НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«До захисту допущено»
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
«___» _____ 2023 р.

Дипломний проєкт

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системне програмування та спеціалізовані комп'ютерні системи»

спеціальності 123 Комп'ютерна інженерія

на тему: «Інформаційна автоматизована система продажу речей»

Виконав (-ла):
студент (-ка) IV курсу, групи КВ-91
Бітлян Андрій Віталійович _____

Керівник:
Асистент СПСКС,
Радченко Костянтин Олександрович _____

Консультант з нормоконтролю:
Доц.каф. СПСКС, к.т.н.,
Клятченко Ярослав Михайлович _____

Рецензент:
Ст.викл. каф. обчислювальної техніки ФІОТ,
Алещенко Олексій Вадимович _____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2023 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет прикладної математики
Кафедра програмного забезпечення комп'ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Системне програмування та спеціалізовані комп'ютерні системи»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
«___»_____ 2023 р.

ЗАВДАННЯ

на дипломний проєкт студенту

Бітляну Андрію Віталійовичу

1. Тема проєкту «Інформаційна автоматизована система продажу речей», керівник роботи асистент Радченко Костянтин Олександрович, затверджені наказом по університету від «31» травня 2023 р. №2107-С

2. Термін подання студентом проєкту «___» червня 2023 р.

3. Вихідні дані до проєкту:
   - електронний ресурс, інформаційна автоматизована система продажу речей.

4. Зміст пояснювальної записки:
   - опис програмного забезпечення
   - опис програмного продукту;
   - тестування ресурсу.

5. Перелік обов'язкового ілюстративного матеріалу:
   - схема бази даних;
   - блок-схема методу Minus контролера CartController;
   - блок-схема додавання товару;
   - блок-схема додавання товару.

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Клятченко Я.М., доц. каф. СПСКС, к.т.н		

7. Дата видачі завдання «31» жовтня 2022 р.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1.	Вивчення літератури за тематикою проєкту	13.11.2022	
2.	Розроблення та узгодження технічного завдання	25.11.2022	
3.	Розроблення структури web-ресурсу	15.12.2022	
4.	Підготовка матеріалів першого розділу дипломного проєкту	30.12.2022	
5.	Розроблення дизайну сторінок та графічних елементів	03.02.2023	
6.	Підготовка матеріалів другого розділу дипломного проєкту	19.02.2023	
7.	Програмна реалізація web-ресурсу	10.03.2023	
8.	Тестування web-ресурсу	17.03.2023	
9.	Підготовка матеріалів третього розділу дипломного проєкту	30.03.2023	
10.	Підготовка матеріалів четвертого розділу дипломного проєкту	12.04.2023	
11.	Підготовка графічної частини дипломного проєкту	21.04.2023	
12.	Оформлення документації дипломного проєкту	26.05.2023	

Студент Андрій БІТЛЯН

Керівник роботи Костянтин РАДЧЕНКО

Анотація

Кваліфікаційна робота включає пояснювальну записку (60 с., 48 рис. 13 табл., 4 додатки).

Об’єкт розробки – інформаційна автоматизована система продажу речей, яка дозволяє вести моніторинг і управління товарами.

Система дозволяє: вести облік продукції (додавати/редагувати/видаляти товари); розміщувати та обробляти замовлення; створювати користувачів з різними рівнями привілеїв. В процесі розробки були використані технологія ORM Entity Framework, паттерн проєктування MVC (Модель-Вид-Контролер). В якості бази даних використовувалась MSSQL.

В ході розробки:

- проведено аналіз методів побудови існуючих web-застосунків для автоматизації збуту продукції;
- сформульовані вимоги до web-застосунку збуту продукції;
- розроблена структура web-застосунку для автоматизації збуту продукції магазину продукції;
- розроблено користувацький додаток для управління і моніторингу роботи web-застосунку для автоматизації збуту продукції магазину продукції;
- розроблено веб-сервіс для автоматизації обліку продукції, що виробляється, який передбачає механізми автоматизації збуту продукції;

Упровадження цього web-застосунку дозволить зменшити трудозатратність збуту продукції і збільшити конкурентоздатність підприємства.

Ключові слова:

WEB-ЗАСТОСУНОК ДЛЯ АВТОМАТИЗАЦІЇ ЗБУТУ ПРОДУКЦІЇ, ORM ENTITY FRAMEWORK, MVC, MSSQL.

ABSTRACT

The qualification work includes an explanatory note (60 pages, 48 figures, 13 tables, 4 appendices).

The object of the development is the creation of a web application for the sales automation, which allows monitoring and management of goods.

The web application allows: to keep track of products (add/edit/delete products); place and process orders; create users with different privilege levels. The ORM Entity Framework technology and the MVC (Model-View-Controller) design pattern were used in the development process. MSSQL was used as a database.

In the course of development:

- an analysis of the methods of building existing web applications for the automation of product sales was carried out;
- requirements for the product sales web application were formulated;
- a system of automation of sales management of store products was developed;
- the structure of the web application was developed to automate the sales of the products;
- a custom application was developed for managing and monitoring the operation of a web application for automating the sale of the store products;
- a web service was developed for automating the accounting of manufactured products, which provides mechanisms for automating the sale of products;

The implementation of this web application will reduce the labor cost and increase the competitiveness of the enterprise.

Keywords:

PRODUCT SALES AUTOMATION COMPUTER SYSTEM, ORM ENTITY FRAMEWORK, MVC, MSSQL.

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045492.001 ОА	Інформаційна	2		
			автоматизована система			
			продажу речей			
			Опис альбому			
	А4	ІАЛЦ. 467200.002 ТЗ	Інформаційна	4		
			автоматизована система			
			продажу речей			
			Технічне завдання			
	А4	ІАЛЦ.045492.003 ТП	Інформаційна	2		
			автоматизована система			
			продажу речей			
			Відомість технічного			
			проєкту			
	А4	ІАЛЦ. 045492.004 ПЗ	Інформаційна	60		
			автоматизована система			
			продажу речей			
			Пояснювальна записка			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.045440.001 ОА		
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей **Опис альбому**	Літ.	Арку 1 / Арку 2
Перевірив		Радченко К.О.					
Консульт.						КПІ ім. Ігоря Сі... ФПМ	
Н. контроль		Клятченко Я.М.					
Зав. каф.		Романкевич В.О.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045440.005 Д1	Інформаційна	1		
			автоматизована система			
			продажу речей			
			Схема бази даних			
	А4	ІАЛЦ.045440.006 Д2	Інформаційна	1		
			автоматизована система			
			продажу речей			
			Блок-схема методу Minus			
			контролера CartController			
	А4	ІАЛЦ.045440.007 Д3	Інформаційна	1		
			автоматизована система			
			продажу речей			
			Блок-схема додавання			
			товару			
	А4	ІАЛЦ.045440.008 Д4	Інформаційна	1		
			автоматизована система			
			продажу речей			
			Структурна схема взаємодії			

						А
Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.045440.001 ОА	2

ЗМІСТ

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Інформаційна автоматизована система продажу речей».

Галузь застосування: організація збуту продукції за допомогою мережі інтернет.

2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на дипломне проєктування на здобуття першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп'ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

3. МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою даного проєкту є створення інформаційної автоматизованої системи продажу речей.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації є технічна та науково-технічна література, технічна документація, публікації у періодичних виданнях та електронні статті у мережі Інтернет.

5. ТЕХНІЧНІ ВИМОГИ

5.1. Вимоги до програмного продукту, що розробляється

- сумісність з будь яким сучасним браузером (Google Chrome, Safari);
- можливість управління замовленнями;
- можливість додавати нові замовлення;

- можливість редагування замовлень;
- зберігання замовлень;
- сортування товарів за різноманітними параметрами;
- реєстрація користувачів;
- можливість керувати користувачами;
- вибір параметрів товару.

5.2. Вимоги до апаратного забезпечення

- Процесор: x86 or x64;
- Оперативна пам'ять: 512 МБ (мінімум), 1 ГБ (рекомендовано);
- Жорсткий диск: може знадобитися до 3 ГБ вільного місця;
- Наявність доступу до мережі Internet.

5.3. Вимоги до програмного та апаратного забезпечення користувача

- Наявність доступу до мережі Internet;
- Підтримка актуальної версії сучасних браузерів, наприклад Google Chrome *110.0. 5481.178*.

6. ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів
1.	Вивчення літератури за тематикою проєкту	15.04.2023
2.	Розроблення та узгодження технічного завдання	30.04.2023
3.	Аналіз існуючих рішень	05.05.2023
4.	Підготовка матеріалів першого розділу дипломного проєкту	10.05.2023
5.	Підготовка матеріалів другого розділу дипломного проєкту	18.05.2023
6.	Підготовка графічної частини дипломного проєкту	20.05.2023
7.	Оформлення документації дипломного проєкту	25.05.2023
8.	Попередній огляд матеріалів диплому на кафедрі	30.05.2023

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.045492.004 ПЗ	Інформаційна	60		
			автоматизована			
			продажу речей			
			Пояснювальна записка			
	А4	ІАЛЦ.045440.005 Д1	Інформаційна	1		
			автоматизована			
			продажу речей			
			Схема бази даних			
	А4	ІАЛЦ.045440.006 Д2	Інформаційна	1		
			автоматизована			
			продажу речей			
			Блок-схема методу			
			контролера			
	А4	ІАЛЦ.045440.007 Д3	Інформаційна	1		
			автоматизована			
			продажу речей			
			Блок-схема додавання			
			товару			

					ІАЛЦ.045440.003 ТП			
Змін.	Арк.	№ докум.	Підпис	Дата				
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей	Літ.	Ар	Аркушів
Перевірив		Радченко К.О.					1	2
Консульт.						КПІ ім. Ігоря Сікорського, ФПМ КВ-91		
Н. контроль		Клятченко Я.М.						
Зав. каф.		Романкевич В.О.						

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість	№ прим.	Примітки
	А4	ІАЛЦ.045440.008 Д4	Інформаційна	1		
			автоматизована			
			продажу речей			
			Структурна схема			

						Арк.
					ДП.045440.003 ТП	
Змін.	Арк.	№ докум.	Підпис	Дата		2

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

Entity Framework - структура об'єктно-реляційного відображення (ORM) з відкритим кодом для ADO.NET. Вона була частиною .NET Framework, але з версії 6 Entity Framework стала відокремленою від .NET Framework.

Міграції (Entity Framework Migration) — це інструмент, який автоматично оновлює схему бази даних під час будь-яких змін у моделі, але без втрати наявних даних або будь-яких об’єктів бази даних.

Model (EDM, Модель) — це набір концепцій, які описують структуру даних, незалежно від їх форми, що зберігається. EDM запозичено з моделі сутності-зв’язку, описаної Пітером Ченом у 1976 році, але також спирається на модель сутності-зв’язку та розширює її традиційне використання.

ViewModel - це клас, який містить поля, які представлені у строго типізованому поданні. Він використовується для передачі даних від контролера до строго типізованого представлення.

IDE – система програмних засобів, яка використовується програмістами для розробки програмного забезпечення (англ. Integrated Development Environment).

REST – підхід до архітектури мережевих протоколів, на основі які забезпечуються доступи до інформаційних ресурсів (англ. Representation State Transfer).

UI – засіб зручної взаємодії користувача з інформаційною системою (англ. User Interface).

UX — зручність, що людина відчуває при користуванні сервісом Основними об'єктами дослідження є враження, емоції та користь, отримані від взаємодії з продуктом. (англ. User Experience).

БД – база даних (англ. database).

СКУД – сукупність програмних і лінгвістичних засобів загального або спеціального призначення, для забезпечення управління створенням і використанням баз даних (англ. Database Management System, DBMS).

SQL – декларативна мова програмування для взаємодії користувача з базами даних, застосовується для формування запитів, оновлення і керування даними у реляційних БД, створення схеми БД і її модифікації, системи контролю за доступом до БД (англ. Structured query language).

MVC – архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення (англ. Model-view-controller).

HTTP – протокол передачі даних, що використовується в комп'ютерних мережах (англ. Hyper Text Transfer Protocol).

HTTPS – розширення протоколу HTTP з підтримкою шифрування з метою підвищення безпеки (англ. Hyper Text Transfer Protocol Secure).

UML – уніфікована мова моделювання (англ. Unified Modeling Language).

JSON — це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дає змогу описувати об'єкти та інші структури даних.

Bootstrap – веб фреймворк з набором інструментів для швидкого та зручного створення веб сайтів та сервісів

VS – Visual Studio

ВСТУП

У світі електронної комерції, де все більше людей шукають зручності і швидкість в покупках, розробка інформаційної автоматизованої системи продажу речей стає все більш актуальною та важливою задачею. Використання Інтернету для продажу товарів та послуг є зручним і ефективним способом привернення нових клієнтів та збільшення прибутку. Інформаційна автоматизована система продажу речей є однією з ключових складових електронної комерції, яка дозволяє підприємствам продавати свої товари в Інтернеті з будь-якої точки світу.

Актуальність даного дослідження полягає у вивченні наявних інформаційних автоматизованих систем продажу речей. І в поліпшенні їх роботи, шляхом підвищення ефективності вже існуючих рішень та створення нових варіантів вирішення задач.

У даному дипломному проєкті буде розглянуто інформаційна автоматизована система продажу речей. Метою проєкту є створення функціонального та зручного для користувача інтернет-магазину з використанням сучасних технологій програмування та дизайну. Проєкт буде включати в себе аналіз вимог користувачів, розробку архітектури та бази даних, вибір технологій програмування, розробку функціональності та дизайну, тестування та впровадження в проєкт.

В результаті виконання дипломної проєкту буде створена інформаційна автоматизована система продажу речей, яка буде забезпечувати зручний та безпечний процес покупки товарів в Інтернеті. Розроблений застосунок буде здатен забезпечити підприємству можливість продажу товарів в Інтернеті та збільшення прибутку завдяки широкому охопленню цільової аудиторії.

1. АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

1.1. Аналіз предметної області

Аналіз ринку сайтів з продажу товарів охоплює дослідження технологій та інструментів, які використовуються для створення та підтримки інтернет-магазинів.

У нинішній час велика кількість компаній, які займаються продажем товарів, від малого до великого розміру потребують використання інформаційних технологій аби бути рівними у конкуренції під час чи не всюдисущого та повсякденного застосування рішень з високим рівнем інформаційних технологій.

Однак специфіка засобів IT і методів їх впровадження, експлуатації і супроводу в залежності від масштабу підприємства може істотно відрізнятися. Якщо необхідний набір засобів IT в випадку з фірмою з чисельністю персоналу до 50 осіб може обмежуватися лише однією автоматизованою системою обліку і одним системним адміністратором середньої кваліфікації, то великої організації необхідний комплексний підхід до створення IT інфраструктури, який включає в себе створення IT відділу. Отже, для чого ж потрібен IT відділ? [13]

Починаючи відповідати на дане питання, варто перерахувати деякі нюанси роботи з інформаційними потоками на підприємстві і розглянути їх:

Однієї людини (а в деяких випадках - і десятьох) не вистачить для забезпечення необхідного обсягу робіт того чи іншого профілю, пов'язаних з IT [13];

Жодна людина не може бути висококваліфікованим фахівцем відразу у всіх областях IT. Отже, в штаті IT співробітників компанії повинні складатися фахівці різного профілю [13];

Кожен фахівець зайнятий виконанням тільки свого завдання, поставлених перед IT інфраструктурою підприємства в цілому [13];

Як випливає з вищевикладеного, робота декількох людей, що виконують різну роботу повинна бути скоординована. Що передбачає залучення до роботи

керуючого по ІТ - ІТ директора, здатного розуміти як діяльність кожного з фахівців, так і цілі і завдання ІТ інфраструктури в рамках діяльності підприємства [13].

Розгляд даних нюансів неминуче наштовхує на висновок про те, що відділ ІТ повинен бути системою, а не просто набором засобів і компанією дружно працюючих фахівців з різних областей. Таким чином, відділ ІТ компанії - це сукупність взаємодіючих засобів ІТ і фахівців в області ІТ, цілями якої є: [13]

а) забезпечення інформаційними технологіями;

б) підвищення ефективності діяльності компанії за допомогою оптимізації інформаційних потоків.

ІТ відділ компанії виконує завдання, такі, як: [13]

а) реалізація ІТ проектів;

б) забезпечення працездатності інформаційних систем;

в) надання керівній ланці компанії відомостей про нові можливості ІТ і технологіях управління ними;

г) діловодство відділу, ведення бюджету ІТ, облік ІТ активів, забезпечення кадрового складу ІТ.

Що стосується кадрового складу відділу ІТ, то в більшості випадків він включає в себе співробітників наступних профілів:

а) спеціаліст з мережевого забезпечення;

б) системний аналітик; в) програміст;

г) системний адміністратор;

д) спеціаліст з підтримки користувачів;

е) керівник відділу інформаційних технологій;

є) веб-майстер [13].

Спеціаліст з мережевого забезпечення визначає чи виникають в ході роботи мережі проблеми; аналізує вимоги користувачів; координує

процес налагодження та підтримки мережевого обладнання; забезпечує сумісність програмного і апаратного мережевого забезпечення; готує бюджет в підзвітною сфері та забезпечує ефективне використання ресурсів; керує менш кваліфікованим технічним персоналом [13].

Системний аналітик проводить аналіз вимог користувачів для визначення конфігурації програмного та апаратного забезпечення; готує технічні специфікації, технічні звіти щодо підтримки програмного і апаратного забезпечення; координує процес випробувань та введення в експлуатацію ІТ забезпечення; проводить аналіз комплексних питань програмування щодо модифікації коду вже існуючих програм і створення коду для нових програм [14].

Програміст вирішує комплексні питання програмування, що стосуються модернізації, модифікації вже існуючого коду або створення нового коду; готує графіки та діаграми робочого процесу; встановлює послідовність проведення операцій по введенню та комп'ютерної обробки даних; контролює проведення тестування і налагодження програмного забезпечення [13].

Системний адміністратор виробляє установку програмного і апаратного забезпечення; здійснює моніторинг та оптимізацію роботи операційних систем обчислювальної техніки; визначає проблеми в програмному забезпеченні; аналізує вимоги користувача, оцінює додаткові можливості щодо поліпшення роботи програмного забезпечення [14].

Спеціаліст з підтримки користувача відповідає за установку і проведення діагностики програмного і апаратного забезпечення; надає технічну підтримку і консультації кінцевим користувачам; несе відповідальність за організацію ремонту комп'ютерної техніки; забезпечує наявність витратних матеріалів для комп'ютерної та оргтехніки; консультує користувачів з технічних питань [13].

Керівник відділу IT керує будь-якою діяльністю, пов'язаною з обслуговуванням обчислювальної техніки; контролює процес підбору, установки, підтримки програмного і апаратного забезпечення [13].

Який, на вашу думку, найбільш цінний навик хорошого фахівця?

Що за універсальне уміння, без якого рядовий співробітник ніколи не стане топ-менеджером, а невеликий стартап не перетвориться в багатомільярдну корпорацію? Правильно, комунікабельність або, простими словами, уміння спілкуватися з людьми. А тепер уявіть, що ви власник досить великої компанії, скажімо зі штатом 100 чоловік. Ви сидите в своєму кабінеті в шкіряному кріслі, за широким дубовим столом, п'єте дивовижну мелену каву. Начебто все добре, але ви точно знаєте, що у вашій компанії є розбіжності між рівнем освідомлення продавців, ви не впевнені чи всі адміністратори інформовані щодо статусу товарів і чи всі продавці мають доступ до останніх змін клієнтської бази. Саме для вирішення цієї проблеми буде створений даний сервіс.

- Інтеграція з іншими технологіями Microsoft: ASP.NET підтримує безпроблемну інтеграцію з іншими продуктами та технологіями від Microsoft, такими як SQL Server для управління базами даних, Azure для хмарного розгортання та інші.

Загалом, ASP.NET є потужним фреймворком, який забезпечує широкі можливості для роз створення інформаційної автоматизованої системи продажу речей. Він пропонує масштабованість, безпеку, швидкість, можливості розширення, кросплатформеність та інтеграцію з іншими технологіями, що робить його привабливим вибором для розробників. Завдяки цим перевагам, ви зможете створити потужну та ефективну систему продажу речей, яка буде гнучко реагувати на зміни та забезпечувати безпеку вашої бізнес-діяльності [15].

1.2. Постановка задачі

Метою даного сервісу є автоматизація управління інформаційною автоматизованою системою продажу речей, підвищення продуктивності працівників.

Головне призначення сервісу – облегшення обліку товарів та користувачів. Автоматизація процесу замовлення товарів за допомогою інтерактивного веб застосунку, який буде доступний з будь-якого пристрою.

Таким чином, при розробці web-застосунку для продажу речей потрібно втілити наступні пункти:

- Типи товарів. Потрібно розглянути різноманітні типи товарів, які можуть продаватись на веб-сайті, такі як електроніка, одяг, книги, іграшки, косметика тощо. При цьому слід врахувати особливості кожного типу товарів та їх специфіку в продажу.
- Функції веб-сайту. Потрібно визначити, які функції будуть доступні на веб-сайті. Це можуть бути функції, такі як пошук товарів, додавання товарів у кошик, оформлення замовлення, сповіщення про знижки та акції, оплата, доставка тощо.
- Безпека: Потрібно забезпечити належний рівень безпеки для користувачів та продавців на платформі. Це можуть бути заходи забезпечення конфіденційності даних, захисту від шахрайства, захисту від кібератак тощо.
- Дизайн: Потрібно розробити дизайн web-застосунку, який буде привабливим та зручним для користувачів. Важливо визначити, які будуть кольорова гама, шрифти, компоненти і елементи дизайну, які будуть використовуватися. Дизайн повинен бути також адаптивним до різних розмірів екранів, щоб користувачі могли зручно використовувати платформу на будь-яких пристроях.
- Кабінет користувача. Створити розділ, в якому користувач зможе управляти своїм аккаунтом. Змінювати дані, наприклад, номер телефону, пароль чи електронну пошту. Додати можливість видалення профілю.

- Керування замовленнями: Потрібно розробити панель, у якій будуть відображатись замовлення. Додати можливість фільтрувати, сортувати і управляти замовленнями.

1.3. Аналіз ринку та існуючих способів рішення задачі

Подібні систему вже існують багато часу, тому можна навести безліч прикладів. Проте, у кожному окремому випадку функціонал і дизайн розробляються відповідно до специфічних вимог підприємства. У даному випадку тема дипломного проєкту передбачає розробку системи з продажу речей, тому розглянемо вже наявні подібні системи.

ASOS - мультибрендовий британський магазин, широко відомий в Європі і взагалі в світі.[12]

Сайт цієї системи можна побачити на рисунку 1.1.

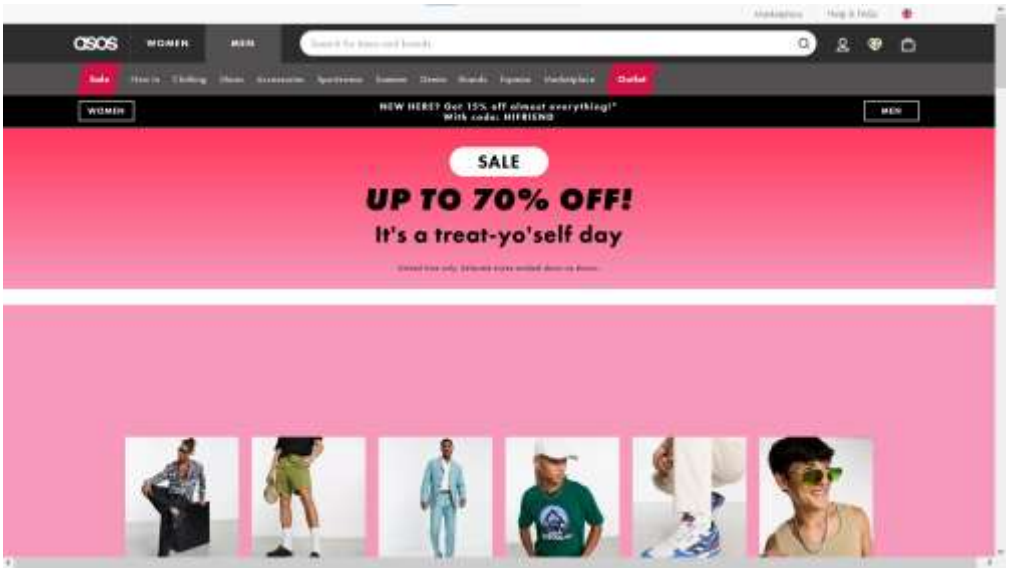


Рисунок 1.1 – Сайт https://www.asos.com

Ресурс надає наступні можливості:

Пошук та перегляд товарів: На сайті можна шукати товари за категоріями, такими як одяг, взуття, аксесуари, косметика та інші. Крім того, є можливість використовувати фільтри для звуження пошуку за розміром, кольором, ціною та іншими параметрами.

Детальна інформація про товар: Кожен товар має власну сторінку з докладним описом, фотографіями, ціною, розмірами, доступними кольорами та іншими характеристиками. Крім того, зазвичай надаються відгуки клієнтів, які допомагають узнати більше про якість та підходять чи не підходять розміри.

Кошик покупок: Користувачі можуть додавати товари до свого кошика покупок, переглядати зміст кошика, змінювати кількість товарів та видаляти їх. Також є можливість зберегти товари в кошику для покупки пізніше.

Замовлення та оплата: Користувачі можуть оформити своє замовлення, вказавши необхідну інформацію про доставку та спосіб оплати. На сайті підтримуються різні методи оплати, такі як кредитні картки, PayPal та інші електронні платіжні системи.

Особистий кабінет: Користувачі можуть створити особистий обліковий запис, де вони зможуть зберігати свої дані, адреси доставки, відстежувати замовлення, переглядати історію покупок та використовувати інші функції.

Сайт leboutique.com є онлайн-платформою, яка пропонує широкий асортимент товарів для покупки онлайн.

Сайт цієї системи можна побачити на рисунку 1.2.

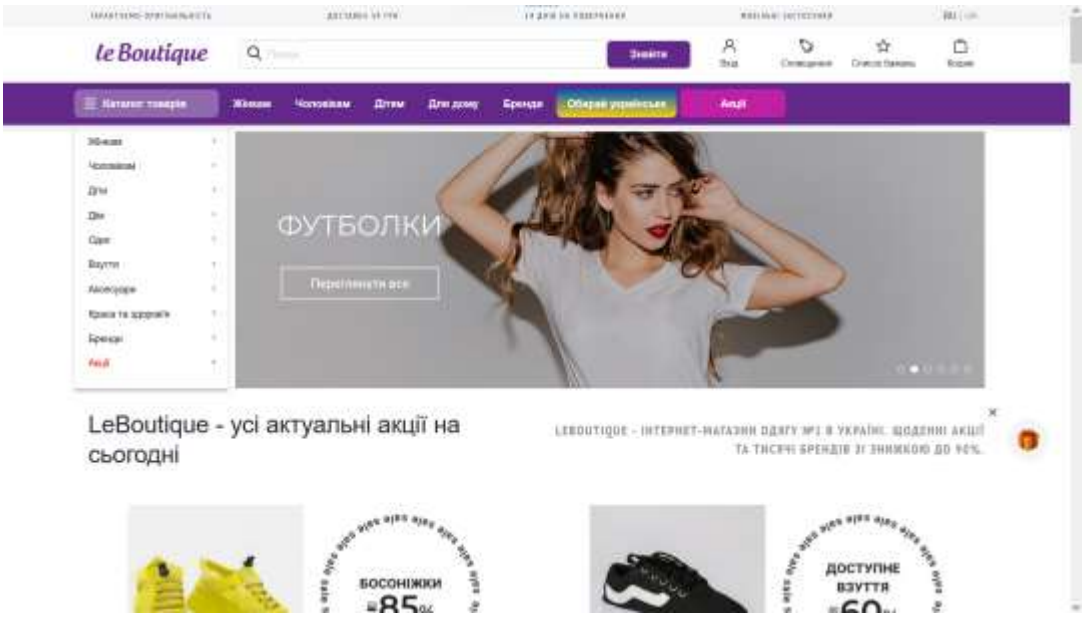


Рисунок 1.2 – Сайт https://leboutique.com/uk

Ресурс надає наступні можливості:

Пошук товарів: Користувачі можуть шукати товари за різними параметрами, такими як назва, категорія, бренд або ключові слова. Пошуковий двигун допомагає знайти потрібний товар швидко та ефективно.

Каталог товарів: Сайт має добре організований каталог, де товари розподілені за категоріями, що полегшує навігацію та пошук. Кожна категорія містить підкатегорії та фільтри, що дозволяють користувачам звузити свій пошук.

Детальна інформація про товар: Кожен товар має свою сторінку з докладним описом, фотографіями, характеристиками, ціною та доступністю. Користувачі можуть ознайомитися з усією інформацією перед придбанням товару.

Кошик покупок: Користувачі можуть додавати товари до свого кошика покупок, який зберігається протягом сесії. У кошику можна змінювати кількість товарів, видаляти їх або зберігати для подальшої покупки.

Оформлення замовлення: Після вибору товарів у кошику користувачі можуть оформити замовлення, вказавши необхідні дані для доставки та спосіб оплати. Сайт забезпечує безпечну обробку платежів та захист персональних даних користувачів.

Особистий кабінет: Зареєстровані користувачі можуть створити особистий кабінет, де вони зможуть переглядати історію замовлень, відстежувати статус доставки та зберігати адреси доставки для зручності.

Порівняння аналогів можна побачити на таблиці 1.1.

Таблиця 1.1 – Аналіз-порівняння аналогів

Назва сервісу	Пошук товарів	Особистий кабінет	Блокування користувачів	Кошик покупок	Видалення аккаунту
ASOS	+	+	-	+	-
Leboutique	+	+	-	+	-
Shopper	+	+	+	+	+

Shopper – інформаційна автоматизована система продажу речей.

2. МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1. Теоретичні відомості

2.1.1. Патерни проєктування

При розробці програмного продукту розробники завжди розробляють архітектуру додатку, звертаючи увагу на шаблони проєктування в першу чергу. Під час розробки програми найкраще використовувати лише один патерн структури програмного забезпечення задля уникнення конфліктів.

Отже, розглянемо відмінності наступних патернів Model-View-ViewModel (MVVM), Model-View-Presenter (MVP) і Model-View-Controller (MVC).

Для чого потрібні патерни Model-View-(C або P або VM)?

Ціль цих архітектур полягає в тому, щоб розділити відповідальність за візуалізацію, обробку та керування даними для додатків інтерфейсу користувача [3].

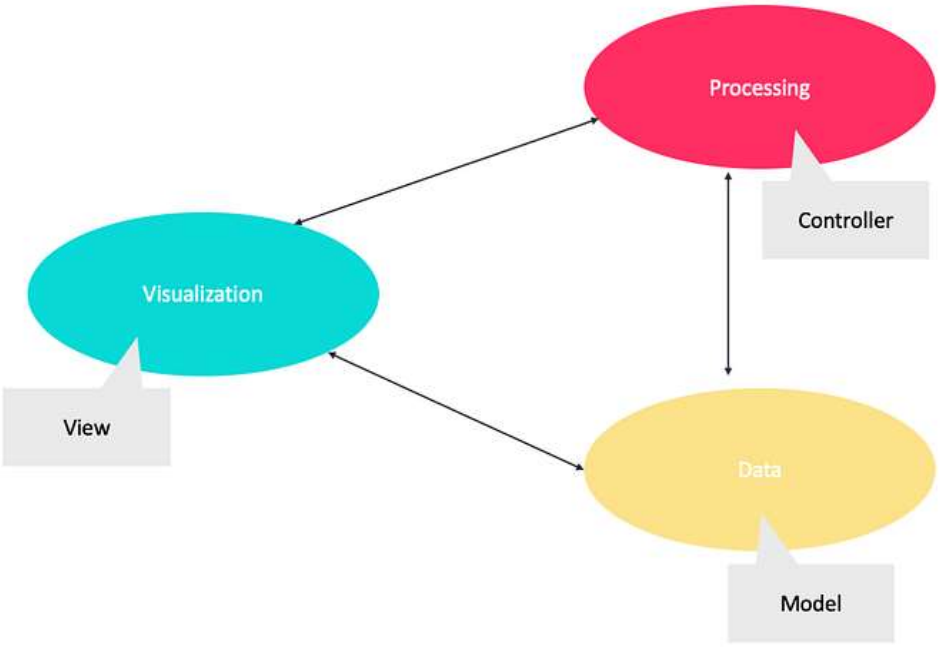


Рисунок 2.1 – розділення відповідальності за візуалізацію, обробку та керування даними

Model-View-Controller

Або, скорочено, MVC — це широко використовуваний шаблон проєктування для розробки програмних програм. Патерн спочатку був розроблений Трюгве Реенскаугом під час його роботи над Smalltalk-80 (1979), де він спочатку називався Model-View-Controller-Editor. Далі MVC був детально описаний у «Патернах проєктування: Елементи багаторазового використання об'єктно-орієнтованого програмного забезпечення» (книга «GoF») у 1994 році, що відіграло роль у популяризації його використання. Шаблон розбиває програму на три компоненти[3].

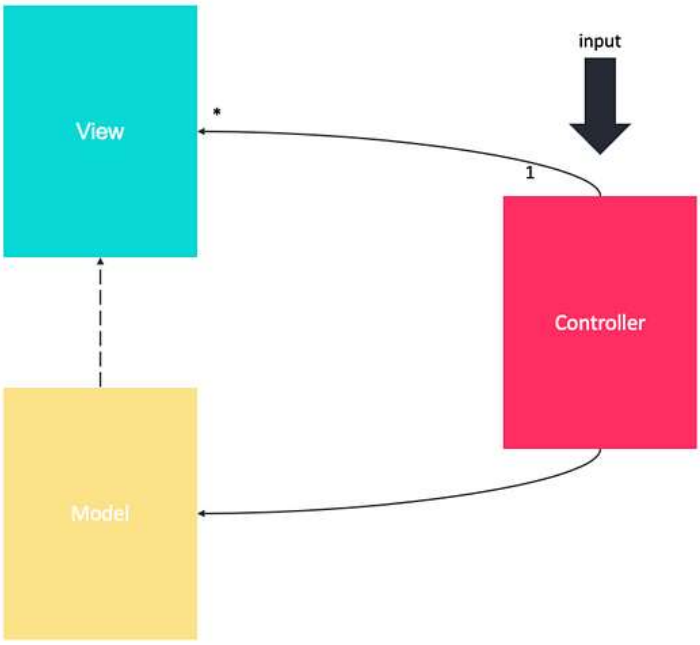


Рисунок 2.2 - Структура патерна MVC

Model — відповідає за бізнес-логіку програми та керує станом програми. Що також включає читання та запис даних, збереження стану додатка, і може навіть включати завдання, пов'язані з керуванням даними, як-от робота в мережі та перевірка даних [3].

View — цей компонент виконує два важливі завдання: представлення даних користувачеві та керування взаємодією користувача [3].

Controller — шар перегляду та шар моделі склеєні разом одним або кількома контролерами [3].

Model–View–Presenter

MVP є похідною від шаблону проєктування MVC, який фокусується на покращенні логіки презентації. Він виник у компанії під назвою Taligent на початку 1990-х років, коли вони працювали над моделлю середовища C++ CommonPoint [3].

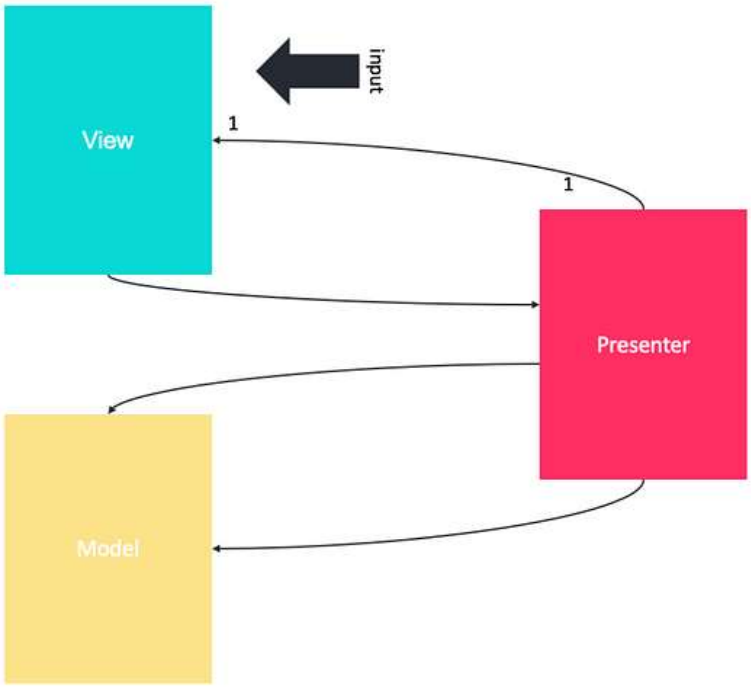


Рисунок 2.3 - Структура патерна MVP

Model — модель представляє набір класів, які описують бізнес-логіку та дані. Також модель визначає бізнес-правила для даних, тобто те, як дані можна змінювати та маніпулювати ними [3].

View — представлення використовується для взаємодії з користувачами, як-от XML, Activity, fragments. Це не має нічого спільного з логікою, яка має бути реалізована в процесі [3].

Presenter — презентатор отримує вхідні дані від View, обробляє дані за допомогою моделі та передає результати назад у View після завершення обробки [3].

Model-View-ViewModel

MVVM спочатку був визначений Microsoft для використання з Windows Presentation Foundation (WPF) і Silverlight, про що був офіційно оголошений у 2005 році Джоном Гроссманом у дописі в блозі про Avalon (кодова назва для WPF). Цей шаблон заснований на MVC і MVP, який намагається чіткіше відокремити розробку інтерфейсу користувача від бізнес-логіки та поведінки в програмі.

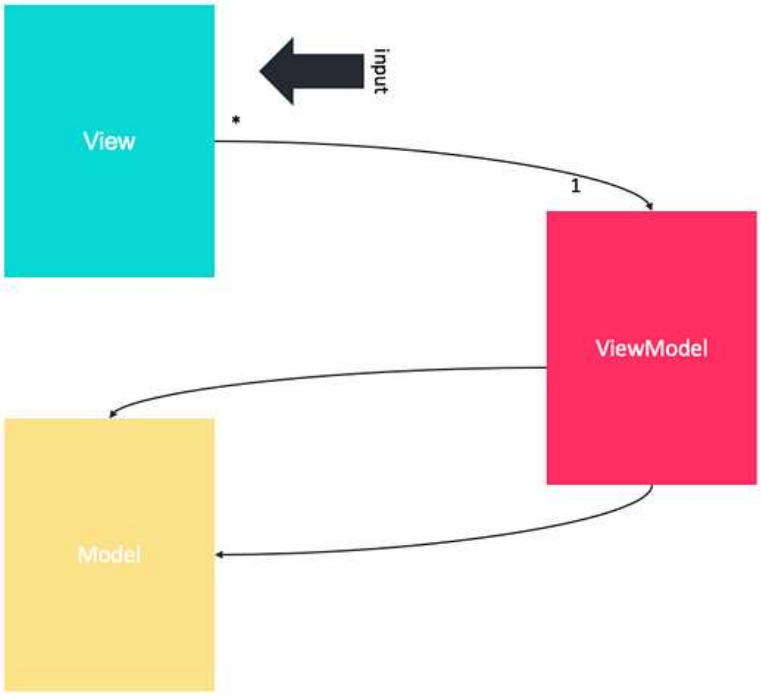


Рисунок 2.4 - Структура патерна MVVM

Model — модель, яка використовується в MVVM, подібна до моделі, яка використовується в MVC, і складається з основних даних, необхідних для запуску програмного забезпечення [3].

View — це графічний інтерфейс між користувачем і шаблоном проєктування, подібний до того, що є в MVC. Він відображає вихід оброблених даних[3].

View-Model — View-Model, з одного боку, є абстракцією View, а з іншого — забезпечує оболонку даних моделі, які потрібно зв’язати. Тобто він містить модель, яка перетворюється на представлення, а також містить команди, які представлення може використовувати для впливу на модель [3].

Коли використовувати MVC

MVC — це реалізація поділу інтересів. Якщо в програмі потрібно розділити дані (модель), обробку даних (контролер) і представлення даних (представлення), то використання патерну MVC є цілком доречним. MVC також підходить, коли джерело даних і/або представлення даних можуть змінюватися в будь-який час [4].

Коли використовувати MVP

MVP доречно використовувати, якщо програма має двонаправлений потік. Якщо під час взаємодії користувачу потрібно запросити щось у моделі, і результат цього запиту змінить інтерфейс користувача, то варто розглядати MVP [4].

Коли використовувати MVVM

MVVM доречно використовувати, якщо треба поділитися проєктом з дизайнером, при цьому робота з проєктування та розробки може відбуватися незалежно. Також цей патерн актуальний у випадку, коли необхідне модульне тестування ваших рішень. Якщо потрібно мати багаторазові компоненти як усередині, так і між проєктами вашої організації, то MVVM підійде [4].

В результаті, був обраний патерн MVC для реалізації структури програмного забезпечення. Оскільки він простий в реалізації та найкраще підходить для даного проєкту. Застосування цього патерну дозволить легко побудувати структуру програми з розділеними інтересами. До того ж в Visual Studio присутній вбудований шаблон проєкту ASP.NET MVC.

Також до переваг патерну MVC можна віднести те, що він є найбільш розповсюдженим. Таким чином, підтримувати програмне забезпечення буде простіше, адже знайти розробника, який знайомий з цією технологію, легше.

2.2. Опис використаних технологій при розробці

2.2.1. Огляд технологій

Razor Pages

Серверна платформа ASP.NET Razor Pages, орієнтована на сторінку, яка вперше була представлена як компонент ASP.NET Core, а тепер є частиною .NET 5, дозволяє розробляти динамічні веб-сайти, керовані даними, із чітким розподілом обов'язків. . Razor Pages, компонент інфраструктури веб-розробки Microsoft ASP.NET Core, пропонує кросплатформну розробку та може бути встановлений на комп'ютерах Windows, Unix і Mac.

Архітектура Razor Pages є компактною та неймовірно адаптивною. Це дає розробнику повний контроль над HTML, який відображається. Запропонованою структурою для кросплатформного створення HTML на стороні сервера є Razor Pages.

Для створення динамічного вмісту для браузерів Razor Pages використовує популярну мову програмування C# для розробки на стороні сервера та простий у вивченні синтаксис шаблонів Razor.

Razor Pages сприяє розподілу відповідальності та є архітектурною реалізацією парадигми MVC [5].

Дана технологія значно спрощує процес розробки програмного забезпечення, оскільки дає можливість писати код C# у файлі HTML. Адже, якщо потрібно додати якусь невелику дію, то не потрібно це робити в окремому файлі на сервері.

Таким чином, Razor Pages є, певною мірою, аналогом мови Java Script, яка надає схожі можливості розробнику програмного забезпечення.

2.2.2. Огляд утиліт для розробки

Visual Studio

Microsoft Visual Studio — серія продуктів фірми Майкрософт, які містять інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів. Ці продукти дають змогу розробляти як консольні програми, так і програми з графічним інтерфейсом, включно з підтримкою технології Windows Forms, а також вебсайти, вебзастосунки, вебслужби як у рідному, так і в керованому кодах для всіх платформ, що підтримуються Microsoft Windows, Windows Mobile, Windows Phone, Windows CE, .NET Framework, .NET Compact Framework та Microsoft Silverlight [6].

Переваги програми:

- Велика кількість розширень, наприклад *beautify*;
- Автоматичний імпорт файлів;
- Доступні практично на всі мови;
- Сумісність з Git;
- Простота налаштування.

Недоліки програми:

- Займає багато місця на диску;
- Швидкодія деколи не достатня;
- Технічна підтримка.

2.3. Налаштування середовища розробки

2.3.1. Налаштування середовища для розробки сервісу

Спочатку встановлюємо середовище для розробки Visual Studio Community 2022.

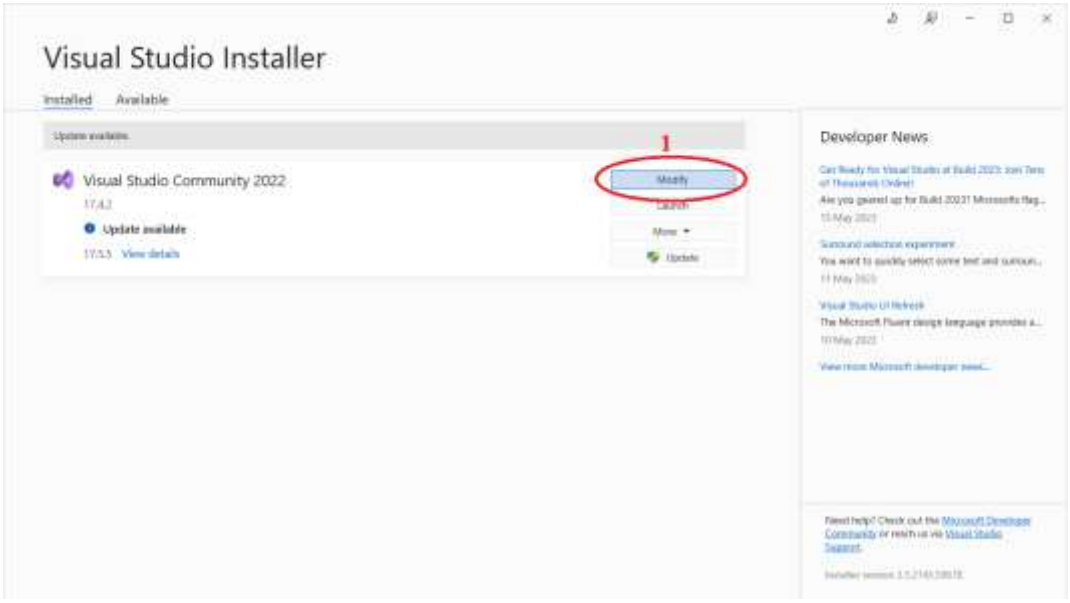


Рисунок 2.5 – Встановлена Visual Studio Community 2022

Після чого заходимо в Visual Studio Installer, і у вкладці *Installed* бачимо, що середовище розробки встановлено успішно, як видно на рисунку 2.5.

Наступним кроком натискаємо на кнопку *Modify*, яка обведена на рисунку 2.5 під номером 1.

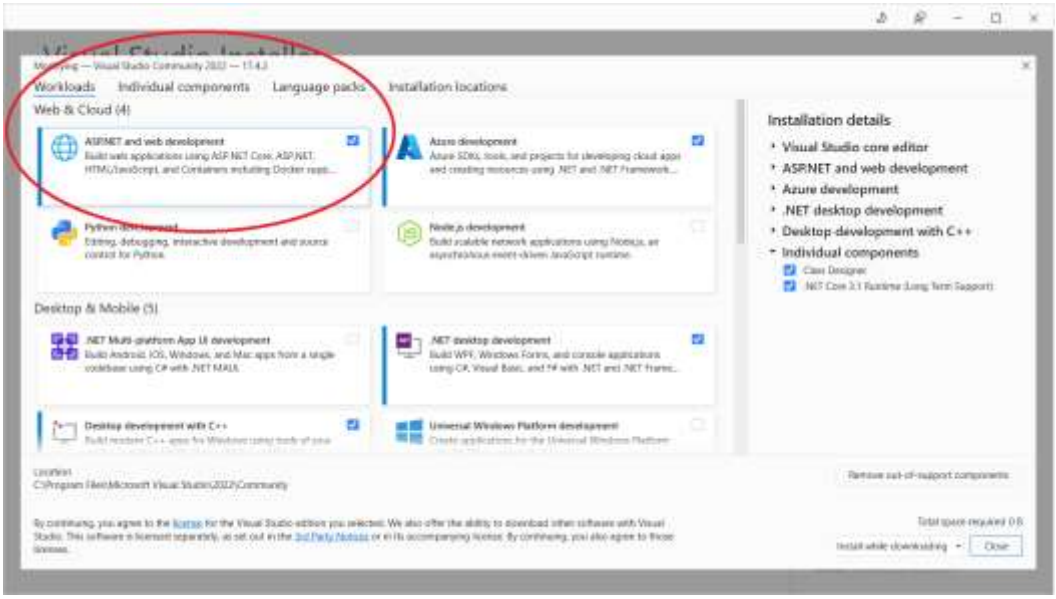


Рисунок 2.6 – Встановлення компонентів для VS 2022

У першій вкладці *Workloads* знаходимо розділ *Web & Cloud*, і встановлюємо компонент *ASP.NET and web development*. Цей компонент обведений на рисунку 2.6.

Для створення проєкту необхідно запустити Visual Studio 2022.

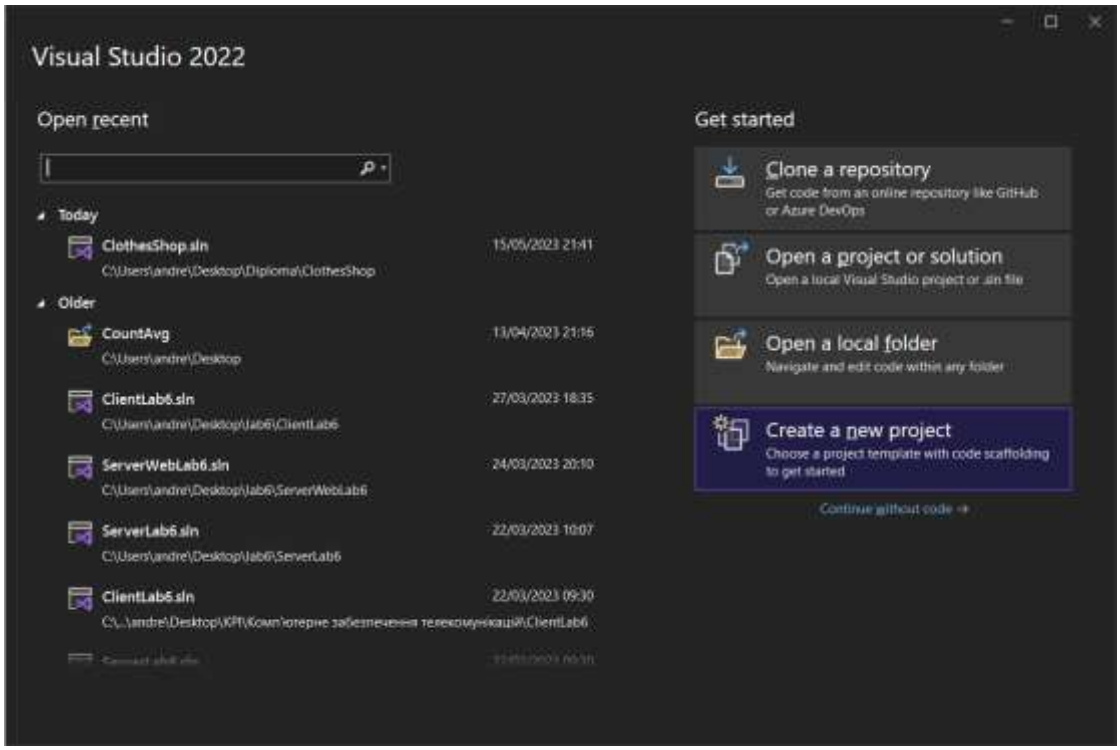


Рисунок 2.7 – Початкова сторінка VS 2022

Обираємо опцію *Create a new project*, як зазначено на рисунку 2.7.

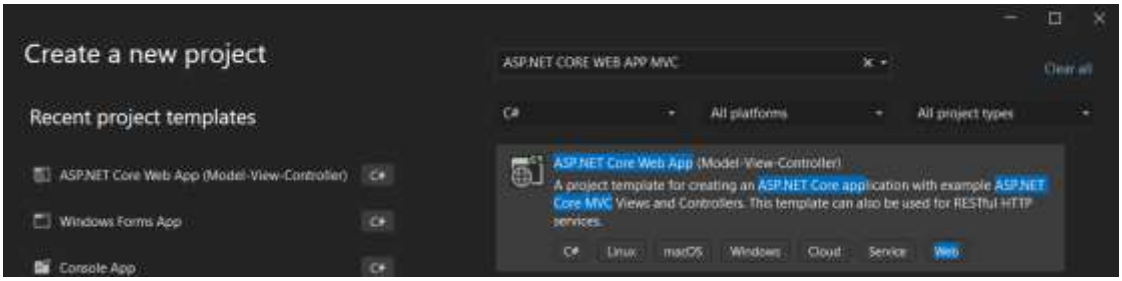


Рисунок 2.8 – Вибір типу проєкту

В наступному меню пишемо потрібно знайти тип проєкту *ASP.NET CORE WEB APP MVC*, і обрати його, як на рисунку 2.8.

Згодом перейти далі, та ввести назву проєкту і його розташування.

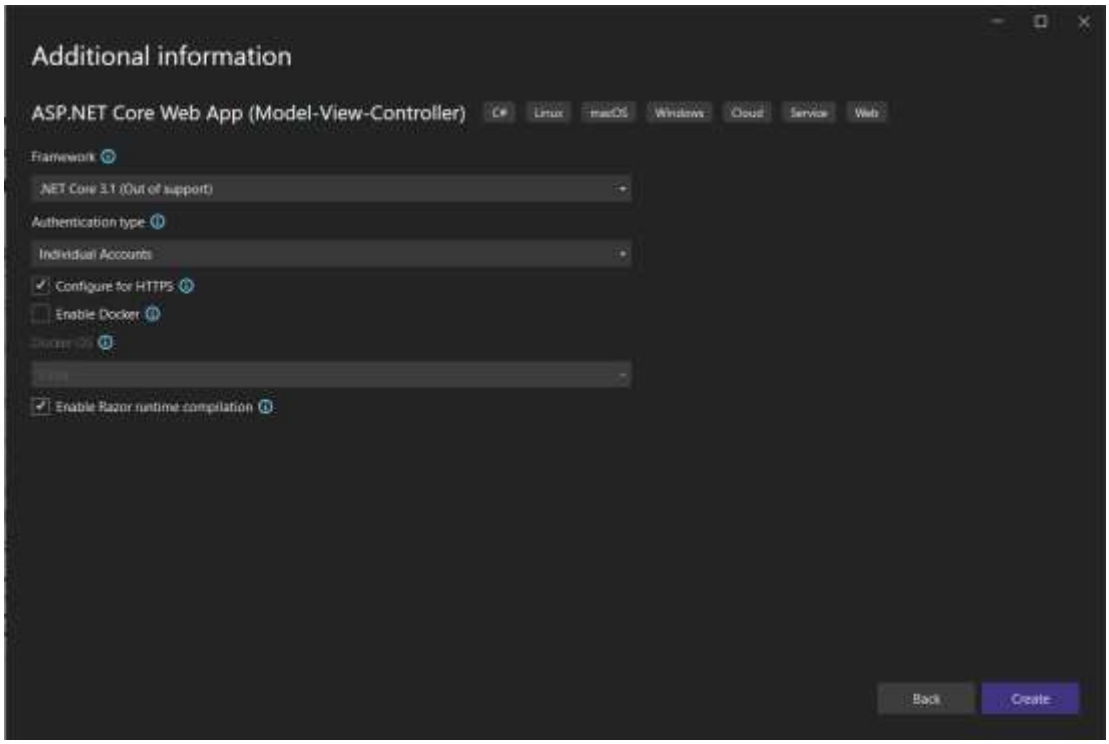


Рисунок 2.9 – Додаткова інформація для створення проєкту

Останнім кроком обираємо налаштування такі, як зазнчені на рисунку 2.9 – і натискаємо кнопку Create, тим самим створюючи проєкт.

2.4. Складання та опис алгоритму

Під час розробки програму було розділено на один основний проєкт і 3 проєкти бібліотеки для розділення відповідальності відповідно до патерну проєктування MVC.

Таким чином маємо 4 проєкти:

- Основний проєкт *ClothesShop*

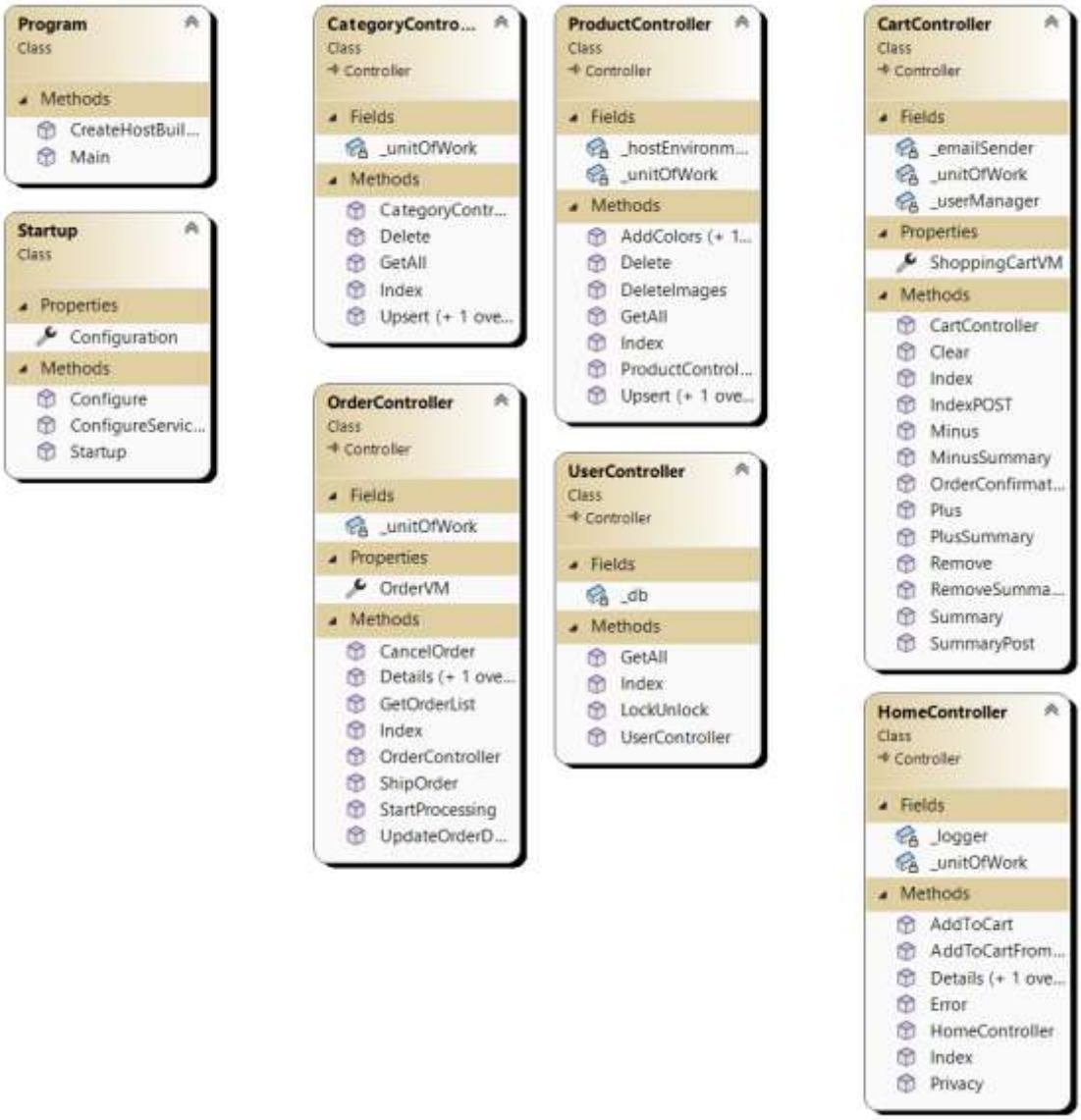


Рисунок 2.10 - Діаграма класів проєкту *ClothesShop*

- Бібліотека *ClothesShop.DataAccess*

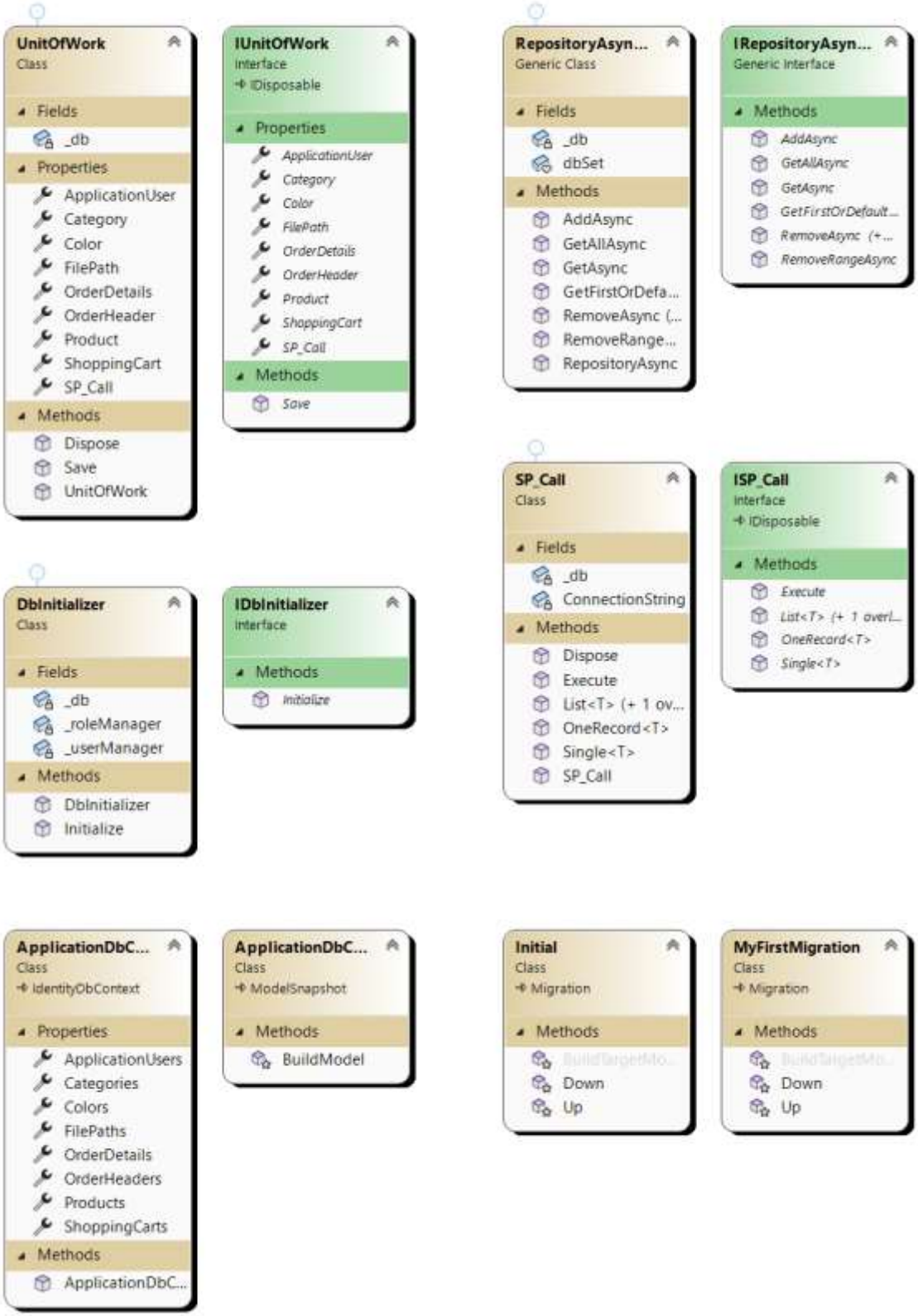


Рисунок 2.11 – Перша частина діаграми класів бібліотеки *ClothesShop.DataAccess*

Рисунок 2.12 – Друга частина діаграми класів бібліотеки *ClothesShop.DataAccess*

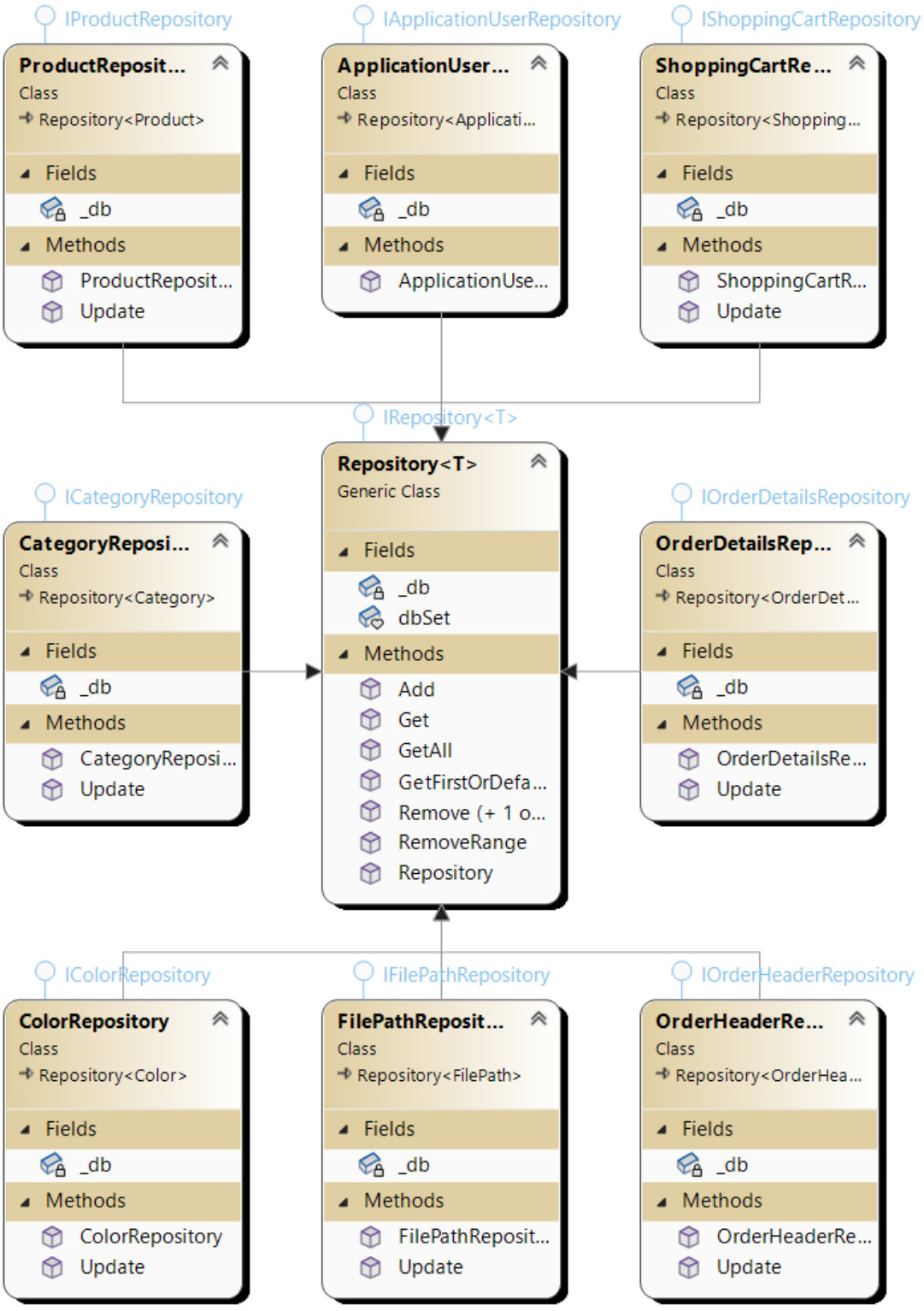


Рисунок 2.13 – Третя частина діаграми класів бібліотеки *ClothesShop.DataAccess*

- Бібліотека *ClothesShop.Models*

Рисунок 2.14 – Діаграма класів бібліотеки *ClothesShop.Models*

- Бібліотека *ClothesShop.Models*

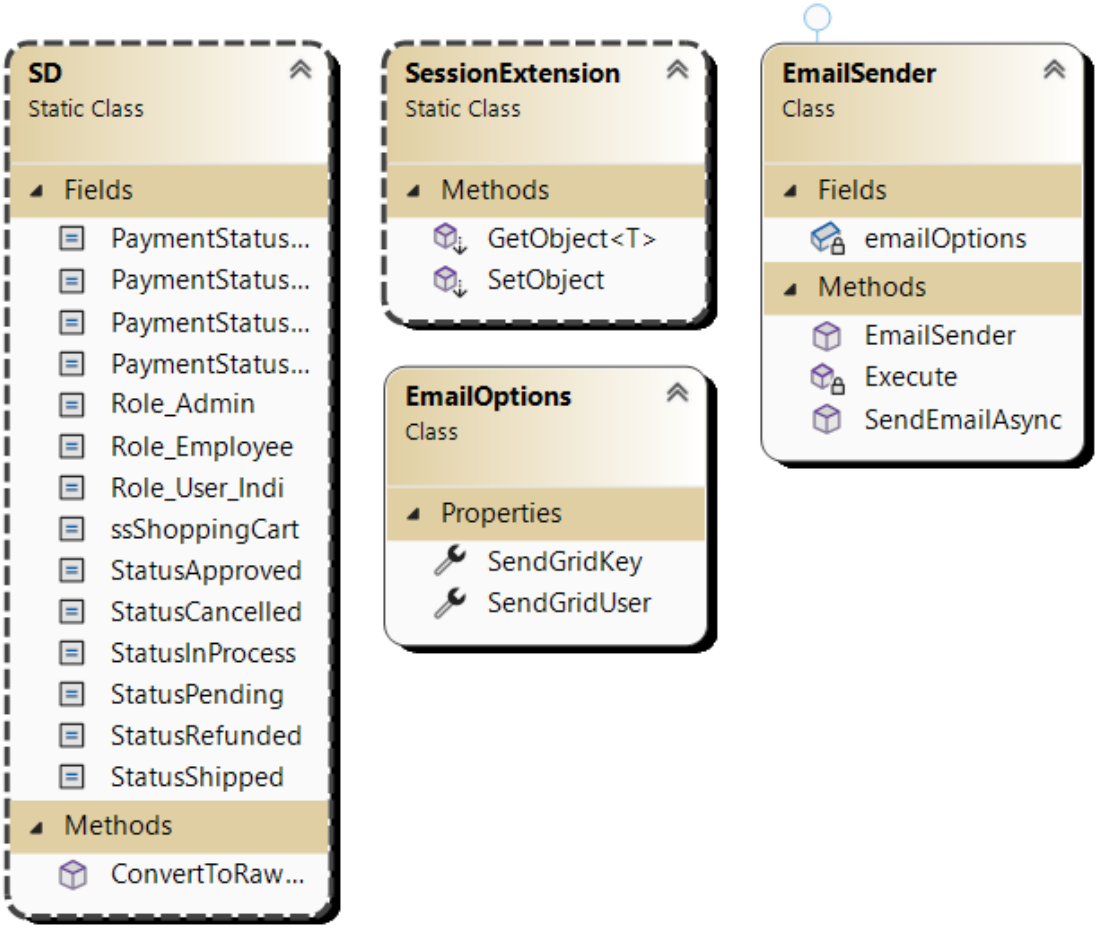


Рисунок 2.14 – Діаграма класів бібліотеки *ClothesShop.Utility*

У даній програмі структура побудови – MVC. Схема роботи застосунку зазначена на рисунку 2.15.

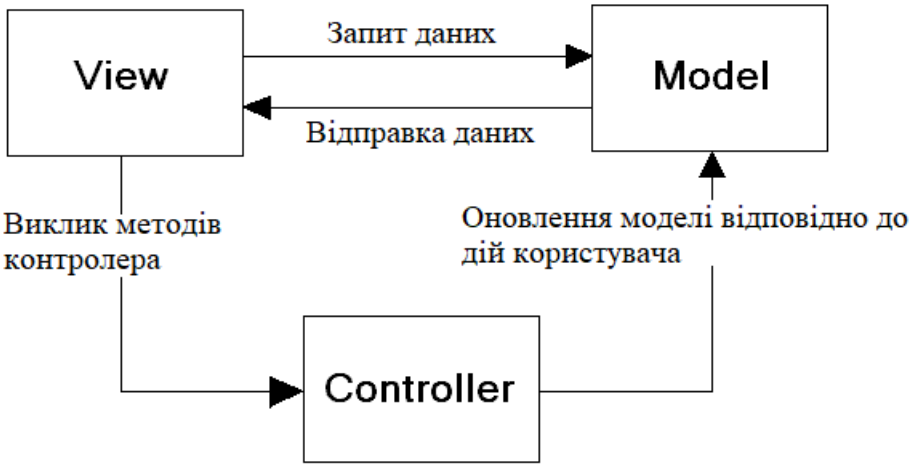


Рисунок 2.15 – Схема роботи програми

2.5. Розроблення архітектури

2.5.1. Огляд методів web-застосунку

У таблицях 2.1-2.4 наведено методи та їх опис в області Admin.

Таблиця 2.1 – Методи класу *CategoryController*

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку категорій	int productPage – номер сторінки товарів	View(categoryVM) – сторінка категорій і VM категорії
2.	Upsert	Редагувати категорію	Category category – модель категорії	View(category) - сторінка категорій і модель категорії
3.	Upsert	Додати категорію	int? id – ідентифікаційний номер категорії	View(category) - сторінка категорій і модель категорії
4.	GetAll	Отримати список категорій	-	Json(List<Category>) – список категорій у форматі Json
5.	Delete	Видалити категорію	int id - ідентифікаційний номер категорії	Json(new { success = true, message = "Delete Successful" }) – інформація про успішне видалення у форматі Json

Таблиця 2.2 – Методи класу *OrderController*

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку замовлень	-	View() – сторінка замовлень
2.	Details	Відобразити обране замовлення	int id - ідентифікаційний номер замовлення	View(OrderVM) – сторінка замовлення і VM замовлення
3.	StartProcessing	Змінити статус замовлення на "В обробці"	int id - ідентифікаційний номер замовлення	RedirectToAction("Index") – переадресація на сторінку Index
4.	ShipOrder	Змінити статус замовлення на "Завершено"	int id - ідентифікаційний номер замовлення	RedirectToAction("Index") – переадресація на сторінку Index
5.	CancelOrder	Змінити статус замовлення на "Скасовано"	int id - ідентифікаційний номер замовлення	RedirectToAction("Index") – переадресація на сторінку Index
6.	UpdateOrderDetails	Оновити деталі замовлення	-	RedirectToAction("Details", "Order", id)
7.	GetOrderList	Отримати список замовлень	string status – статус замовлень, які потрібно отримати	Json(new { data = orderHeaderList }) - список замовлень у форматі Json

Таблиця 2.3 – Методи класу *ProductController*

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку товарів	-	View() – сторінка товарів
2.	AddColors	Додати кольори до товару	int? productId - ідентифікаційний номер товару	View(product) – сторінка товарів і товар
3.	AddColors	Додати кольори до товару	Product product - товар	RedirectToAction(nameof(Index)) - переадресація на сторінку Index
4.	DeleteImages	Видалити зображення	int? id - ідентифікаційний номер товару	RedirectToAction(nameof(Index)) - переадресація на сторінку Index
5.	Upsert	Додати товар	int? id - ідентифікаційний номер товару	View(productVM) - сторінка товарів і VM товару
6.	Upsert	Додати товар	ProductVM productVM - VM товару, int colorsQuantity- кількість кольорів	View(productVM) - сторінка товарів і VM товару
7.	GetAll	Отримати список товарів	-	Json(new { data = allObj })

Таблиця 2.4 – Методи класу *UserController*

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку користувачів	-	View() – сторінка користувачів
2.	GetAll	Отримати список товарів	-	Json(new { data = userList })
3.	LockUnlock	Заблокувати/Розблокувати користувача	string id - ідентифікаційний рядок користувача	Json(new { success = true, message = "Operation Successful." })

У таблицях 2.5-2.6 наведено методи та їх опис в області Customer.

Таблиця 2.5 – Методи класу CartController

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку кошиків покупок	-	View(ShoppingCartVM)– сторінка кошику покупок і VM кошику покупок
2.	Plus	Збільшити на 1 кількість обраного товару в кошику покупок	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(name of(Index)) - переадресація на сторінку Index
3.	Minus	Зменшити на 1 кількість обраного товару в кошику покупок	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(name of(Index)) - переадресація на сторінку Index
4.	Remove	Прибрати товар з кошику покупок	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(name of(Index)) - переадресація на сторінку Index

Продовження таблиці 2.5

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
5.	Clear	Очистити кошик покупок	ShoppingCartVM cartVM - VM кошику покупок	RedirectToAction(nameof(Index)) - переадресація на сторінку Index
6.	PlusSummary	Збільшити на 1 кількість обраного товару в розділі Summary	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(nameof(Summary)) - переадресація на сторінку Summary
7.	MinusSummary	Зменшити на 1 кількість обраного товару в розділі Summary	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(nameof(Summary)) - переадресація на сторінку Summary
8.	RemoveSummary	Прибрати товар в розділі Summary	int cartId - ідентифікаційний номер товару в кошику покупок	RedirectToAction(nameof(Summary)) - переадресація на сторінку Summary

Продовження таблиці 2.5

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
9.	IndexPOST	Передати дані з кошику покупок до розділу Summary	ShoppingCartVM cartVMObject - VM кошику покупок	RedirectToAction(nameof(Summary)) - переадресація на сторінку Summary
10.	Summary	Відобразити сторінку Summary	-	View(ShoppingCartVM)– сторінка Summary і VM кошику покупок
11.	SummaryPost	Підтвердити замовлення та відправити його на сервер	ShoppingCartVM cartVMObject - VM кошику покупок	RedirectToAction("OrderConfirmation", "Cart", new { id = ShoppingCartVM.OrderHeader.Id }) – переадресація на сторінку підтвердження замовлення
12.	OrderConfirmation	Відобразити сторінку підтвердження замовлення	int id - ідентифікаційний номер замовлення	View(id) – сторінка підтвердження замовлення і ідентифікаційний номер замовлення

Таблиця 2.6 – Методи класу HomeController

№ п/п	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1.	Index	Відобразити головну сторінку	string categoryName – назва категорї, string searchString – рядок пошуку	View(productList) – головна сторінка і список товарів
2.	Details	Відобразити сторінку товару	int id - ідентифікаційний номер товару	View(cartObj) – сторінка товару і об’єкт типу ShoppingCart
3.	AddToCart	Додати товар у кошик покупок	int id - ідентифікаційний номер товару	RedirectToAction(nameof(Details)) – переадресація на сторінку Details
4.	AddToCartFromIndex	Додати товар у кошик покупок з головної сторінки	int id - ідентифікаційний номер товару	RedirectToAction(nameof(Cart)) – переадресація на сторінку Cart
5.	Details	Відобразити сторінку товару	ShoppingCart CartObject – об’єкт типу ShoppingCart	View(cartObj) – сторінка товару і об’єкт типу ShoppingCart

2.5.2 Розробка бази даних з Entity Framework Code First

У даному розділі буде розглянуто процес розробки моделі бази даних з використанням Entity Framework Code First. Entity Framework є фреймворком, розробленим компанією Microsoft, який дозволяє розробникам працювати з базами даних, використовуючи об'єктно-орієнтований підхід. Code First дозволяє визначати модель бази даних за допомогою класів і атрибутів, а сама база даних буде автоматично створена на основі цієї моделі.

Розробка моделі бази даних з Entity Framework Code First може бути розбита на кілька етапів. Спочатку необхідно визначити класи, які відповідають таблицям бази даних. Кожен клас представляє таблицю, а властивості класу відображають стовпці цієї таблиці. Можна використовувати атрибути для встановлення правил валідації або відношень між таблицями.

Після визначення класів створюється контекст бази даних, який успадковується від класу ApplicationDbContext з Entity Framework. Контекст бази даних забезпечує зв'язок між моделлю бази даних і самою базою даних. В ньому визначаються набори даних (DbSet), які представляють таблиці бази даних.

Після налаштування контексту бази даних можна виконати міграції для створення або оновлення бази даних на основі моделі. Міграції дозволяють зберігати історію змін моделі і автоматично застосовувати необхідні зміни до бази даних. За допомогою команд міграцій можна створювати нові таблиці, стовпці, індекси або змінювати існуючі.

У цьому розділі будуть детальніше розглянуті кроки розробки моделі бази даних з використанням Entity Framework Code First, а також надані приклади коду для кожного з цих кроків. Це допоможе отримати глибше розуміння основних концепцій та навичок роботи з Entity Framework і розробки моделі бази даних з Code First підходом.

Процес розробки моделі бази даних можна побачити на рисунках 2.16-2.32

Першим кроком створюємо моделі, на основі яких згодом будуть створені таблиці бази даних.

Створюємо бібліотеку ClothesShop.Models як на рисунку 2.16 для того, щоб зберігати моделі окремо від основного проєкту.

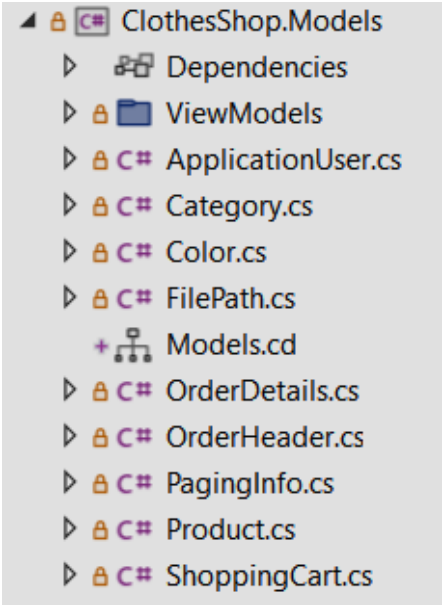


Рисунок 2.16 – Бібліотека ClothesShop.Models

Вміст моделей вказаний на рисунках 2.17-2.25:

```
namespace ClothesShop.Models
{
    public class ApplicationUser : IdentityUser
    {
        public string StreetAddress { get; set; }
        public string City { get; set; }
        public bool IsMale { get; set; }
        [Required]
        public string Name { get; set; }

        [NotMapped]
        public string Role { get; set; }
    }
}
```

Рисунок 2.17 – Модель ApplicationUser

```
namespace ClothesShop.Models
{
    public class Category
    {
        [Key]
        public int Id { get; set; }

        [Display(Name = "Category MainName")]
        [Required]
        [MaxLength(50)]
        public string MainName { get; set; }

        [Display(Name = "Category SubName")]
        [Required]
        [MaxLength(50)]
        public string SubName { get; set; }
    }
}
```

Рисунок 2.18 – Модель Category

```
namespace ClothesShop.Models
{
    public class Color
    {
        public int Id { get; set; }
        public string Name { get; set; }
    }
}
```

Рисунок 2.19 – Модель Color

```
namespace ClothesShop.Models
{
    public class FilePath
    {
        public int Id { get; set; }
        public string Path { get; set; }
    }
}
```

Рисунок 2.20 – Модель FilePath

```
namespace ClothesShop.Models
{
    public class OrderDetails
    {
        [Key]
        public int Id { get; set; }

        [Required]
        public int OrderId { get; set; }
        [ForeignKey("OrderId")]
        public OrderHeader OrderHeader { get; set; }


        [Required]
        public int ProductId { get; set; }
        [ForeignKey("ProductId")]
        public Product Product { get; set; }
        public string Size { get; set; }
        public string Color { get; set; }

        public int Count { get; set; }
        public double Price { get; set; }
    }
}
```

Рисунок 2.21 – Модель OrderDetails

```
namespace ClothesShop.Models
{
    public class PagingInfo
    {
        public int TotalItem { get; set; }

        public int ItemsPerPage { get; set; }

        public int CurrentPage { get; set; }

        public string urlParam { get; set; }
        public int TotalPage => (int)Math.Ceiling((decimal)TotalItem / ItemsPerPage);
    }
}
```

Рисунок 2.22 – Модель PagingInfo

```
namespace ClothesShop.Models
{
    public class OrderHeader
    {
        [Key]
        public int Id { get; set; }

        public string ApplicationUserId { get; set; }
        [ForeignKey("ApplicationUserId")]
        public ApplicationUser ApplicationUser { get; set; }

        [Required]
        public DateTime OrderDate { get; set; }
        [Required]
        public DateTime ShippingDate { get; set; }
        [Required]
        public Double OrderTotal { get; set; }
        public string OrderStatus { get; set; }
        public string PaymentStatus { get; set; }
        public DateTime PaymentDate { get; set; }
        public string City { get; set; }
        public string StreetAddress { get; set; }
        public string Comment { get; set; }
        [Required]
        public string PhoneNumber { get; set; }
        [Required]
        public string Name { get; set; }
        [Required]
        public string DeliveryMethod { get; set; }
    }
}
```

Рисунок 2.23 – Модель OrderHeader

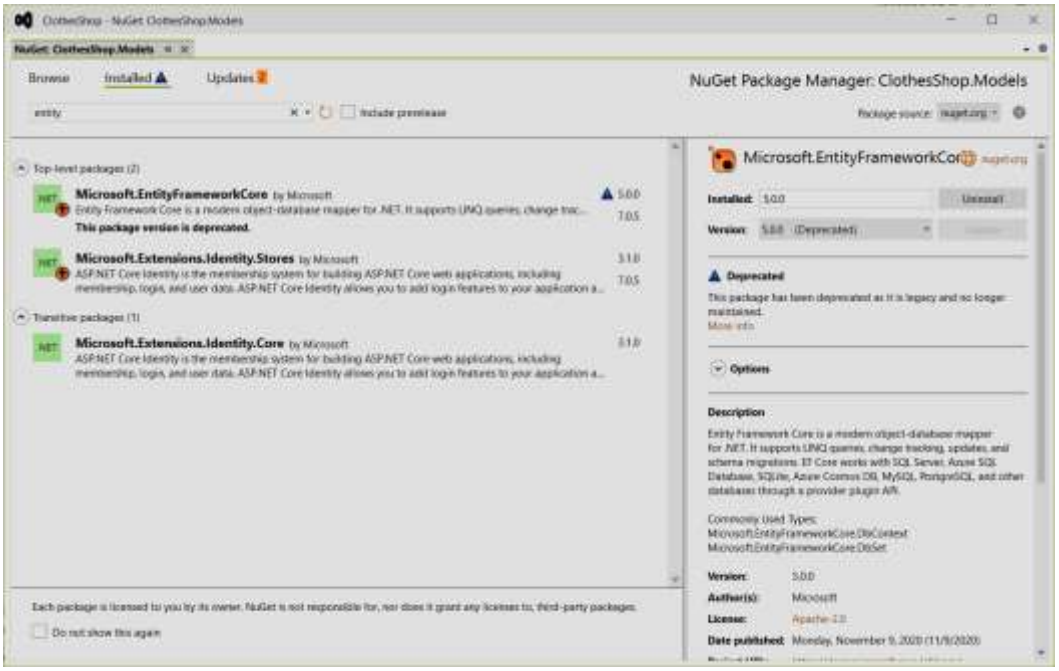
```
namespace ClothesShop.Models
{
    public class Product
    {
        [Key]
        public int Id { get; set; }
        [Required]
        public string Title { get; set; }
        public string Description { get; set; }
        public string Consist { get; set; }
        public string CareConditions { get; set; }
        [Required]
        [Range(1, 10000000)]
        public double Price { get; set; }
        public List<FilePath> ImagesUrl { get; set; }
        public List<Color> Colors { get; set; }
        [Required]
        public int CategoryId { get; set; }
        [ForeignKey("CategoryId")]
        public Category Category { get; set; }

    }
}
```

Рисунок 2.24 – Модель Product

```
namespace ClothesShop.Models
{
    public class ShoppingCart
    {
        public ShoppingCart()
        {
            Count = 1;
        }
        [Key]
        public int Id { get; set; }

        public string ApplicationUserId { get; set; }
        [ForeignKey("ApplicationUserId")]
        public ApplicationUser ApplicationUser { get; set; }

        public int ProductId { get; set; }
        [ForeignKey("ProductId")]
        public Product Product { get; set; }
        public string Size { get; set; }
        public string Color { get; set; }
        public Double OneTypeTotal { get; set; }

        [Range(1, 1000, ErrorMessage = "Please enter a value between 1 and 1000")]
        public int Count { get; set; }

        [NotMapped]
        public double Price { get; set; }

    }
}
```

Рисунок 2.25 – Модель ShoppingCart

Налаштовуємо Entity Framework в середовищі розробки



Створюємо контекст ApplicationDbContext в окремій бібліотеці ClothesShop.DataAccess на рисунку 2.26.

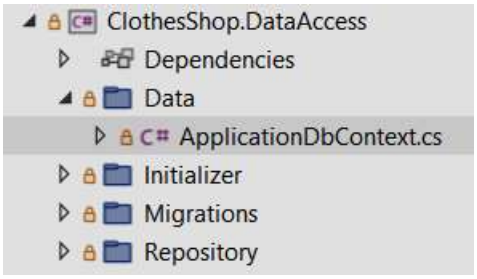


Рисунок 2.26 – Бібліотека DataAccess

У файлі ApplicationDbContext визначаємо контекст, який походить від System.Data.Entity.DbContext і надає типізований DbSet<TEntity> для кожного класу в нашій моделі на рисунку 2.27.

```
using ClothesShop.Models;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;

namespace ClothesShop.DataAccess.Data
{
    public class ApplicationDbContext : IdentityDbContext
    {
        public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
            : base(options)
        {
        }
        public DbSet<Category> Categories { get; set; }
        public DbSet<Product> Products { get; set; }
        public DbSet<ApplicationUser> ApplicationUsers { get; set; }
        public DbSet<ShoppingCart> ShoppingCarts { get; set; }
        public DbSet<OrderHeader> OrderHeaders { get; set; }
        public DbSet<OrderDetails> OrderDetails { get; set; }
        public DbSet<FilePath> FilePaths { get; set; }
        public DbSet<Color> Colors { get; set; }
    }
}
```

Рисунок 2.27 – Контекст ApplicationDbContext

Додаємо сервіс AddDbContext у файл Startup.cs як на рисунку 2.28.

```
services.AddDbContext<ApplicationDbContext>(options =>
    options.UseSqlServer(
        Configuration.GetConnectionString("DefaultConnection")));
```

Рисунок 2.28 – AddDbContext у файлі Startup.cs

У файлі appsettings.json вказуємо рядок підключення до бази даних з назвою ClothesShopper.

Рисунок 2.29 – Файл appsettings.json

Встановлюємо Microsoft SQL Server Management Studio.

Командою Add-Migration створюємо міграцію як на рисунку 2.30, при чому потрібно обрати Default project: ClothesShop.DataAccess.

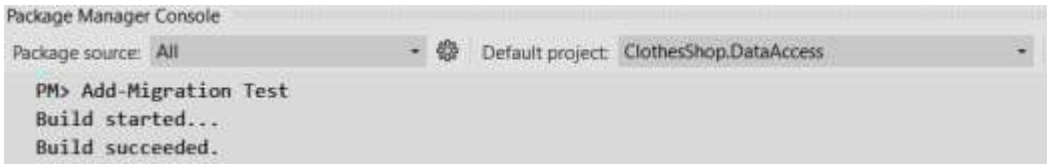


Рисунок 2.30 – Створення міграції з назвою Test

Рисунок 2.31 – Створення/оновлення бази даних

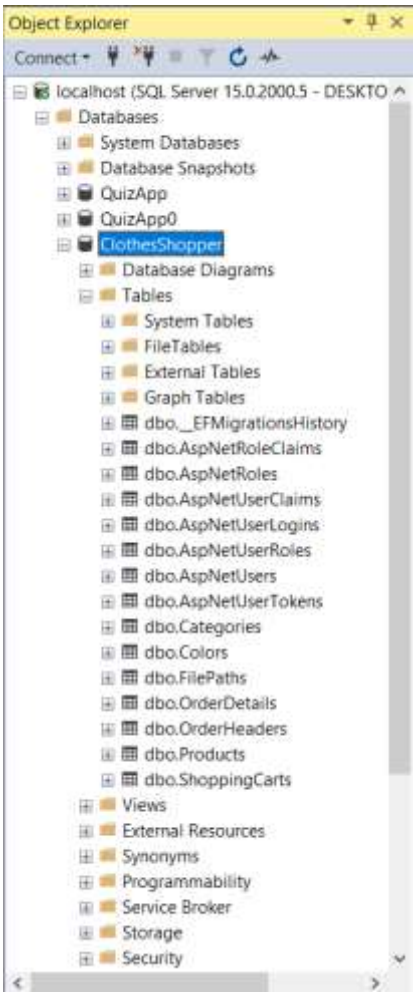


Рисунок 2.32 – База даних в MSSQL

2.5.3 Опис структур таблиць бази даних

В процесі розробки була створена база даних, яка буде зберігати всю інформацію інформаційної автоматизованої системи продажу речей. Далі розглянемо всі таблиці бази даних у табличному форматі: ім’я таблиці, назва стовпця, тип даних та детальна інформація (таблиці 2.6 – 2.12).

Таблиця 2.6 – Таблиця категорій

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
Categories	Id	int	Ідентифікатор категорії
	MainName	nvarchar	Заголовок категорії
	SubName	nvarchar	Підзаголовок категорії

Таблиця 2.7 – Таблиця кольорів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
Colors	Id	int	Ідентифікатор кольору
	Name	nvarchar	Назва кольору
	ProductId	int	Ідентифікатор товару

Таблиця 2.8 – Таблиця шляхів до файлів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
FilePath	Id	int	Ідентифікатор шляху до файлу
	Path	nvarchar	Шлях до файлу
	ProductId	int	Ідентифікатор товару

Таблиця 2.9 – Таблиця деталей замовлення

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
OrderDetails	Id	int	Ідентифікатор деталей замовлення
	OrderId	int	Номер замовлення
	ProductId	int	Ідентифікатор товару
	Size	nvarchar	Розмір товару
	Color	nvarchar	Колір товару
	Count	int	Кількість товару
	Price	float	Ціна товару

Таблиця 2.10 – Таблиця заголовків замовлення

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
OrderDetails	Id	int	Ідентифікатор заголовку замовлення
	ApplicationUserId	nvarchar	Ідентифікатор користувача
	OrderDate	datetime2	Дата замовлення
	ShippingDate	datetime2	Дата відправки замовлення
	OrderTotal	float	Сума замовлення
	OrderStatus	nvarchar	Статус замовлення
	PaymentStatus	nvarchar	Статус оплати
	PaymentDate	datetime2	Дата оплати
	City	nvarchar	Місто доставки
	StreetAddress	nvarchar	Вулиця доставки
	Comment	nvarchar	Коментар користувача
	PhoneNumber	nvarchar	Номер телефону користувача
	Name	nvarchar	Ім’я користувача
	DeliveryMethod	nvarchar	Спосіб доставки замовлення

Таблиця 2.11 – Таблиця товарів

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
Products	Id	int	Ідентифікатор товару
	Title	nvarchar	Назва товару
	Description	nvarchar	Опис товару
	Consist	nvarchar	Склад товару
	CareConditions	nvarchar	Умови догляду за товаром
	Price	float	Ціна товару
	CategoryId	int	Ідентифікатор категорії

Таблиця 2.12 – Таблиця кошиків покупок

Назва таблиці	Назва стовпця	Тип даних	Детальна інформація
ShoppingCarts	Id	int	Ідентифікатор кошику покупок
	ApplicationUserId	nvarchar	Ідентифікатор користувача
	ProductId	int	Ідентифікатор товару
	Size	nvarchar	Розмір товару
	Color	nvarchar	Колір товару
	OneTypeTotal	float	Загальна вартість
	Count	int	Кількість товару

2.6 Висновки до розділу

У даному розділі було розглянуто моделювання та конструювання інформаційної автоматизованої системи продажу речей. Було продемонстровано, що система складається з багатьох комплексних рішень і компонентів.

Розглянуто можливості програмного забезпечення, яке використовувалося для розробки.

Побудована блок-схема методів та класів програми.

Створена база даних за допомогою Entity Framework, і патерну Code First.

Розгляуто створення проєкту в середовищі розробки Visual Studio.

3. АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Результати тестування входу до системи і здійснення покупки

У цьому розділі буде проведене тестування інформаційної автоматизованої системи продажу речей. Для прикладу візьмемо наповнення магазину одягу.

Тестуємо вхід у систему (рисунок 3.1).

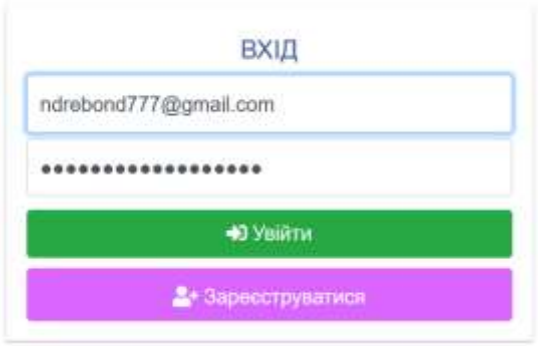


Рисунок 3.1 – Вхід у систему

Після входу у систему переходимо у меню користувача (рисунок 3.2).

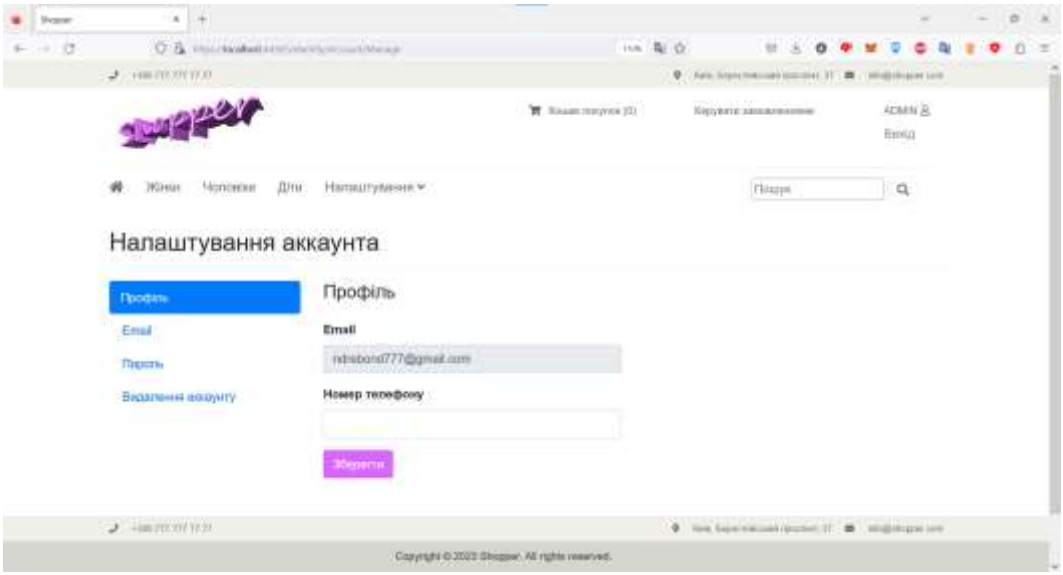


Рисунок 3.2 – Меню користувача

Як видно на рисунку 3.2, вхід до системи пройшов успішно.

Тестування додавання додавання товару в кошик. Натискаємо на головній сторінці (рисунок 3.3) кнопку «Купити» - і переходимо у кошик покупок з обраним товаром.

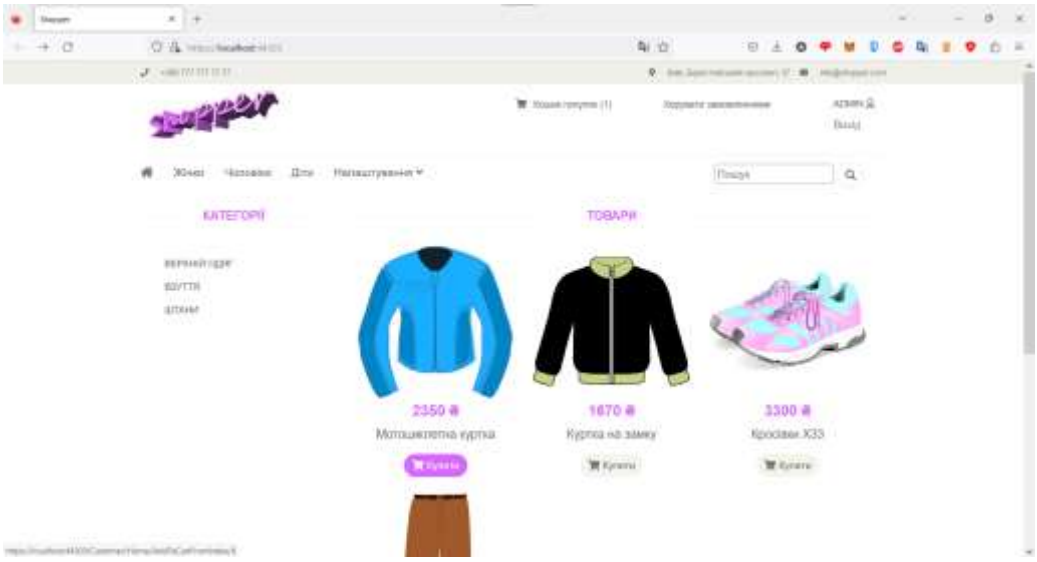


Рисунок 3.3 – Головна сторінка

Додавання товару пройшло успішно, як видно на рисунку 3.4 товар доданий в кількості одної одиниці в кошик покупок.

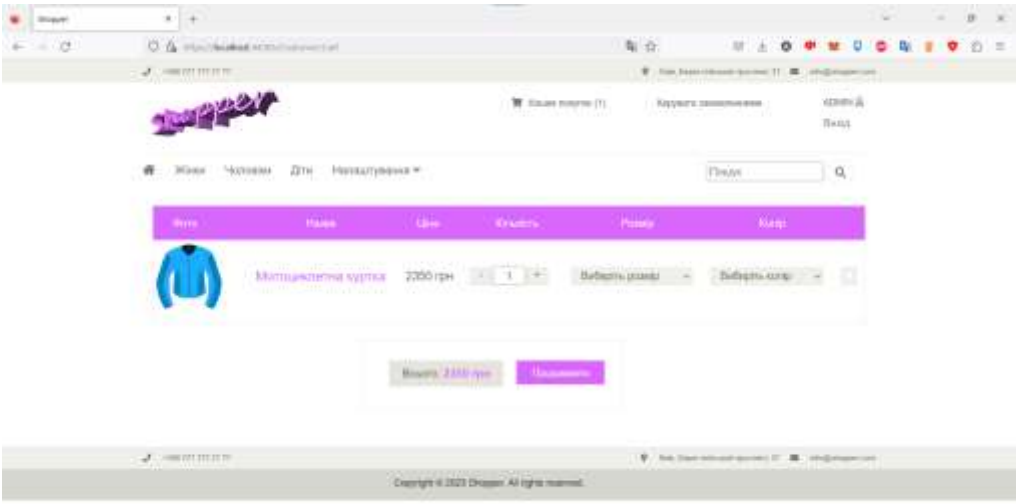


Рисунок 3.4 – Кошик покупок

Тестуємо збільшення кількості товарів, змінюємо розмір на S, колір на жовтий (рисунок 3.5) – і натискаємо «Продовжити».

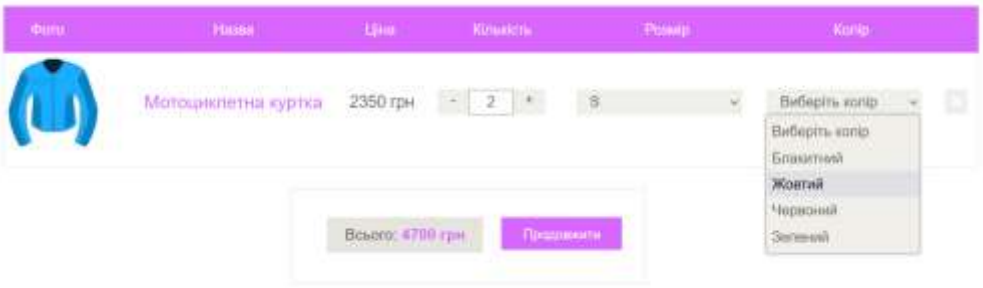


Рисунок 3.5 – Кошик покупок з обраними параметрами товару

На наступній сторінці «Підсумок замовлення» (рисунок 3.6) параметри передалися коректно, дві куртки, розмір S і жовтий колір. Заповнюємо контакті дані і натискаємо «Розмістити замовлення».

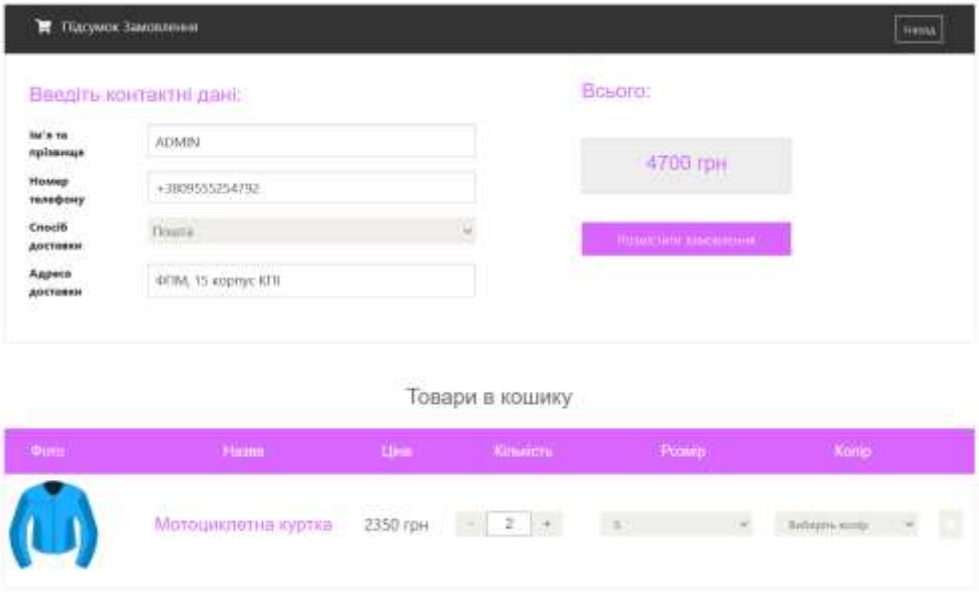


Рисунок 3.6 - Підсумок замовлення

Отримуємо повідомлення, що «Замовлення прийнято успішно!», номер замовлення 4 (рисунок 3.7).

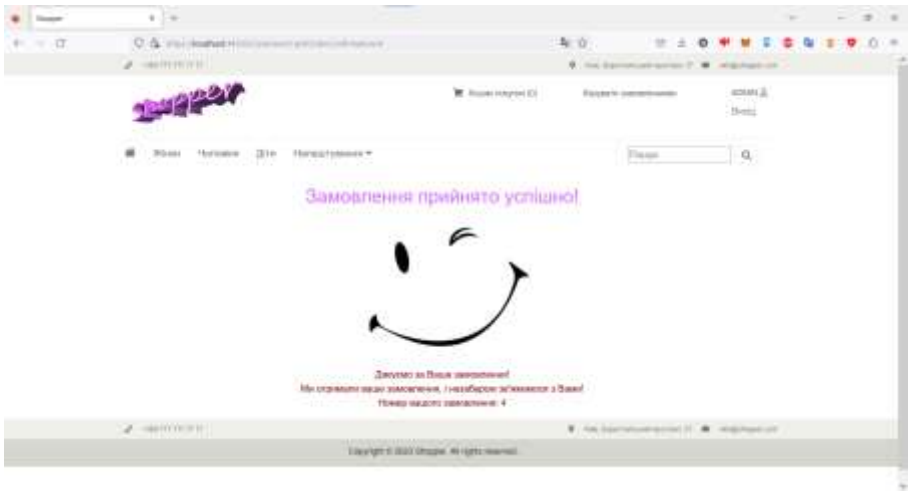


Рисунок 3.7 – Підтвердження замовлення

У списку замовлень присутнє замовлення під номером 4 з коректними контактними даними (рисунок 3.7).

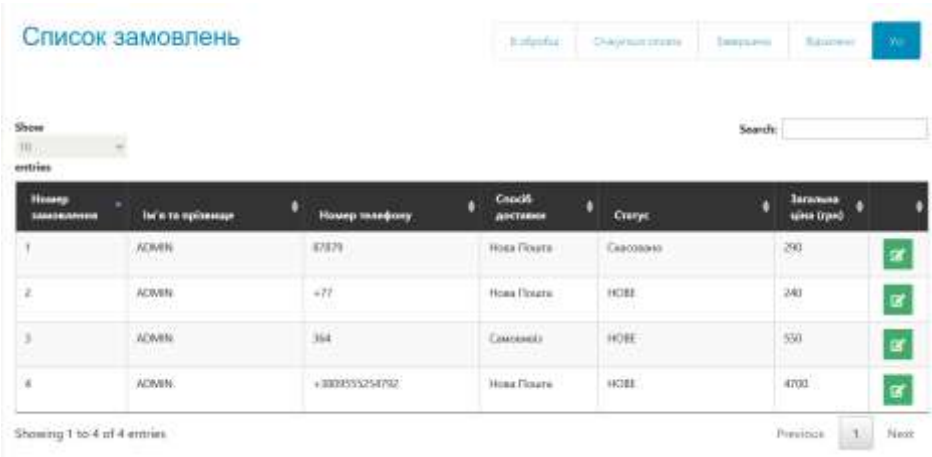


Рисунок 3.7 – Список замовлень

Товари у замовленні теж правильні (рисунок 3.8).

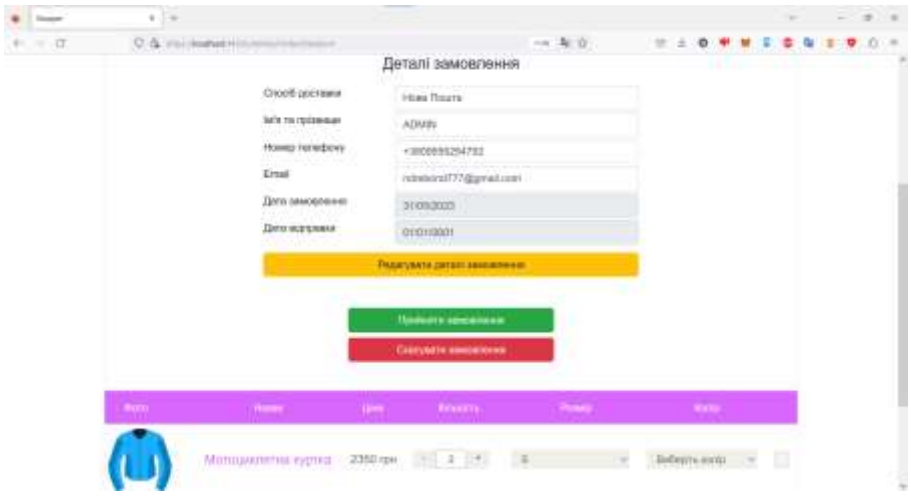


Рисунок 3.8 – Деталі замовлення

Отже, процес замовлення товарів відбувається коректно.

3.2 Тестування реєстрації користувача

Залишаємо порожні поля – та натискаємо «Зареєструватись» (рисунок 3.9). Отримуємо повідомлення про те, які поля обов’язкові.

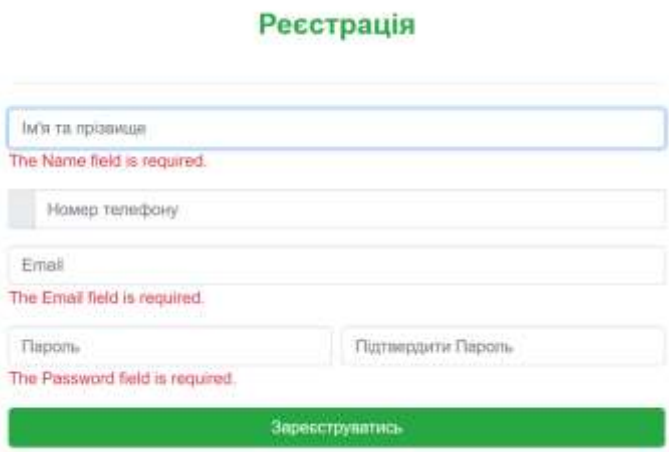


Рисунок 3.9 – Реєстрація з порожніми полями

Вводимо дані для реєстрації в некоректному форматі (рисунок 3.10) – і отримуємо повідомлення про те, що дані введенні некоректно.

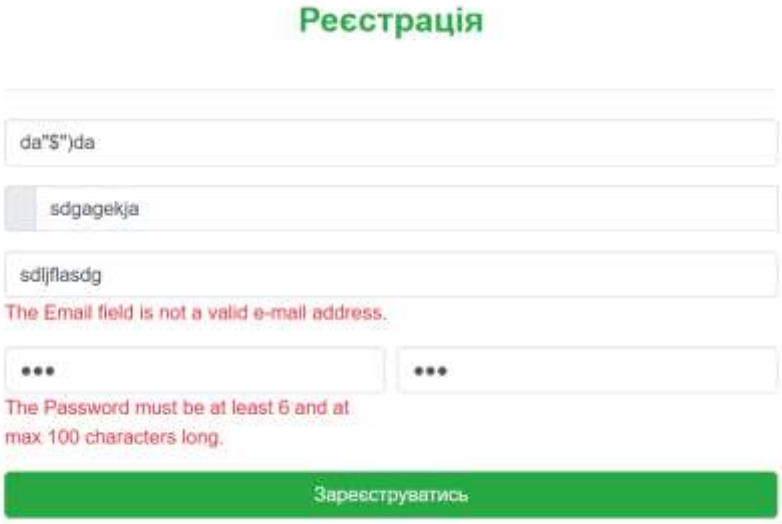


Рисунок 3.10 – Реєстрація з полями у некоректному форматі

При введенні даних у коректному форматі користувач реєструється без помилок.

3.3 Тестування функціоналу адміністратора

Адміністратор має наступні можливості:

- керувати замовленнями;
- керувати категоріями;
- керувати товарами;
- керувати користувачами.

Для прикладу спробуємо додати і видалити товар. Переходимо у вкладку «Налаштування» і обираємо «Товари», після чого опиняємося на сторінці зі списком товарів (рисунок 3.11).

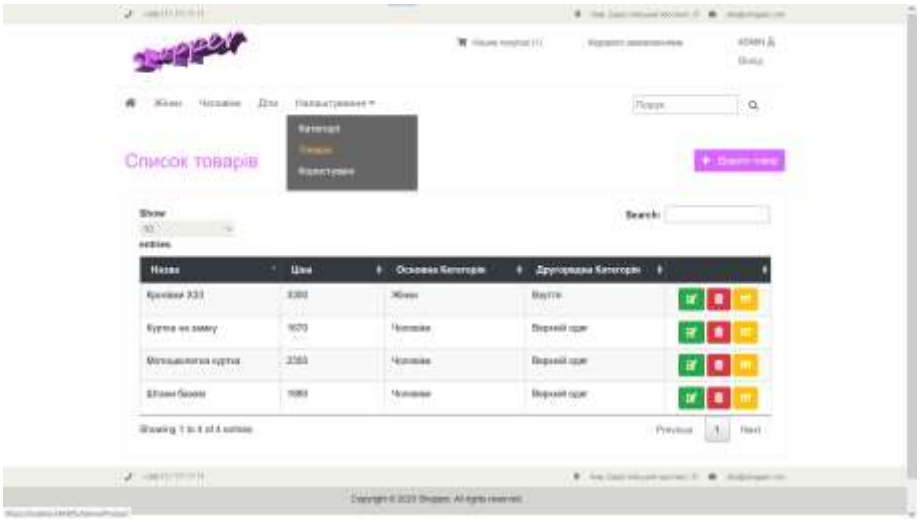


Рисунок 3.11 – Список товарів

Наступним кроком натискаємо на кнопку «Додати товар». Заповнюємо необхідні поля інформацією і тиснемо «Продовжити» (рисунок 3.13).

Додаємо кольори (рисунок 3.12).

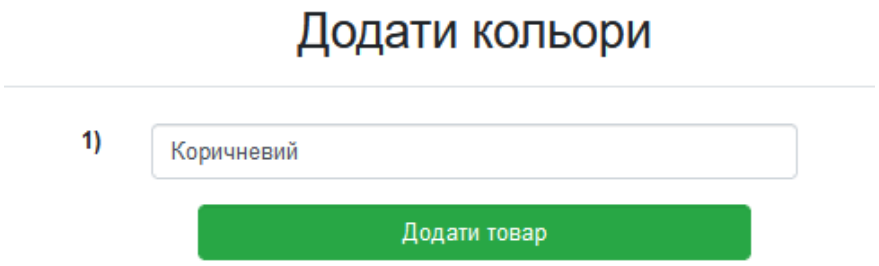


Рисунок 3.12 – Додати кольори

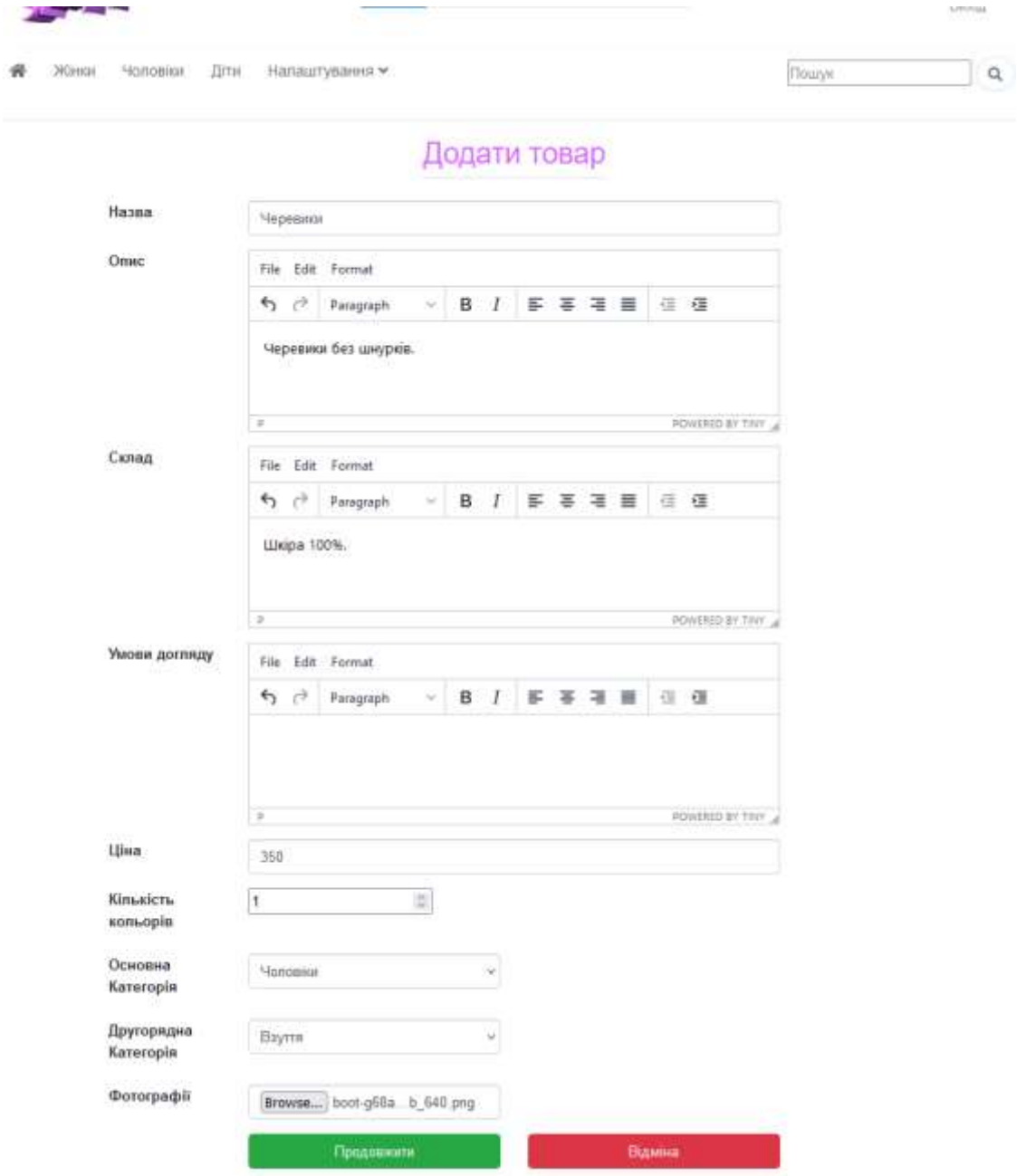


Рисунок 3.13 – Сторінка «Додати товар»

Товар успішно доданий (рисунок 3.14).

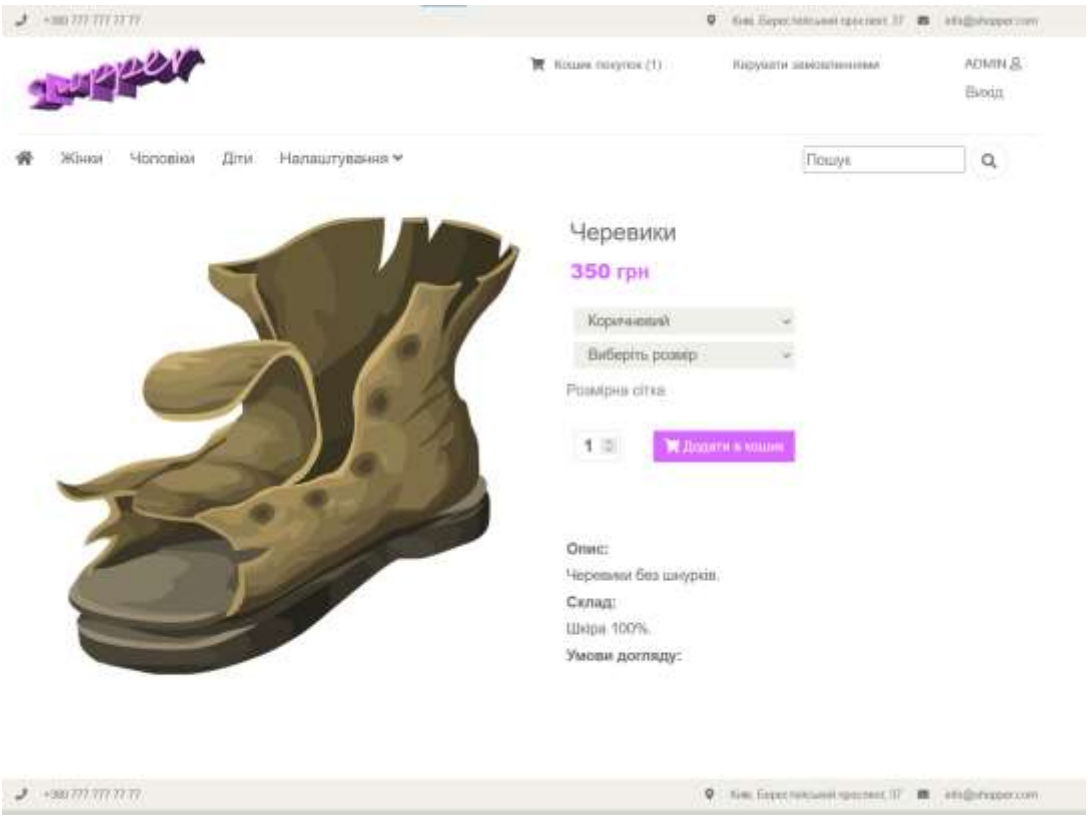


Рисунок 3.14 – Сторінка доданого товару під час тестування

3.4. Висновки до розділу

В даному розділі було протестовано роботу інформаційної автоматизованої системи продажу речей. Система правильно спрацьовувала при введенні некоректних даних. В залежності від привілегій користувача надавала доступ тільки до дозволених розділів. Додавання товару пройшло успішно, уся інформація була збережена та записана в базу даних. Таким чином, система працює стабільно і коректно.

ВИСНОВКИ

У рамках дипломного проєкту була розроблена інформаційна автоматизована система продажу речей, яка має великий потенціал для оптимізації та покращення процесів у сфері електронної комерції. Ця система включає в себе ряд функцій, які допомагають власникам магазинів ефективно керувати своїм бізнесом, забезпечуючи зручну і швидку покупку та продаж товарів.

Під час розробки системи було враховано потреби і вимоги сучасних споживачів, їхні побажання щодо простоти, швидкості та безпеки у процесі покупок. Система пропонує зручний інтерфейс для замовлення товарів, забезпечує швидке оброблення замовлень та можливість відстеження доставки. Крім того, система включає модуль для управління запасами, що дозволяє ефективно контролювати наявність товарів на складі.

Результати тестування показали, що система продажу речей є стабільною, надійною та функціональною. Вона може успішно впроваджуватись у різних сферах електронної комерції, від малих інтернет-магазинів до великих роздрібних мереж. Бізнес-власники матимуть змогу оптимізувати свою роботу, автоматизувати багато рутинних процесів і підвищити рівень задоволення своїх клієнтів.

Однак, важливо зазначити, що система продажу речей не є універсальним рішенням і може потребувати налаштувань та додаткових функцій для задоволення конкретних потреб бізнесу. Крім того, з огляду на швидкий розвиток технологій і зміну вимог ринку, необхідно постійно оновлювати та вдосконалювати систему, щоб вона залишалася конкурентоспроможною.

Загалом, розроблена інформаційна автоматизована система продажу речей є важливим кроком у напрямку вдосконалення електронної комерції. Вона пропонує широкий спектр функцій для оптимізації бізнес-процесів та полегшення життя споживачів. Дана система може стати цінним інструментом для підвищення ефективності та конкурентоспроможності компаній, що займаються продажем речей через Інтернет.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. ISO/IEC 2382:2015, Information technology — Vocabulary — Part 1: Terms and definitions. [Чинний від 2015-05].
2. **Інтернет-ресурс**

   Entity Framework. – Режим доступу: https://learn.microsoft.com/en-us/aspnet/entity-framework – Дата доступу : 03.05.2023.
3. **Інтернет-ресурс**

   MVC vs MVP vs MVVM. – Режим доступу: https://levelup.gitconnected.com/mvc-vs-mvp-vs-mvvm-35e0d4b933b4 – Дата доступу : 05.05.2023.
4. **Інтернет-ресурс**

   MVC, MVP, MVVM: Which One to Choose? – Режим доступу: https://www.makeuseof.com/mvc-mvp-mvvm-which-choose/ – Дата доступу : 05.05.2023.
5. **Інтернет-ресурс**

   Welcome To Learn Razor Pages. – Режим доступу: https://www.learnrazorpages.com/ – Дата доступу : 09.05.2023.
6. **Інтернет-ресурс**

   Microsoft Visual Studio. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio – Дата доступу : 12.05.2023.
7. **Інтернет-ресурс**

   JavaScript. – Режим доступу: https://developer.mozilla.org/en-US/docs/Web/JavaScript – Дата доступу : 08.05.2023.
8. **Інтернет-ресурс**

   Security, Authentication, and Authorization with ASP.NET MVC. – Режим доступу: https://learn.microsoft.com/en-us/aspnet/mvc/overview/security/ – Дата доступу : 16.05.2023.

9. Інтернет-ресурс

E-commerce worldwide - statistics & facts. – Режим доступу: https://www.statista.com/topics/871/online-shopping/#topicOverview – Дата доступу : 14.05.2023.

10. Інтернет-ресурс

HTTP Documentation. – Режим доступу: https://httpwg.org/specs/ – Дата доступу : 19.05.2023.

11. Інтернет-ресурс

.NET is open source. – Режим доступу: https://dotnet.microsoft.com/en-us/platform/open-source – Дата доступу : 21.05.2023.

12. Інтернет-ресурс

ASOS. – Режим доступу: https://web.archive.org/web/20131217035050/http://www.telegraph.co.uk/finance/newsbysector/retailandconsumer/8551844/ASOS-profits-jump-41pc-on-international-expansion.html – Дата доступу : 18.05.2023.

13. Інтернет-ресурс

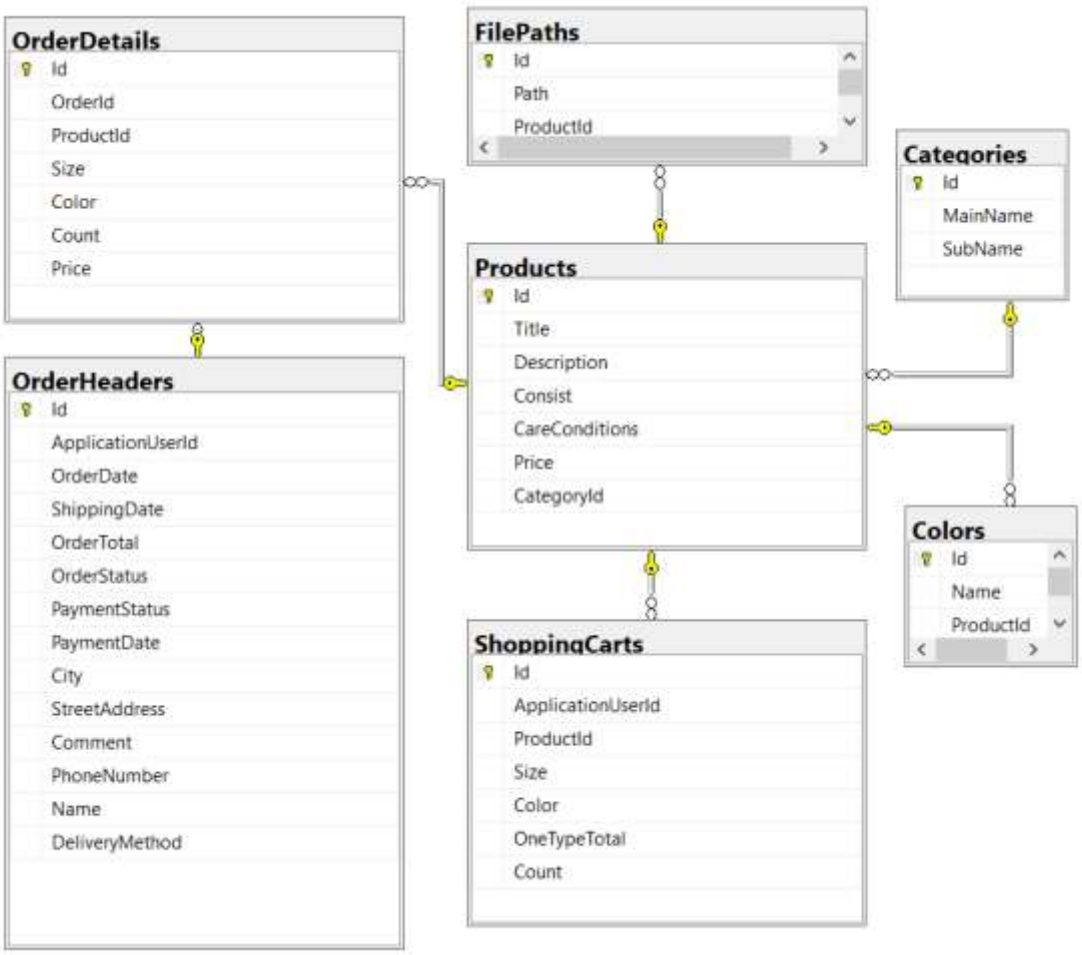
Аналіз діяльності ТОО "Central Trade". – Режим доступу : https://ukrbukva.net/page,9,75321-Analiz-deyatel-nosti-TOO-Central-Trade.html – Дата доступу : 19.05.2023.

14. Agile Software Development: The Business of Innovation / J. Highsmith, A. Cockburn., 2001. http://sunnyday.mit.edu/16.355/highsmith-agile.pdf– Дата доступу : 27.05.2023.

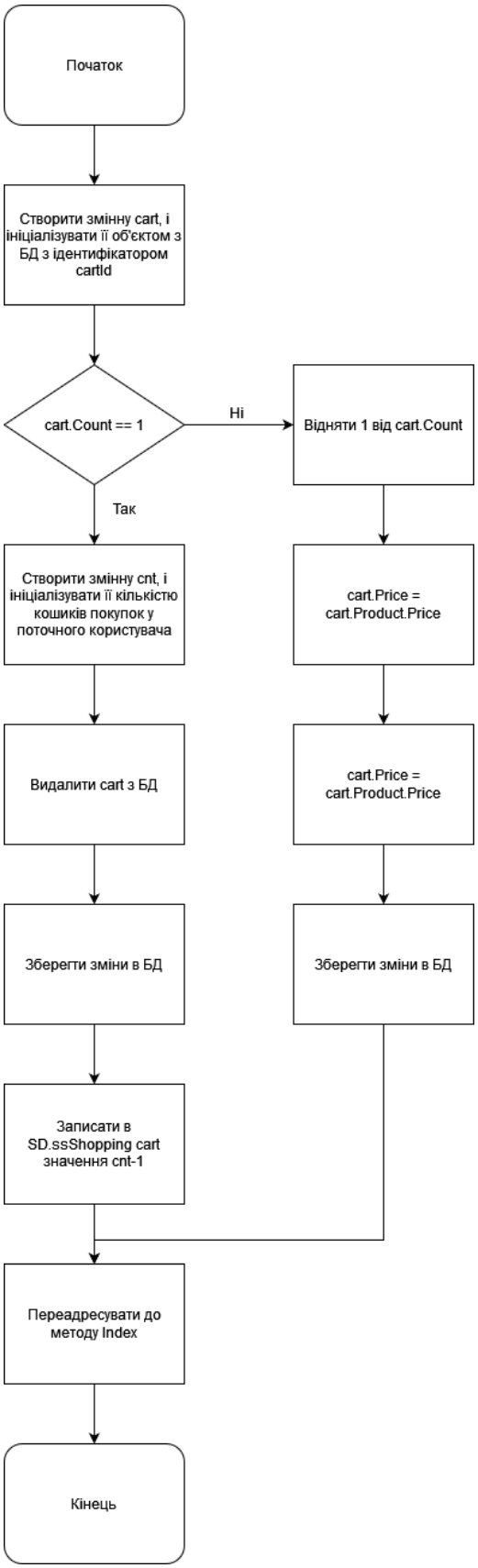
15. Debra Howcroft, John Carroll. A Proposed Methodology for Web Development. Manchester : ECIS 2000 Proceeding. 8 с.

ДОДАТКИ
до дипломного проєкту

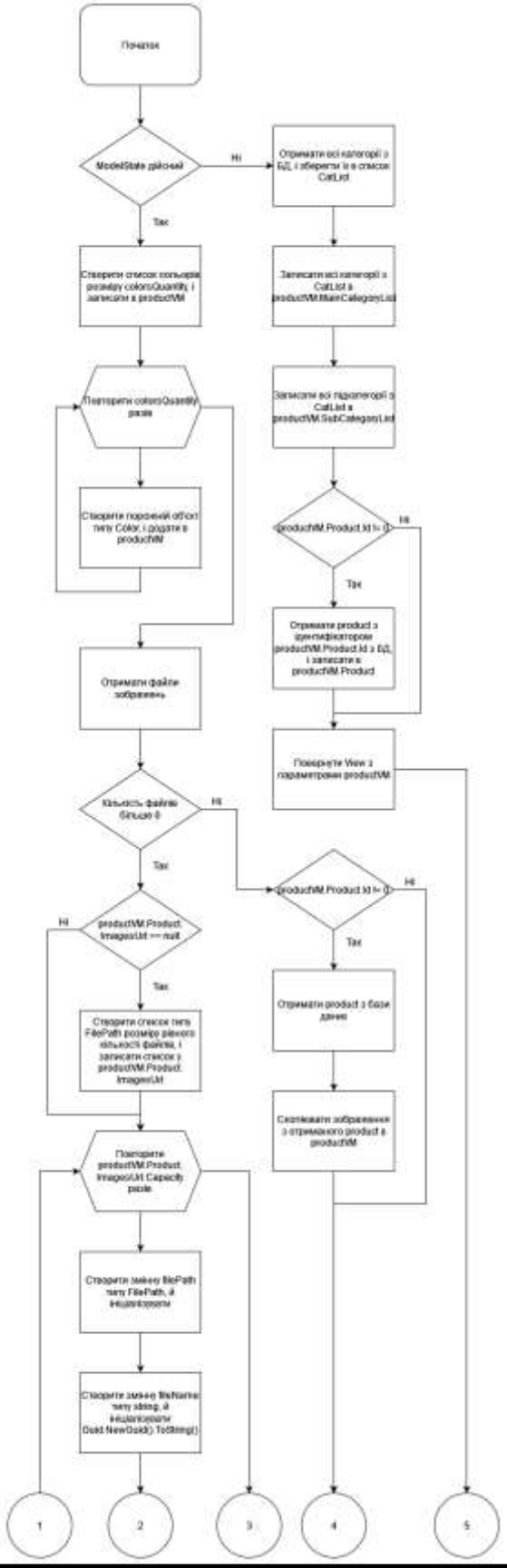
ДОДАТОК А. ГРАФІЧНІ МАТЕРІАЛИ

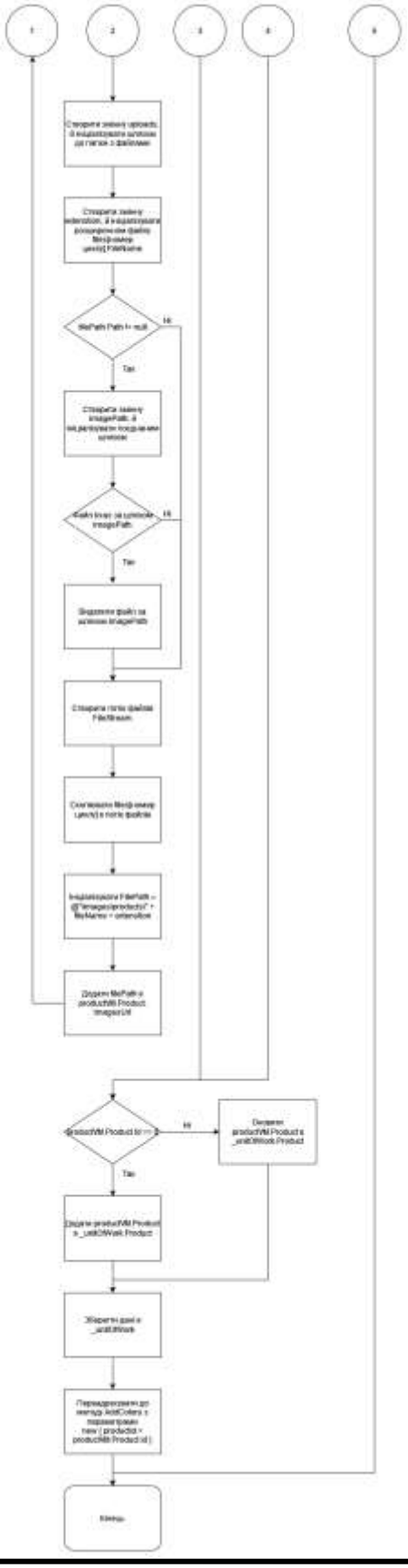


					ІАЛЦ.045440.005 Д1		
Змін	Арк.	№ докум.	Підпис	Дата			
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей	Літ.	Аркуш 1 / Аркушів 1
Перевірив		Радченко К.О.					
Н. контроль		Клятченко Я.М			***Схема бази даних***	КПІ ім. Ігоря Сікорського, ФПМ КВ-91	
Затвердив		Романкевич В.О.					

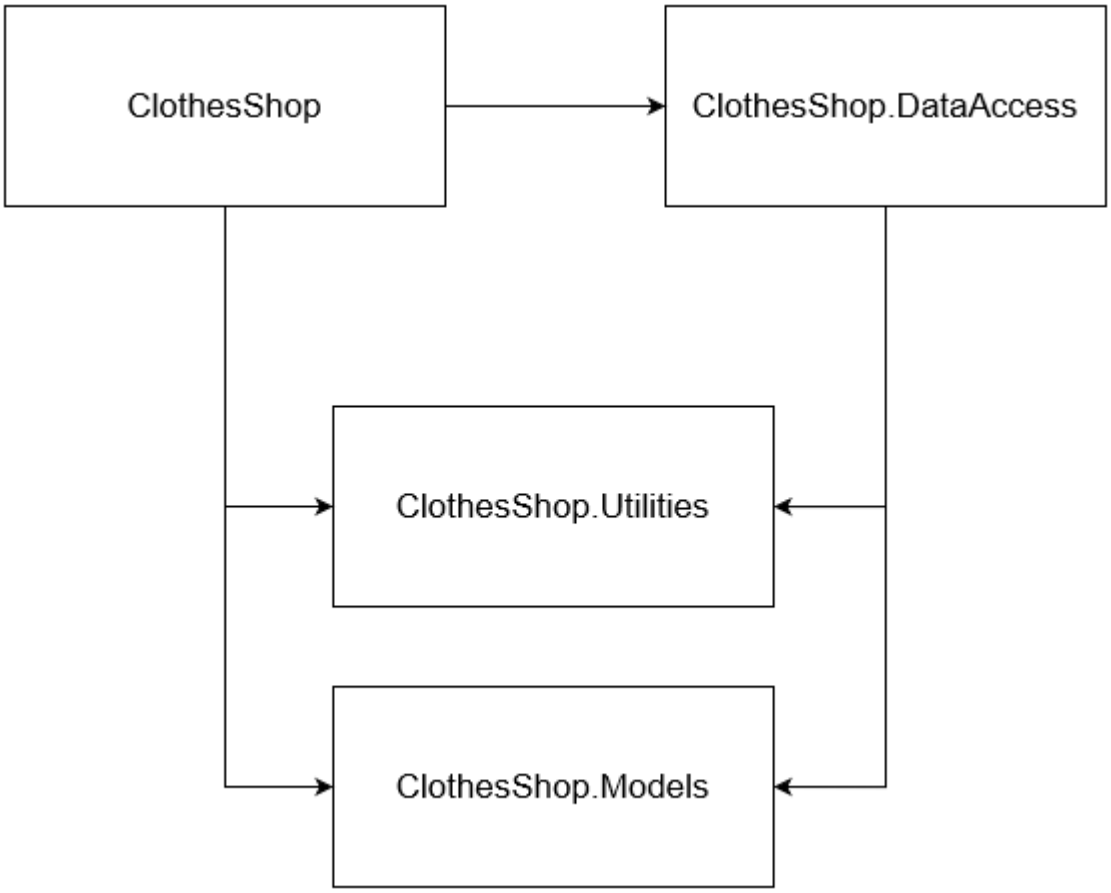


					ІАЛЦ.045440.006 Д2	
Змін	Арк.	№ докум.	Підпис	Дата		
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей ***Блок-схема методу Minus контролера CartController***	Літ. / Аркуш 1 / Аркушів 1
Перевірив		Радченко К.О.				КПІ ім. Ігоря Сікорського, ФПМ КВ-91
Н. контроль		.Клятченко Я.М				
Затвердив		Романкевич В.О.				





					ІАЛЦ.045440.007 ДЗ		
Змін	Арк.	№ докум.	Підпис	Дата			
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей ***Блок-схема додавання товару***	Літ.	Аркуш 1 / Аркушів 1
Перевірив		Радченко К.О.					
Н. контроль		Клятченко Я.М				КПІ ім. Ігоря Сікорського, ФПМ КВ-91	
Затвердив		Романкевич В.О.					



					ІАЛЦ.045440.008 Д4		
Змін	Арк.	№ докум.	Підпис	Дата			
Розробив		Бітлян А.В.			Інформаційна автоматизована система продажу речей ***Структурна схема взаємодії***	Літ.	Аркуш 1 / Аркушів 1
Перевірив		Радченко К.О.					
Н. контроль		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПМ КВ-91	
Затвердив		Романкевич В.О.					