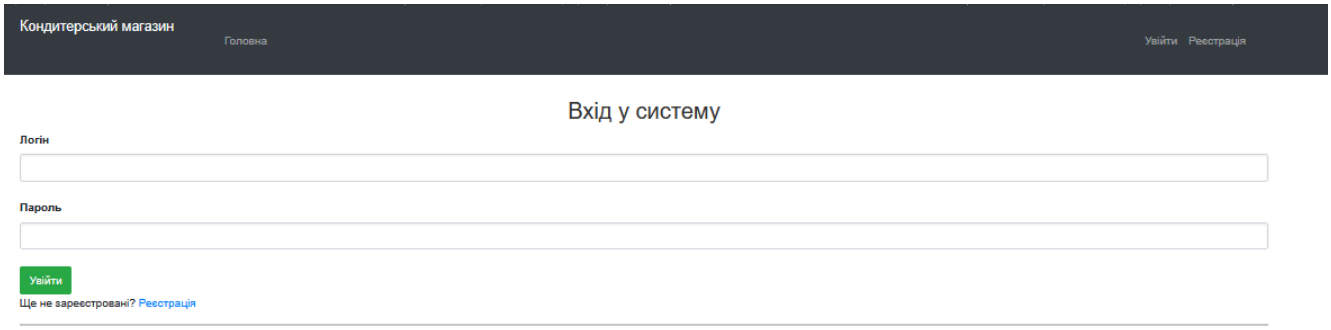


Рисунок 3.3 – Перехід на вкладку входу в особистий акаунт.

Якщо відвідувач не має особистого профіля, тоді натискаючи на кнопку «Регістрація», відвідувачу відкриється наступна вкладка (див. рис. 3.4).

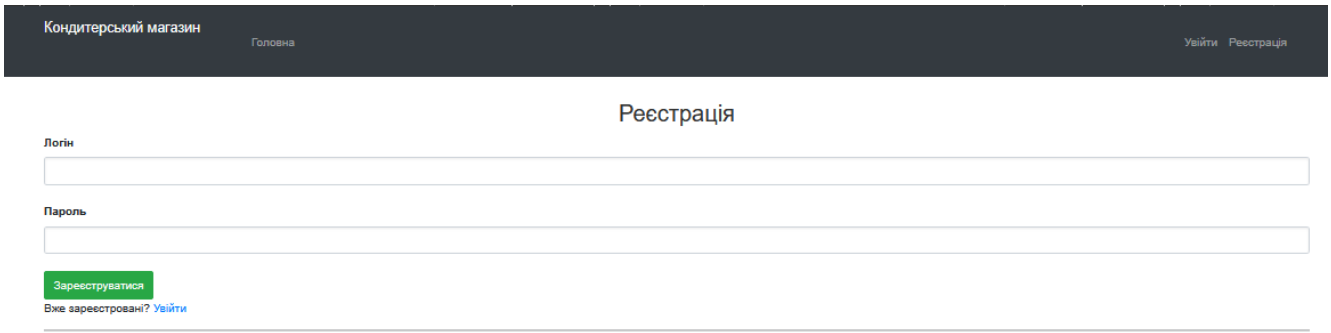


Рисунок 3.4 – Результат надсилання повідомлення користувачами.

Після входу в особистий профіль, як користувач або як адміністратор, буде доступ до додавання товарів у корзину. Вигляд порожнього кошика виглядає наступним чином (див. рис. 3.5).

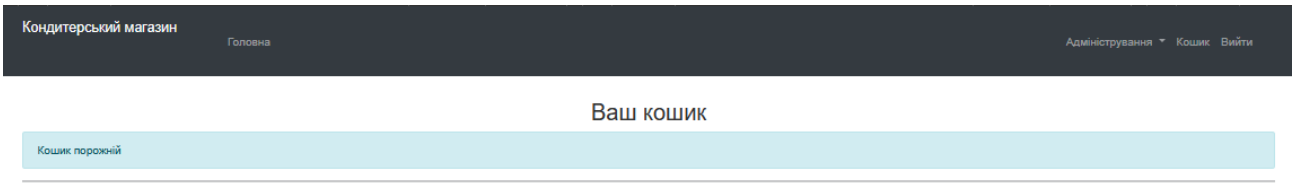


Рисунок 3.5 – Вікно вкладки «Кошик» без доданих даних у нього.

Вкладка «Кошик» зі заповненими даними (див. рис. 3.6).

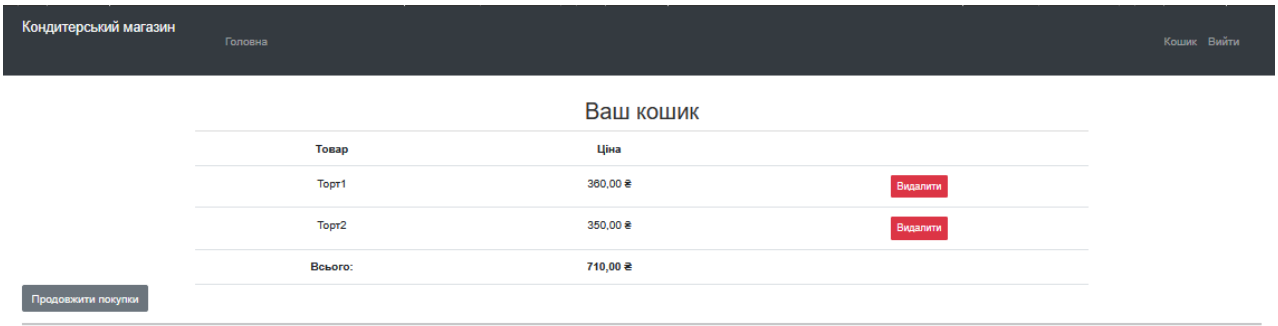


Рисунок 3.6 – Вікно вкладки «Кошик» з доданими данними.

Також, передбачено виняткові ситуації, в разі введення невірних даних. Приклад виведення такого повідомлення наступний (див. рис. 3.7).

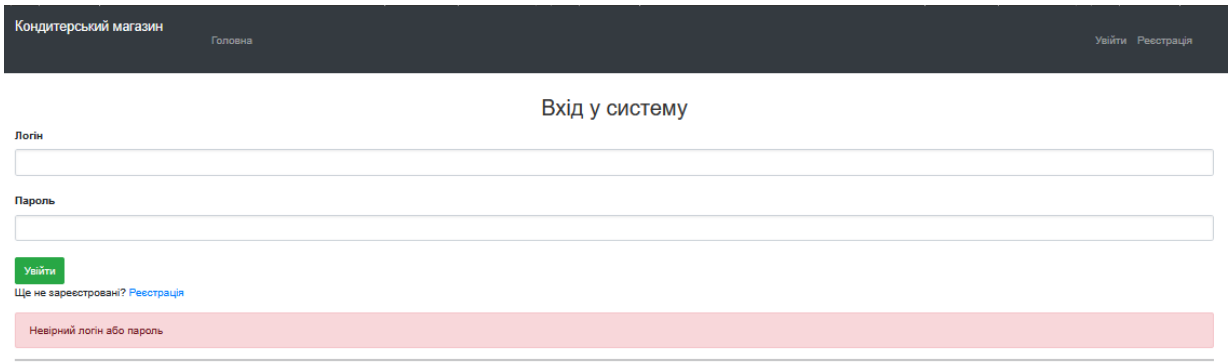


Рисунок 3.7 – Приклад виняткової ситуації.

Розроблений вебсайт зручний у використанні з простим дизайном, зручним для користувачів. Інтуїтивний інтерфейс дозволяє користувачам легко робити покупку товарів з кондитерського магазину.

4 ОПИС РЕЗУЛЬТАТІВ ТЕСТУВАННЯ ПРОЄКТУ

4.1 План тестування

Тестування вебсайту – це процес який завершує створення проєкту. Тестування передбачає добре сплановану структуру і відпрацьовану послідовність дій [8].

Метою проведення тестування – є всебічна перевірка між очікуваною роботою сайту та відповідністю реальної продуктивності пропонованим вимогам [11]. Для досягнення цієї мети тестувальники виконують такі завдання:

– створення штучних ситуацій, які потенційно можуть виникнути в процесі роботи;

– моніторинг процесів виконання цільових дій всередині сайту під дією «стимулів»;

Існує безліч видів тестування, котрі відрізняються як за рівнем аналізу, так і за різними критеріями якості роботи. Зупинимося на основних аспектах, які дозволять покращити роботу сайту і навіть дати йому нове життя, а саме:

- функціональне тестування;
- ручне тестування сайту;

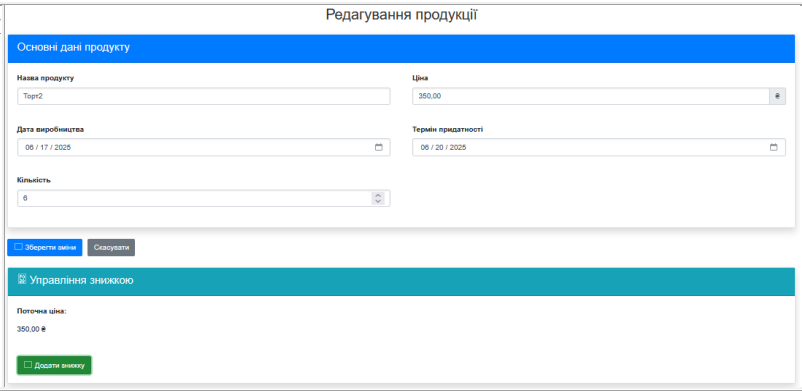
Інфраструктура ASP.NET MVC Framework спроектовано для максимального полегшення підготовки автоматизованих тестів ASP.NET MVC надає ідеальну платформу для автоматизованих тестувань, а в Visual Studio надає ряд зручних засобів тестування.

Для перевірки коректності роботи програми необхідно здійснити її тестування.

Ручне тестування можна розглядати як взаємодію професійного тестувальника і софтвера з метою пошуку багів. Таким чином, під час ручного тестування можна отримувати відгук, що неможливо при автоматизованій перевірці. Іншими словами, взаємодіючи з додатком безпосередньо, той хто тестує

може порівнювати очікуваний результат з реальним і залишати рекомендації Опис такого тестування наведено у таблиці 4.1.

Таблиця 4.1 – Створення Bug-report для вебсайту

Додаток чату «OfficeStore»	
1	**2**
Короткий опис (Summary)	Несправність кнопки додання знижки на продукцію
Проект (Project)	**Навчальна_Практика_Гута**
Компонет додатку (Component)	EditProduction.cshtml
Номер версії (Version)	1.1.0.0
Серйозність (Severity)	S2 Критичний (Critical)
Пріоритет (Priority)	P1 (High)
Статус (Status)	Повторний
Автор (Author)	Гута Євген
Призначений (Assigned To)	Розробнику – Гуті Євгену
Середовище	
ОС / Сервіс Пак і т.д. / Браузер + версія / ...	Windows 10 Pro 21H2, .NET Framework 4.7.2, Intel(R) Core(TM) i5-10300H, ОЗП 16ГБ, Chrome 126.0.6478.57
...	
Опис	
Кроки відтворення (Steps to Reproduce)	1. Зайти як адміністратор 2. Зайти у вкладку для редагування даних про товар 3. Відкрити панель редагування товару 4. Натиснути на кнопку додання знижки на продукцію
Фактичний результат (Result)	Не з’являється модальне вікно про додання знижки на товар
Очікуваний результат (Expected Result)	З’являється модальне вікно про додання знижки на товар та зі змогою задати відповідні дані
Доповнення	
Прикріпленні файли (Attachment)	

Мета функціонального тестування – виявлення невідповідностей між реальною поведінкою реалізованих функцій та очікуваною поведінкою відповідно до специфікації і вимог. Текстові сценарії, що поєднують окремі тести, орієнтовані на перевірку якості розв'язку функціональних задач.

Функціональні тести створюються за зовнішніми специфікаціями функцій, проєктної інформації, що стосуються його функціональних характеристик і застосовуються на процесі комплексного тестування для визначення повноти реалізації функціональних задач і їхньої відповідності вхідним вимогам. Набір test-case наведений в таблиці 4.2 [9].

Таблиця 4.2 – Створення тест-кейсів для вебсайту

ID	**Тип**	**Опис**	**Очікувано**	**Реально**	**Pass/ Fail**
1	**2**	**3**	**4**	**5**	**6**
1	positive	Перевірка відображення на різних розширеннях монітора	Усі компоненти знаходяться на своїх місцях, немає ніяких спотворень інтерфейсу	Усі компоненти знаходяться на своїх місцях, немає ніяких спотворень інтерфейсу	PASS
2	positive	Функція відображення паролю char *	При введенні паролю всі символи відображаються у вигляді *	При введенні паролю всі символи відображаються у вигляді *	PASS
3	positive	Функція переходу по компонентах клавішею Tab (function button Tab)	Повинен здійснюватися перехід курсора між полями для вводу та кнопками	При переході функція клавіші Tab працює у всіх випадках	PASS
4	negative	В поле логіну користувача введено невірний логін	Поле з помилкою	Поле з помилкою	PASS
5	negative	В поле паролю користувача введено невірний пароль	Поле з помилкою	Поле з помилкою	PASS
6	negatev e	При реєстрації нового користувача заповнено не усі поля	Поле з помилкою	Поле з помилкою	PASS

Продовження таблиці 4.2

1	2	3	4	5	6
7	positive	Усі поля при реєстрації користувача коректно заповнені	Повідомлення про успішну реєстрацію	Повідомлення про успішну реєстрацію	PASS
8	positive	В поле логіну і паролю введено вірні данні	Вхід в систему відповідно до облікового запису	Вхід в систему відповідно до облікового запису	PASS
9	positive	Вхід під обліковим записом користувача	Доступні лише ті функції програми, які визначенні правами доступу	Доступні лише ті функції програми, які визначенні правами доступу	PASS

Тест кейс 1:

Перевірка відображення сторінки		
Дія	**Очікуваний результат**	**Результат тесту**
1	**2**	**3**
Відкрити сторінку «Index.cshtml»	– Форма «Index.cshtml» відкрита; – Назва форми – Index.cshtml; – На формі головна сторінка сайту.	PASS

Опис тест кейсу 1:

Назва: Перевірка відображення сторінки

Дія: Відкрити сторінку «Index.cshtml»

Перевірка: Перевірити, щоб сторінка яка відображається співпадала з сторінкою, що на рисунку 4.1.

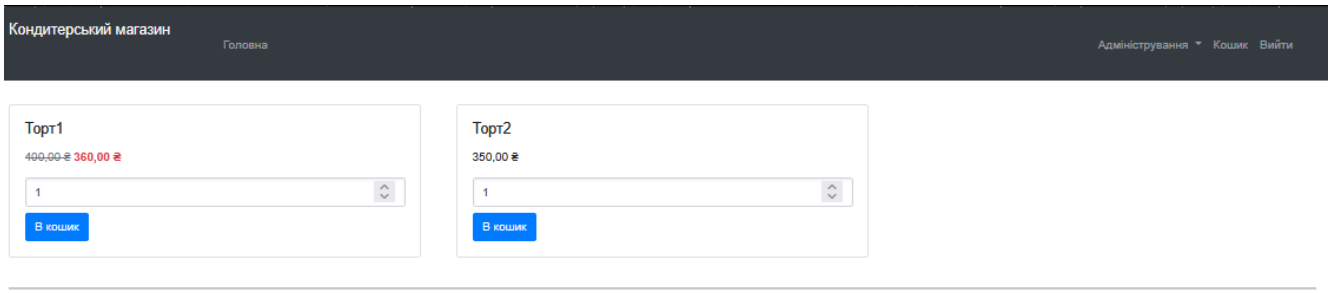


Рисунок 4.1 – Відображення сторінки «Index.cshtml»

Тест кейс 2:

Перевірка відображення сторінки		
Дія	**Очікуваний результат**	**Результат тесту**
1	**2**	**3**
Відкрити сторінку «Basket.cshtml»	– Форма «Basket.cshtml» відкрита; – Назва форми – Basket.cshtml; – На сторінці товар у кошику.	PASS

Опис тест кейсу 2:

Назва: Перевірка відображення сторінки

Дія: Відкрити сторінку «Basket.cshtml»

Перевірка: Перевірити, щоб сторінка коректно відображає дані у кошику на рисунку 4.2.

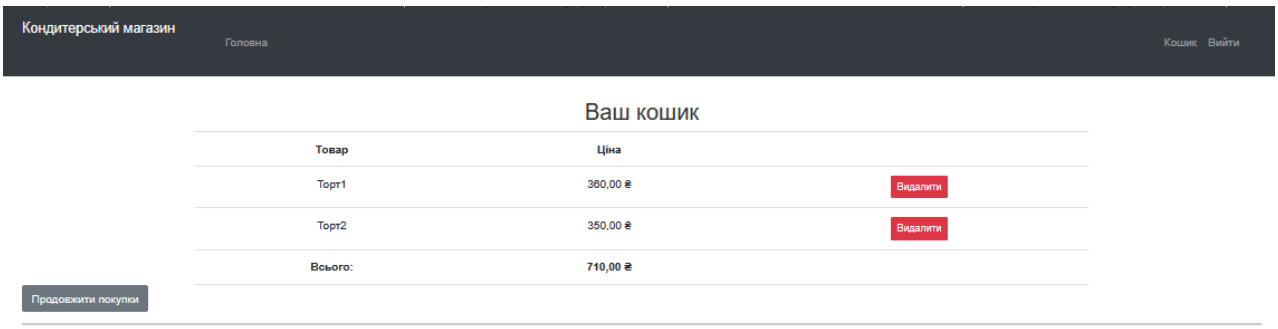


Рисунок 4.2 – Відображення сторінки «Basket.cshtml»

З проведеного тестування розробленого вебсайту можна зробити висновок, що програма без помилок не існує, тому тестування потрібно виконувати на кожному етапі.

ВИСНОВКИ

Веб застосунок, який створений під час навчальної практики, може бути використаний для продажу товарів з магазину кондитерських виробів.

За допомогою вебсайту можна здійнювати вибір у корзину товару, редагування інформації про продукцію, покупців та користувачів.

Вебсайт реалізований в середовищі Visual Studio 2017 на мові C# з використанням технології ASP.NET та jQuery і має перспективи розвитку у майбутньому при умові визначення додаткового функціоналу.

ПЕРЕЛІК ПОСИЛАНЬ

1. Реалізація в проекті ASP .NET MVC5: https://www.c-sharpcorner.com/article/simple-login-application-using-Asp-Net-mvc/ (Дата звернення 04.06.2024).

2. Мова Visual Studio C#. URL: https://programm.top/uk/c-sharp/tutorial/introduction/ (Дата звернення 05.06.2024).

3. jQuery Tutorial | Learn jQuery Online Free. – URL: https://www.geeksforgeeks.org/jquery-tutorial/ (Дата звернення 8.06.2022).

4. Основи програмування на мовах Ci та Ci # для початківців. – URL: http://cppstudio.com/ (Дата звернення 8.06.2024)

5. Основи ASP.NET: https://hyperhost.ua/uk/wiki/asp-net-hosting (Дата звернення 03.06.2024).

6. Мови розмітки HTML, CSS. URL: http://fcit.tneu.edu.ua/web-rozrobka/html-css (Дата звернення 11.06.2024).

7. Мова Visual Studio C#. URL: https://programm.top/uk/c-sharp/tutorial/introduction/ (Дата звернення 8.06.2024).

8. Все про Web Forms. URL: https://docs.microsoft.com/ru-ru/aspnet/web-forms/what-is-web-forms (Дата звернення 09.06.2024).

9. Інформація про тестування вебсайтів. URL: https://brainlab.com.ua/uk/blog-uk/yak-testuvati-veb-sayt-osnovn-etapi-poradi (Дата звернення 07.06.2024).

10. Тест кейси розгорнута інформація. URL: http://www.protesting.ru/testing/testcase.html (Дата звернення 11.06.2024).

11. Тестування програмного забезпечення. URL: https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення/ (Дата звернення 10.06.2024).

ДОДАТКИ

Додаток А

Лістинг додатку

HomeController.cs

```
using System;
using System.Linq;
using System.Data.Entity;
using System.Web.Mvc;
using Навчальна_практика_Гута.Models;

public class HomeController : Controller
{
    private ConfectionerShopDBEntities db = new ConfectionerShopDBEntities();

    public ActionResult Index()
    {
        var products = db.Production
            .Include(p => p.Discount)
            .Where(p => p.Quantity > 0)
            .ToList();

        return View(products);
    }

    [HttpPost]
    public ActionResult AddToCart(int productId, int quantity)
    {
        if (Session["UserId"] == null) return RedirectToAction("Login", "Account");

        var userId = (int)Session["UserId"];
        var buyer = db.Buyer.FirstOrDefault(b => b.UserId == userId);
        if (buyer == null) return RedirectToAction("Create", "Buyer");

        var product = db.Production.Find(productId);
        if (product == null || product.Quantity < quantity)
        {
            TempData["ErrorMessage"] = "Недостатня кількість товару";
            return RedirectToAction("Index");
        }

        // Отримуємо першу активну знижку для продукту
        var discount = db.Discount.FirstOrDefault(d => d.ProductionId == productId);

        // Створюємо запис у кошику
        var basketItem = new Basket
        {
            ProductName = product.ProductName,
            DiscountedPrice = discount != null ? discount.DiscountedPrice : product.Price,
            BuyerFirstName = buyer.FirstName,
            BuyerLastName = buyer.LastName,
            BuyerId = buyer.Id,
            DiscountId = discount?.Id
        };

        db.Basket.Add(basketItem);
        product.Quantity -= quantity;
        db.SaveChanges();

        TempData["SuccessMessage"] = product.ProductName + " додано до кошика";
        return RedirectToAction("Index", "Basket");
    }}
```

Продовження додатку А

AccountControler.cs

```
using System;
using System.Collections.Generic;
using System.Web.Mvc;
using System.Linq;
using Навчальна_практика_Гута.Models;

public class AccountController : Controller
{
    private ConfectionerShopDBEntities db = new ConfectionerShopDBEntities();

    // GET: Account/Login
    public ActionResult Login()
    {
        return View();
    }

    // POST: Account/Login
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult Login(string login, string password)
    {
        var user = db.User.FirstOrDefault(u => u.Login == login && u.Password == password);
        if (user != null)
        {
            Session["UserId"] = user.Id;
            Session["UserRole"] = DetermineUserRole(user.Login);

            return RedirectToAction("Index", "Home");
        }
        ViewBag.ErrorMessage = "Невірний логін або пароль";
        return View();
    }

    // GET: Account/Register
    public ActionResult Register()
    {
        return View();
    }

    // POST: Account/Register
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult Register(User model)
    {
        if (ModelState.IsValid)
        {
            db.User.Add(model);
            db.SaveChanges();
            return RedirectToAction("Login");
        }
        return View(model);
    }

    public ActionResult Logout()
    {
        Session["UserId"] = null;
        Session["UserRole"] = null;
        return RedirectToAction("Index", "Home");
    }
```

Продовження додатку А

```
    private string DetermineUserRole(string login)
    {
        if (login.Contains("Admin") || login.Contains("admin")) return "Admin";
        return "Buyer";
    }
}
```

_Layout.cshtml

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title - Кондитерський магазин</title>
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.2/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/Scripts/pageupdate.js"></script>
    @Styles.Render("~/Content/css")
    @Scripts.Render("~/bundles/modernizr")
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css">
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
icons/1.10.0/font/bootstrap-icons.min.css">
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
    <link rel="stylesheet" href="~/Content/Site.css" />
    @RenderSection("Styles", required: false)
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <div class="container">
            @Html.ActionLink("Кондитерський магазин", "Index", "Home", new { area = "" }, new
{ @class = "navbar-brand" })
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle
navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav me-auto">
                    <li class="nav-item">
                        @Html.ActionLink("Головна", "Index", "Home", null, new { @class =
"nav-link" })
                    </li>
                </ul>
                <ul class="navbar-nav">
                    @if (Session["UserRole"] != null)
                    {
                        var userRole = Session["UserRole"].ToString();
                        if (userRole == "Admin")
                        {
                            <li class="nav-item dropdown">
                                <a class="nav-link dropdown-toggle" href="#"
id="adminDropdown" role="button" data-bs-toggle="dropdown" aria-expanded="false">
                                    Адміністрування
```

Продовження додатку А

```
                                </a>
                                <ul class="dropdown-menu dropdown-menu-end" aria-
labelledby="adminDropdown">
                                    <li>@Html.ActionLink("Продукція", "Products", "Admin",
null, new { @class = "dropdown-item" })</li>
                                    <li>@Html.ActionLink("Покупці", "Buyers", "Admin", null,
new { @class = "dropdown-item" })</li>
                                    <li>@Html.ActionLink("Користувачі", "Users", "Admin",
null, new { @class = "dropdown-item" })</li>
                                </ul>
                            </li>
                        }
                        <li class="nav-item">
                            @Html.ActionLink("Кошик", "Index", "Basket", null, new { @class =
"nav-link" })
                        </li>
                        <li class="nav-item">
                            @Html.ActionLink("Вийти", "Logout", "Account", null, new { @class
= "nav-link" })
                        </li>
                    }
                    else
                    {
                        <li class="nav-item">
                            @Html.ActionLink("Увійти", "Login", "Account", null, new { @class
= "nav-link" })
                        </li>
                        <li class="nav-item">
                            @Html.ActionLink("Реєстрація", "Register", "Account", null, new {
@class = "nav-link" })
                        </li>
                    }
                </ul>
            </div>
        </div>
    </nav>
    <div class="container body-content">
        @RenderBody()
        <hr />
        <footer>
            <p>&copy; @DateTime.Now.Year - Кондитерський магазин</p>
        </footer>
    </div>
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>
```

Index.cshtml

```
@model IEnumerable<Навчальна_практика_Гута.Models.Production>

@{
    var db = new Навчальна_практика_Гута.Models.ConfectionerShopDBEntities();
}
```

Продовження додатку А

```
<div class="row">
    @foreach (var product in Model)
    {
        var discount = db.Discount.FirstOrDefault(d => d.ProductionId == product.Id);
        <div class="col-md-4 mb-4">
            <div class="card">
                <div class="card-body">
                    <h5 class="card-title">@product.ProductName</h5>
                    <p class="card-text">
                        @if (discount != null)
                        {
                            <span class="text-muted text-decoration-line-
through">@product.Price.ToString("C")</span>
                            <span class="text-danger fw-bold">
@discount.DiscountedPrice.ToString("C")</span>
                        }
                        else
                        {
                            @product.Price.ToString("C")
                        }
                    </p>
                    @using (Html.BeginForm("AddToCart", "Home", FormMethod.Post))
                    {
                        @Html.Hidden("productId", product.Id)
                        <input type="number" name="quantity" value="1" min="1"
max="@product.Quantity" class="form-control mb-2">
                        <button type="submit" class="btn btn-primary">В кошик</button>
                    }
                </div>
            </div>
        </div>
    }
</div>
```

Error.cshtml

```
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Error</title>
</head>
<body>
    <hgroup>
        <h1>Error.</h1>
        <h2>An error occurred while processing your request.</h2>
    </hgroup>
</body>
</html>
```

Basket.cshtml

```
@model IEnumerable<Навчальна_практика_Гута.Models.Basket>

@{
    ViewBag.Title = "Ваш кошик";
```

Продовження додатку А

```
}
<h2>Ваш кошик</h2>
@if (!Model.Any())
{
    <div class="alert alert-info">Кошик порожній</div>
}
else
{
    <table class="table">
        <tr>
            <th>Товар</th>
            <th>Ціна</th>
            <th></th>
        </tr>
        @foreach (var item in Model)
        {
            <tr>
                <td>@item.ProductName</td>
                <td>@item.DiscountedPrice.ToString("C")</td>
                <td>
                    @Html.ActionLink("Видалити", "Remove", new { id = item.Id },
                        new { @class = "btn btn-danger btn-sm" })
                </td>
            </tr>
        }
        <tr>
            <th>Всього:</th>
            <th>@Model.Sum(i => i.DiscountedPrice).ToString("C")</th>
            <th></th>
        </tr>
    </table>

    <div>
        @Html.ActionLink("Продовжити покупки", "Index", "Home", null, new { @class = "btn btn-
secondary" })
    </div>
}
```

AdminControler.cs

```
using System;
using System.Linq;
using System.Web.Mvc;
using System.Data.Entity;
using Навчальна_практика_Гута.Models;

public class AdminController : Controller
{
    private ConfectionerShopDBEntities db = new ConfectionerShopDBEntities();

    [AuthorizeAdmin]
    public ActionResult Products()
    {
        var products = db.Production
            .Include(p => p.Discount)
            .ToList();

        return View(products);
    }

    [AuthorizeAdmin]
```

Продовження додатку А

```
public ActionResult CreateProduct()
{
    return View();
}

[HttpPost]
[ValidateAntiForgeryToken]
[AuthorizeAdmin]
public ActionResult CreateProduct(Production product)
{
    if (ModelState.IsValid)
    {
        db.Production.Add(product);
        db.SaveChanges();
        TempData["SuccessMessage"] = "Продукцію успішно додано";
        return RedirectToAction("Products");
    }
    return View(product);
}

[AuthorizeAdmin]
public ActionResult EditProduct(int id)
{
    var product = db.Production.Find(id);
    if (product == null)
    {
        return HttpNotFound();
    }
    return View(product);
}

[HttpPost]
[ValidateAntiForgeryToken]
[AuthorizeAdmin]
public ActionResult EditProduct(Production product)
{
    if (ModelState.IsValid)
    {
        db.Entry(product).State = EntityState.Modified;
        db.SaveChanges();
        TempData["SuccessMessage"] = "Продукцію успішно оновлено";
        return RedirectToAction("Products");
    }
    return View(product);
}

[AuthorizeAdmin]
public ActionResult DeleteProduct(int id)
{
    // Починаємо транзакцію
    using (var transaction = db.Database.BeginTransaction())
    {
        try
        {
            // 1. Знаходимо продукт
            var product = db.Production.Find(id);
            if (product == null)
            {
                return HttpNotFound();
            }

            // 2. Знаходимо знижку, пов'язану з продуктом
            var discount = db.Discount.FirstOrDefault(d => d.ProductionId == product.Id);
```

Продовження додатку А

```
                if (discount != null)
                {
                    // 3. Знаходимо всі записи кошика, пов'язані зі знижкою
                    var basketItems = db.Basket.Where(b => b.DiscountId == 
discount.Id).ToList();

                    // 4. Видаляємо записи кошика
                    db.Basket.RemoveRange(basketItems);

                    // 5. Видаляємо знижку
                    db.Discount.Remove(discount);
                }

                // 6. Видаляємо сам продукт
                db.Production.Remove(product);

                // 7. Зберігаємо зміни
                db.SaveChanges();

                // 8. Підтверджуємо транзакцію
                transaction.Commit();

                TempData["SuccessMessage"] = "Продукцію та всі пов'язані дані успішно видалено";
            }
            catch (Exception ex)
            {
                // Відкатуємо транзакцію у разі помилки
                transaction.Rollback();
                TempData["ErrorMessage"] = $"Помилка при видаленні: {ex.Message}";

                if (ex.InnerException != null)
                {
                    TempData["ErrorMessage"] += $" | {ex.InnerException.Message}";
                }
            }
        }

        return RedirectToAction("Products");
    }

    [HttpPost]
    [AuthorizeAdmin]
    public JsonResult AddDiscount(int productId, string discountName, decimal discountValue)
    {
        try
        {
            // Логування для відладки
            System.Diagnostics.Debug.WriteLine($"AddDiscount called with: 
productId={productId}, discountName={discountName}, discountValue={discountValue}");

            var product = db.Production.Find(productId);
            if (product == null)
            {
                return Json(new { success = false, message = "Продукт не знайдено" });
            }

            // Перевірка чи знижка вже існує
            if (db.Discount.Any(d => d.ProductionId == productId))
            {
                return Json(new { success = false, message = "Для цього продукту вже є знижка" 
});
            }
```

Продовження додатку А

```
            // Валідація розміру знижки
            if (discountValue <= 0 || discountValue > 100)
            {
                return Json(new { success = false, message = "Розмір знижки повинен бути від
0.01 до 100%" });
            }

            // Розрахунок ціни зі знижкою
            var discountValueDecimal = discountValue / 100;
            var discountedPrice = product.Price * (1 - discountValueDecimal);

            var discount = new Discount
            {
                DiscountName = discountName,
                DiscountValue = discountValueDecimal,
                DiscountedPrice = discountedPrice,
                ProductionId = productId
            };

            db.Discount.Add(discount);
            db.SaveChanges();

            System.Diagnostics.Debug.WriteLine("Discount added successfully");

            return Json(new { success = true, message = "Знижку успішно додано" });
        }
        catch (Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"Error in AddDiscount: {ex.Message}");
            return Json(new { success = false, message = ex.Message });
        }
    }

    [HttpPost]
    [AuthorizeAdmin]
    public JsonResult EditDiscount(int discountId, string discountName, decimal discountValue)
    {
        try
        {
            var discount = db.Discount.Find(discountId);
            if (discount == null)
            {
                return Json(new { success = false, message = "Знижка не знайдена" });
            }

            var product = db.Production.Find(discount.ProductionId);
            if (product == null)
            {
                return Json(new { success = false, message = "Продукт не знайдено" });
            }

            // Валідація розміру знижки
            if (discountValue <= 0 || discountValue > 100)
            {
                return Json(new { success = false, message = "Розмір знижки повинен бути від
0.01 до 100%" });
            }

            // Оновлення знижки
            discount.DiscountName = discountName;
            discount.DiscountValue = discountValue / 100;
            discount.DiscountedPrice = product.Price * (1 - discountValue / 100);
```

Продовження додатку А

```
        db.Entry(discount).State = EntityState.Modified;
        db.SaveChanges();

        return Json(new { success = true });
    }
    catch (Exception ex)
    {
        return Json(new { success = false, message = ex.Message });
    }
}

[HttpPost]
[AuthorizeAdmin]
public ActionResult RemoveDiscount(int id)
{
    try
    {
        var discount = db.Discount.Find(id);
        if (discount != null)
        {
            db.Discount.Remove(discount);
            db.SaveChanges();
            return Json(new { success = true });
        }
        return Json(new { success = false, message = "Знижка не знайдена" });
    }
    catch (Exception ex)
    {
        return Json(new { success = false, message = ex.Message });
    }
}

[AuthorizeAdmin]
public ActionResult Users(string search = null)
{
    var users = db.User.AsQueryable();

    if (!string.IsNullOrEmpty(search))
    {
        users = users.Where(u => u.Login.Contains(search));
    }

    return View(users.ToList());
}

[AuthorizeAdmin]
public ActionResult Buyers(string search = null)
{
    var buyers = db.Buyer.Include(b => b.User).AsQueryable();

    if (!string.IsNullOrEmpty(search))
    {
        buyers = buyers.Where(b =>
            b.FirstName.Contains(search) ||
            b.LastName.Contains(search));
    }

    ViewBag.Search = search;
    return View(buyers.ToList());
}

[AuthorizeAdmin]
public ActionResult BuyerDetails(int id)
```

Продовження додатку А

```
{
    var buyer = db.Buyer
        .Include(b => b.User)
        .Include(b => b.Basket)
        .FirstOrDefault(b => b.Id == id);

    if (buyer == null)
    {
        return HttpNotFound();
    }

    return View(buyer);
}

[AuthorizeAdmin]
public ActionResult EditUser(int id)
{
    var user = db.User.Find(id);
    if (user == null)
    {
        return HttpNotFound();
    }
    return View(user);
}

[HttpPost]
[ValidateAntiForgeryToken]
[AuthorizeAdmin]
public ActionResult EditUser(User user)
{
    if (ModelState.IsValid)
    {
        db.Entry(user).State = EntityState.Modified;
        db.SaveChanges();
        TempData["SuccessMessage"] = "Дані користувача успішно оновлено";
        return RedirectToAction("Users");
    }
    return View(user);
}

[AuthorizeAdmin]
public ActionResult MakeAdmin(int id)
{
    var user = db.User.Find(id);
    if (user == null)
    {
        return HttpNotFound();
    }

    if (!user.Login.Contains("Admin"))
    {
        user.Login = "Admin_" + user.Login;
        db.SaveChanges();
    }

    return RedirectToAction("Users");
}

[AuthorizeAdmin]
public ActionResult EditBuyer(int id)
{
    var buyer = db.Buyer.Find(id);
```

Продовження додатку А

```
        if (buyer == null)
        {
            return HttpNotFound();
        }
        return View(buyer);
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    [AuthorizeAdmin]
    public ActionResult EditBuyer(Buyer buyer)
    {
        if (ModelState.IsValid)
        {
            db.Entry(buyer).State = EntityState.Modified;
            db.SaveChanges();
            TempData["SuccessMessage"] = "Дані покупця успішно оновлено";
            return RedirectToAction("Buyers");
        }
        return View(buyer);
    }

    [AuthorizeAdmin]
    public ActionResult DeleteUser(int id)
    {
        var user = db.User.Find(id);
        if (user == null)
        {
            return HttpNotFound();
        }

        // Перевірка наявності пов'язаних записів
        var hasRelatedBuyers = db.Buyer.Any(b => b.UserId == id);
        if (hasRelatedBuyers)
        {
            TempData["ErrorMessage"] = "Неможливо видалити користувача, оскільки є пов'язані
записи покупців";
            return RedirectToAction("Users");
        }

        try
        {
            db.User.Remove(user);
            db.SaveChanges();
            TempData["SuccessMessage"] = "Користувача успішно видалено";
        }
        catch (Exception ex)
        {
            TempData["ErrorMessage"] = "Помилка при видаленні користувача: " + ex.Message;
        }

        return RedirectToAction("Users");
    }

    [AuthorizeAdmin]
    public ActionResult DeleteBuyer(int id)
    {
        var buyer = db.Buyer.Find(id);
        if (buyer == null)
        {
            return HttpNotFound();
        }
```

Продовження додатку А

```
        try
        {
            // Видалення пов'язаних записів кошика
            var basketItems = db.Basket.Where(b => b.BuyerId == id).ToList();
            db.Basket.RemoveRange(basketItems);

            db.Buyer.Remove(buyer);
            db.SaveChanges();
            TempData["SuccessMessage"] = "Покупця успішно видалено";
        }
        catch (Exception ex)
        {
            TempData["ErrorMessage"] = "Помилка при видаленні покупця: " + ex.Message;
        }

        return RedirectToAction("Buyers");
    }
}

public class AuthorizeAdminAttribute : AuthorizeAttribute
{
    protected override bool AuthorizeCore(System.Web.HttpContextBase httpContext)
    {
        var userRole = httpContext.Session["UserRole"] as string;
        return userRole == "Admin";
    }
}
```

Додаток Б

Екранні форми проєкту

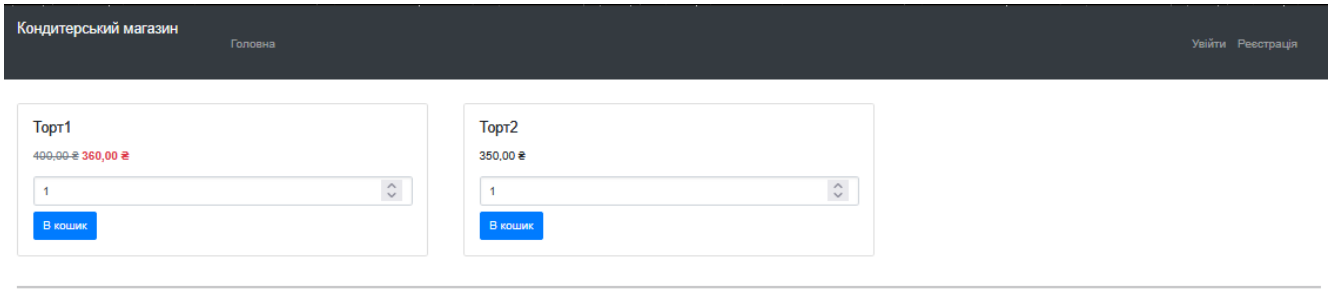


Рисунок Б.1 – Головна сторінка.

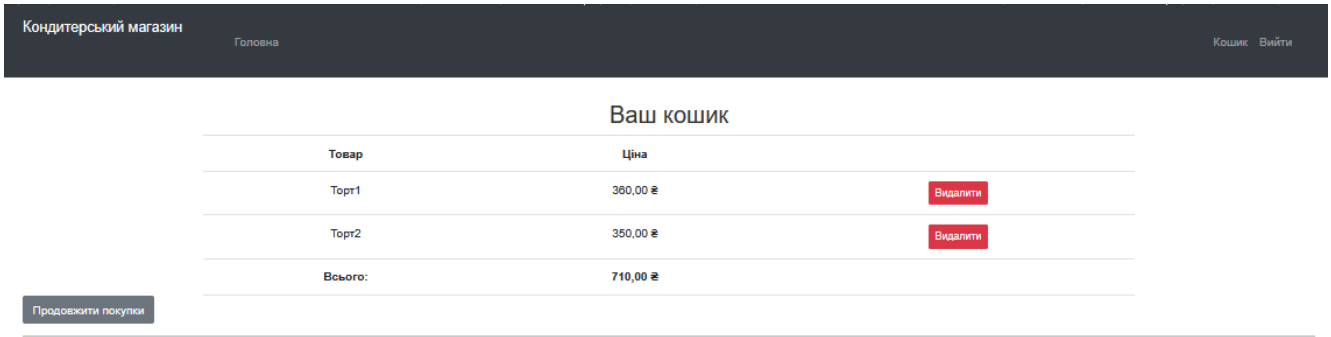


Рисунок Б.2 –Вікно «Кошик».

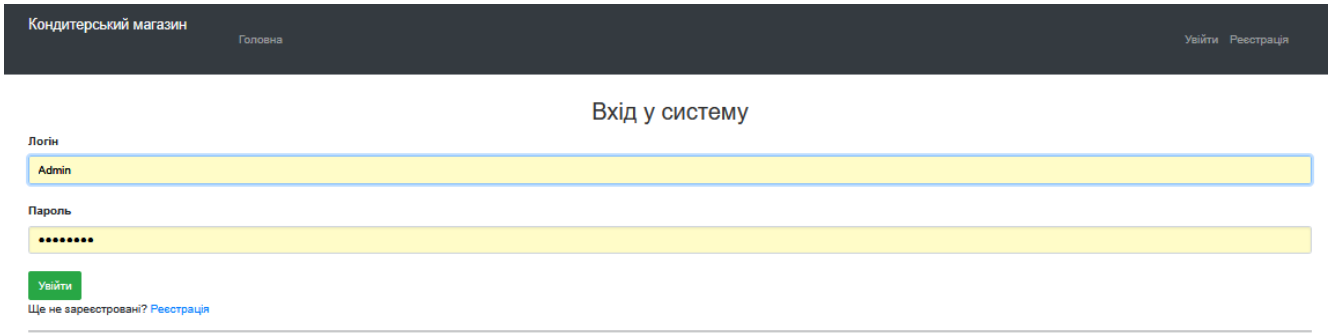


Рисунок Б.3 – Вікно входу у профіль.

Продовження додатку Б

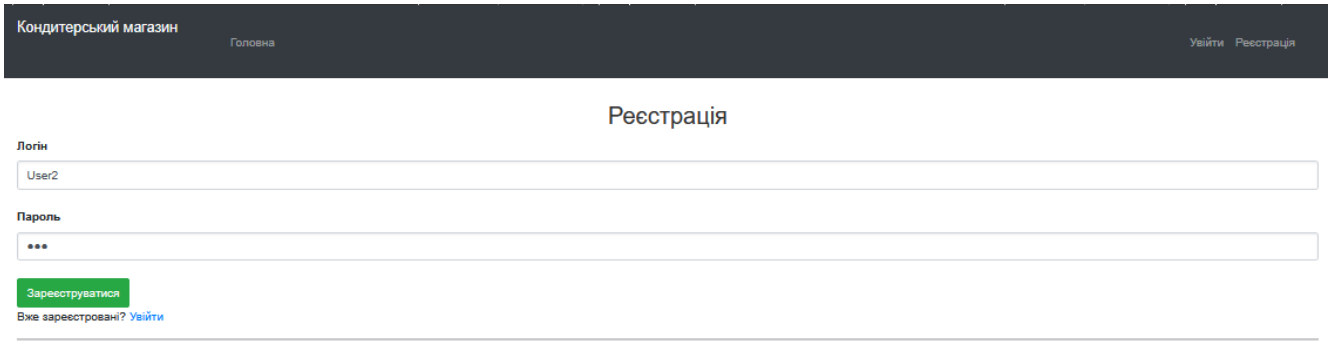


Рисунок Б.4 – Вікно реєстрації.

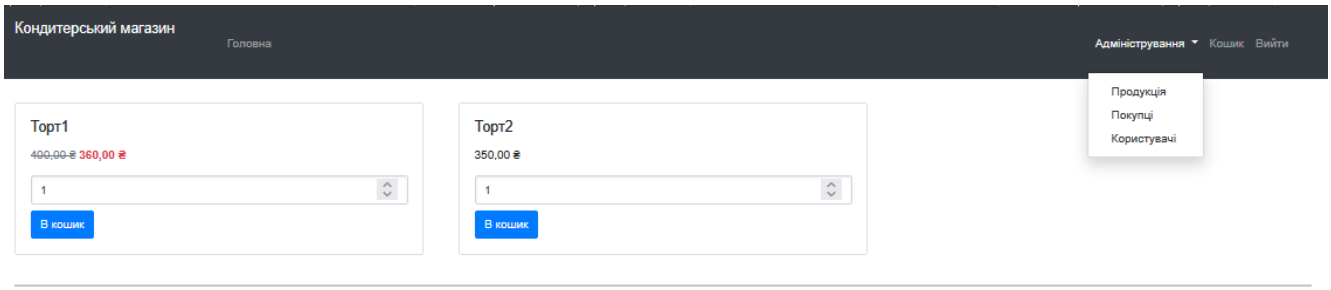


Рисунок Б.5 – Вікно головної сторінки у ролі адміністротора.

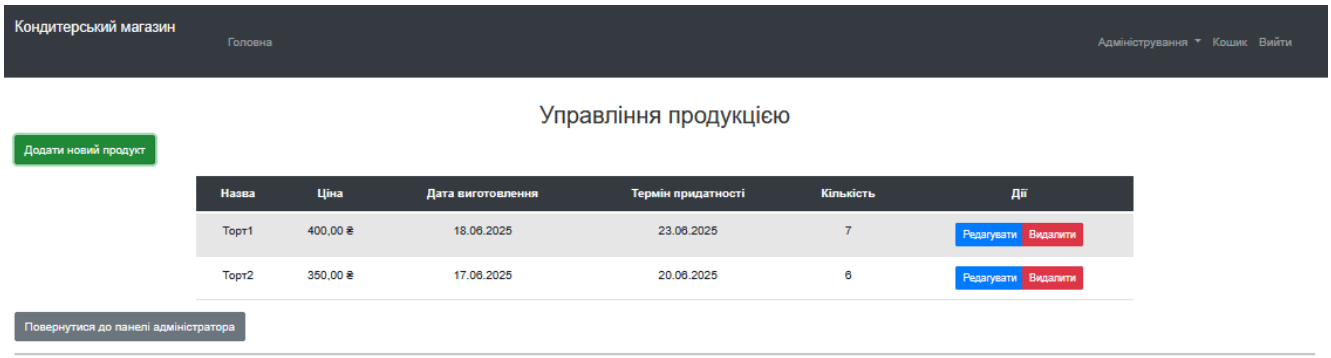


Рисунок Б.6 – Вікно управління продукцією у ролі адміністротора.

Продовження додатку Б

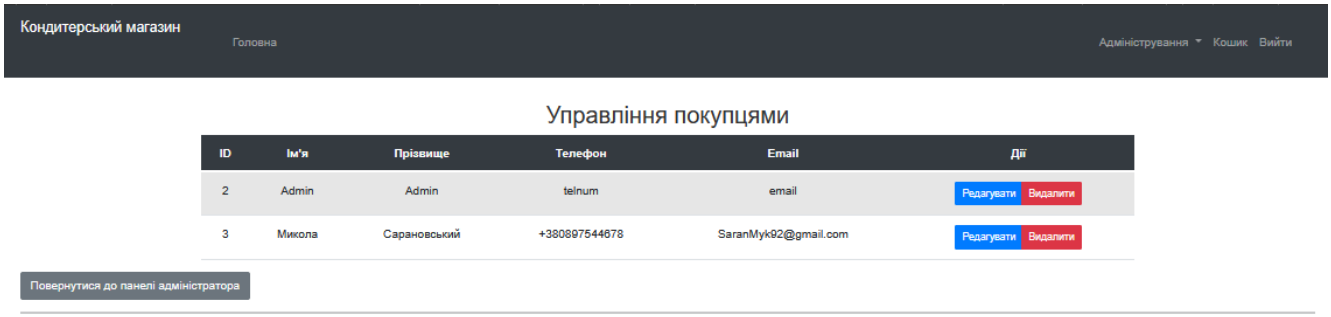


Рисунок Б.7 – Вікно управління покупцями у ролі адміністротора.

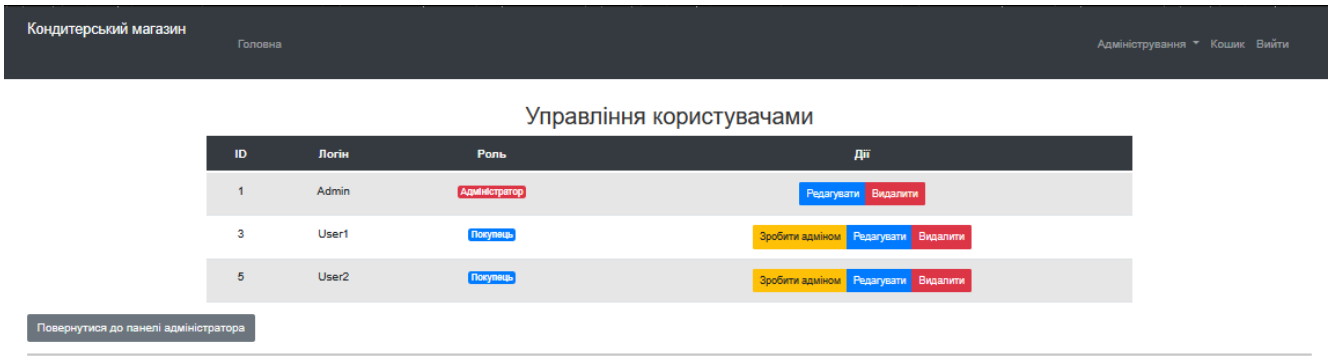


Рисунок Б.8 – Вікно управління користувачами у ролі адміністротора.