МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: Маркетплейс на базі CRM-систем та управління даними

Виконав студент групи КН-5Т
спеціальності 122 Комп'ютерні науки
Гришин Леонід Анатолійович

Керівник асистент
Штефан Наталія Зінов'ївна

Консультант д.т.н., доцент
Ведикодний Станіслав Сергійович

Рецензент к. ф.-м. н., доцент
Ткач Тетяна Борисівна

Одеса 2021

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ 5
ВСТУП 6
1 АНАЛІТИЧНА ЧАСТИНА 8
1.1 Опис предметної області 8
1.2 Характеристика об’єкту розробки 9
1.3 Огляд аналогів 9
1.3 Інструментальні засоби розробки 13
1.3.1 .NET Core – багатоплатформовий .NET 13
1.3.2 Entity Framework Core 5.0 16
1.3.3 MS SQL Server 18
1.3.5 Фреймворк Bootstrap та діалогові окна SweetAlert 20
1.3.6 Система керування версіями GitHub 21
1.3.7 Бібліотека jQuery 22
2 ПРОЕКТНА ЧАСТИНА 23
2.1 Особливості розробки проекту 23
2.2 Трьохрівнева архітектура веб-додатків 27
2.3 Моделювання системи UML-засобами 28
3 ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ 31
3.1 Проектування моделі даних 31
3.2 Опис інтерфейсу користувача 34
3.4 Опис кабінету адміністратору 41
3.5 Подальший розвиток проекту 45
ВИСНОВКИ 48
СПИСОК ВИКОРИСТОВАНИХ ДЖЕРЕЛ 49

ПЕРЕЛІК СКОРОЧЕНЬ

СКБД – Систем Керування Базами Даних

API – Application Programming Interface

CRM-системи – Customer Relationship Management

CSS 3 – Cascading Style Sheets

HTML – HyperText Markup Language

Sass – Syntactically Awesome Stylesheets

SEO – Search Engine Optimization

SQL – Structured Query Language

XHTML – Extensible Hypertext Markup Language

Bootstrap – фреймворк

CMS – система керування контентом

C# – мова програмування

Entity – сутність

Entity Framework Core – технологія для доступу до баз даних

jQuery – бібліотека JavaScript

GitHub – система керування версіями

GUI – графічний інтерфейс користувача

Mono – проект зі створення повноцінного втілення системи. NET

Primary Key – первинний ключ

ORM-інструмент – Object-Relational Mapping

SCM – Source Code Management

Xamarin – фреймворк для кросплатформної розробки мобільних додатків

.NET Framework – програмна платформа

ВСТУП

Технології стрімко змінюють світ. Трохи більше 20 років тому став загальнодоступним інтернет, а сьогодні людина вже не може уявити собі життя без смартфона, який цілодобово підключений до мережі. Змінюються і правила ведення бізнесу, продажів, реклами, логістики тощо. Покупці все частіше шукає інформацію онлайн.

Традиційний маркетинг йде в минуле. За нею «поспішають» і традиційні моделі продажів: на зміну e-commerce-сайтам і інтернет-магазинах приходять електронні торгові майданчики і прайс-агрегатори, які також відомі як маркетплейси.

Маркетплейс – платформа електронної комерції, онлайн-магазин електронної торгівлі, що надає інформацію про продукт або послугу третіх осіб, чиї операції обробляються його оператором. В цілому маркетплейс є оптимізовану онлайн-платформу з надання продуктів і послуг.

Більшість маркетплейсів заробляють на продажу своїх товарів та комісії з продажів інших продавців. Маркетплейси найчастіше беруть на себе просування товарів на своїй платформі, але іноді використовується гібридна система. Наприклад, Prom.ua пропонує своїм продавцям оплачувати просування товарів всередині майданчика [1].

Великі майданчики можуть собі дозволити виділяти величезні рекламні бюджети на контекстну рекламу, SEO, таргетовану рекламу і так далі. Тому нішеві магазини можуть використовувати їх потужності в свою користь.

Цільова аудиторія – це потенційні покупці, які можуть замовити товар і принести прибуток продавцям і маркетплейсу в цілому.

Так як маркетплейси практично завжди мають дуже широку цільову аудиторію, у диаломній роботі було прийнято рішення зупунитися лише та тематиці фантастичних серіалів компанії Marvell та інших топових кінострічок [2].

Загальні характеристики кваліфікаційної роботи:

− повний обсяг сторінок пояснювальної записки – 50;

− кількість рисунків – 28;

− кількість посилань – 15.

1 АНАЛІТИЧНА ЧАСТИНА

1.1 Опис предметної області

Покупка на платформах маркетплейсів набирає швидкі оберти. Особливо, якщо мова йде про придбання товару вузько-тематичного направлення.

Структури більшості маркетплейсов схожі. У них обов'язково повинен бути каталог товарів, форма для реєстрації продавців, особисті кабінети продавців і покупців, можливість подивитися статистику продажів, можливість вказати доступні способи оплати та доставки, а також форми для залишення відгуків і врегулювання суперечок.

Є ціла категорія клієнтів, які роблять покупки тільки на маркетплейсах. Вони довіряють репутації великого гравця і не бачать сенсу в серфінгу по невеликих інтернет-магазинах, якщо все необхідне можна знайти в одному місці. Це свідомо лояльні покупці, які готові відгукнутися на вашу пропозицію.

Комікси – мистецтво порівняно молоде: з настання «золотого століття» коміксів до наших днів минуло трохи більше 80 років. За цей час жанр встиг пережити злети, падіння і ренесанс всенародної любові.

Сьогодні комікси знову на піку популярності, не в останню чергу завдяки голлівудським екранізацій. А специфічний стиль малюнка і сторітеллінга як і раніше знаходить мільйони шанувальників.

Набирають популярність у прихильників коміксів і графічні альбоми з колекціями зображень за мотивами популярних анімаційних, художніх фільмів і комп'ютерних ігор.

Інтерес до коміксів в найближчі роки продовжить рости. Графічні романи вже стали частиною культурного життя в таких країнах, як Японія, Південна Корея, США, Бельгія, все більше шанувальників цього жанру з'являється і Росії, в тому числі і колекціонерів рідкісних випусків.

1.2 Характеристика об’єкту розробки

Об’єкт дипломної роботи – це маркетплейс на базі CRM-систем та управління даними. Планується розробити повний функціонал реального продажу тематичного товару з урахуванням панелі для адміністратору. Крім цього, під час тестування маркетплейсу його буде розміщено на безкоштовному хостінгі для виявлення можливих помилок у своїй роботі.

Для користувача необхідно реалізувати пошукову функцію за категоріями лотів (комікси, одяг, посуд, техніка та іграшки) та ціновий діапазон. Для кожного лоту необхідно сформувати картку товару з його характеристикою і додатковими фото.

Покупець повинен мати можливість додати товари до кошику або до категорії «Улюблені товари». Для покупки потрібна реєстрація користувача, яка буде допомагати розсилати повідомлення щодо розпродажів та нових товарів.

Для вибору інструментальних засобів розробки та можливих доповнень щодо вимог до об’єкту розробки необхідно зробити аналітичний огляд топових аналогів у мережі інтернет.

1.3 Огляд аналогів

Купівля та продаж коміксів стала масовою частиною колекційного хобі, аукціони коміксів слугують необхідною платформою для орієнтації у широкому світі коміксів. Часто можливості, де люди могли купувати та продавати комікси, були обмежені, але тепер, коли існує Інтернет, у простого колекціонера є величезне багатство місць, де можна отримати ідеальний предмет для своєї колекції.

Крім того, колекціонер, який хоче знайти рідку знахідку, може звернутися до відповідного аукціону коміксів, щоб швидко знайти охочого покупця.

Найпопулярнішою платформою з продажів вже довгий час займає «eBay» – це найбільший аукціонни1 сайт (рис. 1).

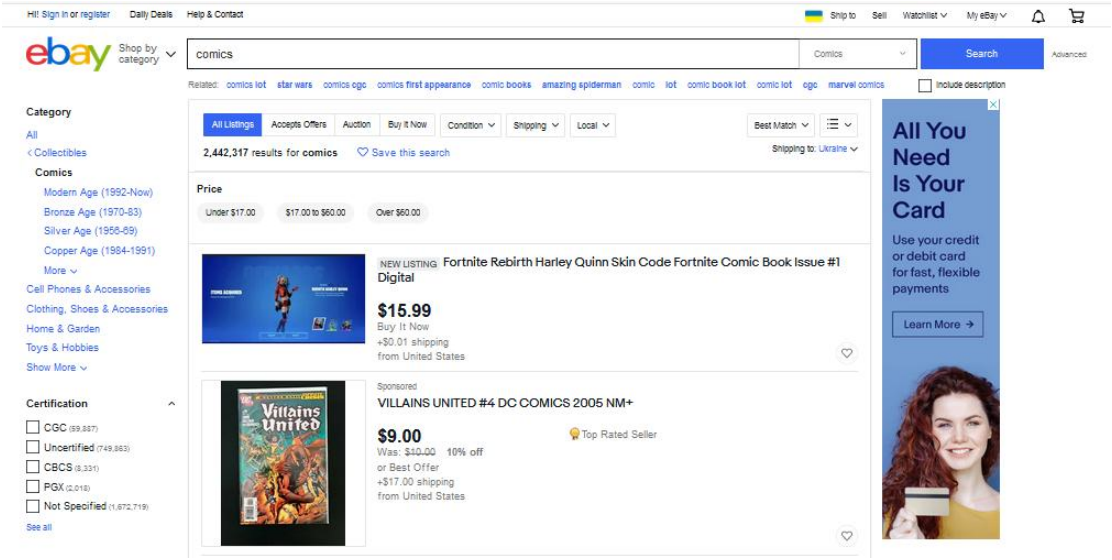

Рисунок 1 – Скріншот сторінки «eBay»

Вони надзвичайно вигідні та популярні. Їх рядок тегів: "Ви можете знайти його на eBay". І коли вони говорять це, вони мають на увазі це. Оскільки на вибір так багато аукціонів, користувач обов’язково знайдете ідеальний предмет для своєї колекції, а також зможете продати свій комікс із такою великою базою користувачів.

З недоліків для українського користувача залишається вартість доставки та податки даного ресурсу. Але розширена пошукова функція і неймовірна кількість продавців гарантує ідеальну покупку [3].

Наступним аналогом розглянуто маркетплейс «Comic Connect» (рис. 2) за посиланням (https://www.comicconnect.com/?tzf=1). Він зробив декілька великих вибухів на ринку аукціонів коміксів завдяки рекордним продажам коміксів, таких як «8,5 CGC Action Comics#1», який продався за 1,5 мільйона доларів.

Вони в основному продають комікси для срібла та золотого віку і мають на вибір десятки тисяч коміксів. Власники та оператори «Comic Connect»

також можуть похвалитися чудовою статистикою зі своїм генеральним директором Стівеном Фішером, який створив 10-бальну шкалу оцінок для коміксів та їх керівником, Вінсентом Зурцоло, заснувавши «Big Apple Comic-Con».

Це люди, які живуть і дихають коміксами, і їх сайт чудовий для пошуку заповітних коміксів.

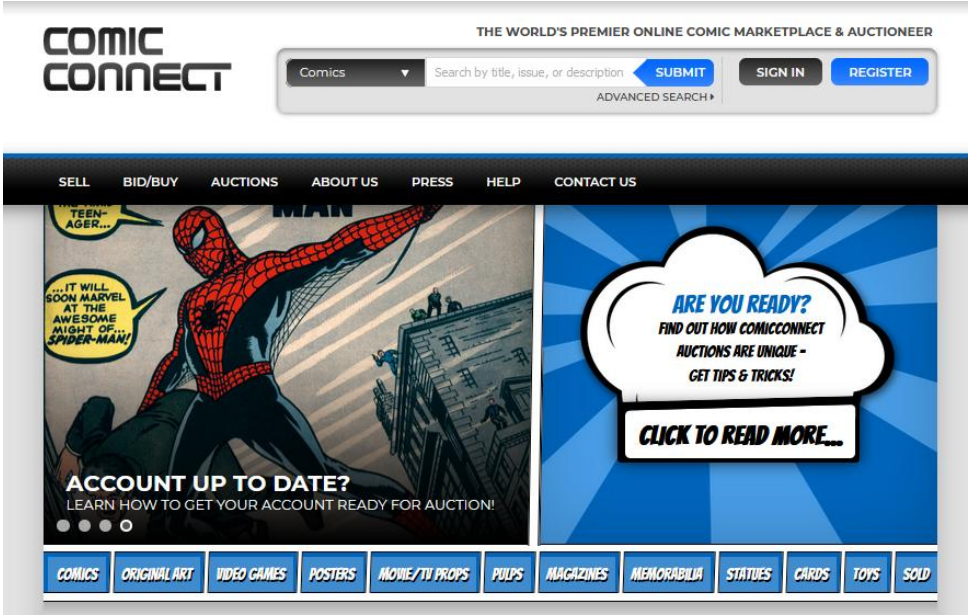

Рисунок 2 – Скріншот сторінки «Comic Connect»

У сайту вдалий, стилізований до тематики інтерфейс і дуже зручне налаштування пошукових функцій. Як і на попередньому аналогу, тут необхідна реєстрація користувача [3].

«COMIC QUERY» (https://comicquery.com) представляє собою дуже зручну пошукову систему, яка на 100% спрямована на те, щоб допомогти користувачу знайти комікси, які він хоче придбати (рис. 3). Це перша пошукова система, яка дозволяє шукати конкретну проблему та повертає результати для всіх магазинів у своїй базі даних, у яких ця проблема доступна для придбання.

Під час перегляду результатів пошуку просто натисніть список, про який ви хочете дізнатись більше, і «Comic Query» перенаправить вас на сто-

рінку товару на веб-сайті магазину, де ви можете його придбати. Наразі Comic Query сканує та індексує наявний інвентар понад 20 інтернет-магазинів коміксів (включаючи Сертифікований магазин коміксів) і продовжує постійно додавати до своєї бази даних більше магазинів та коміксів [3].

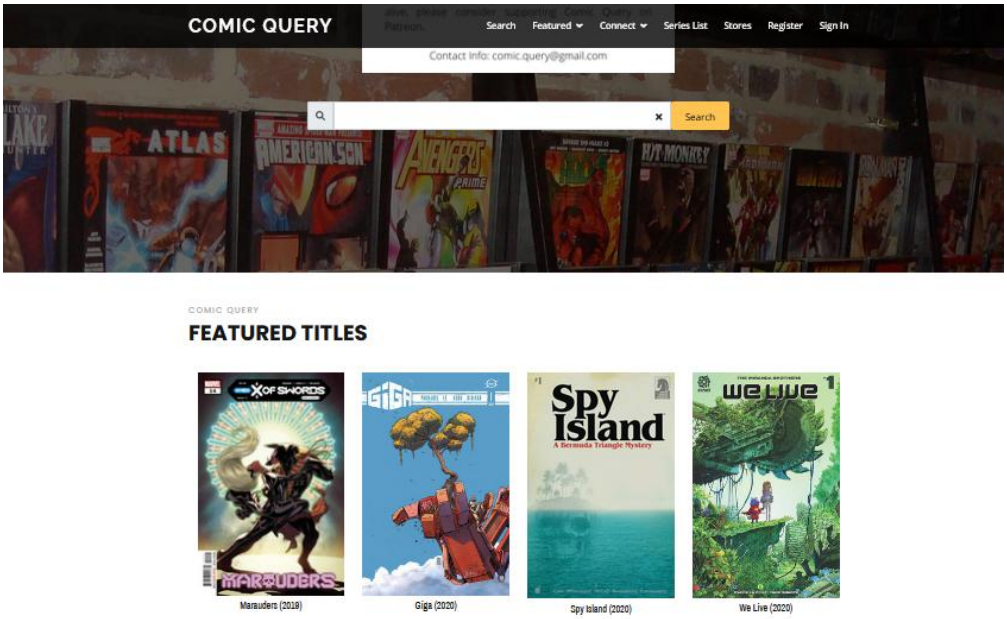

Рисунок 3 – Скріншот сторінки «Comic Query»

Серед популярних маркетплейсів топову позицію також займає «Facebook Marketplace» (рис. 4). Нажаль, користувач не знайдете величезної кількості списків для локального перегляду, завжди варто періодично проводити пошук, щоб побачити, які комікси пропонуються до продажу заданій країні. Для України такий ресурс не поширено.

Останній аналогом було розглянуто українську площадку з продажу коміксів «Bookstock» (https://bookstock.net). Кожен лот представлено описом та додатковими фото. Крім цього є можливість побачити контакт продавця і залишити відгук.

На даному маркетплейсі вбудована функція пошуку та онлайн сплати за замовлення. Ресурс «Bookstock» (рис. 5) на даний час максимально відповідає вимогам до проекту, тому його було прийнять як за приклад.

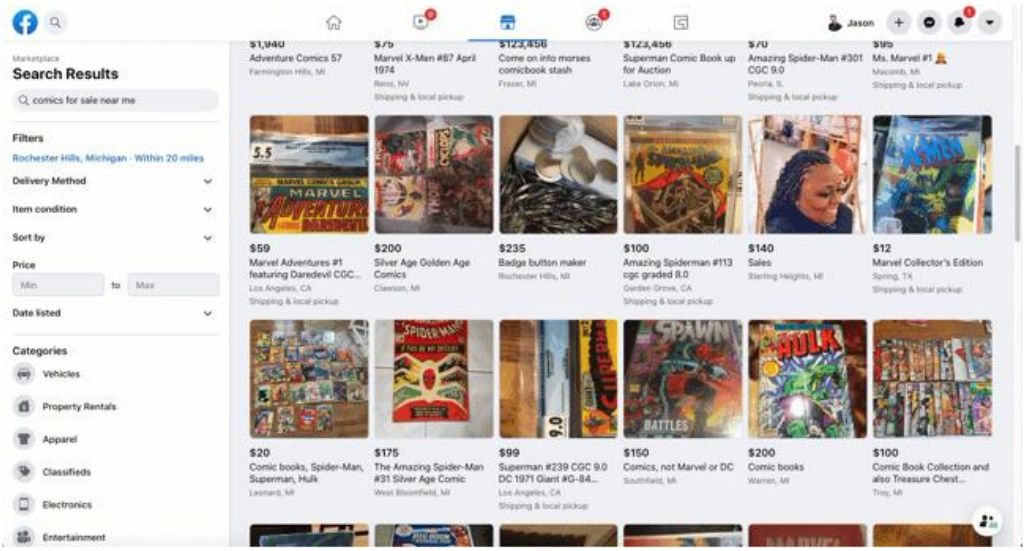

Рисунок 4 – Скріншот роботи пошукової функції у «Facebook Marketplace»

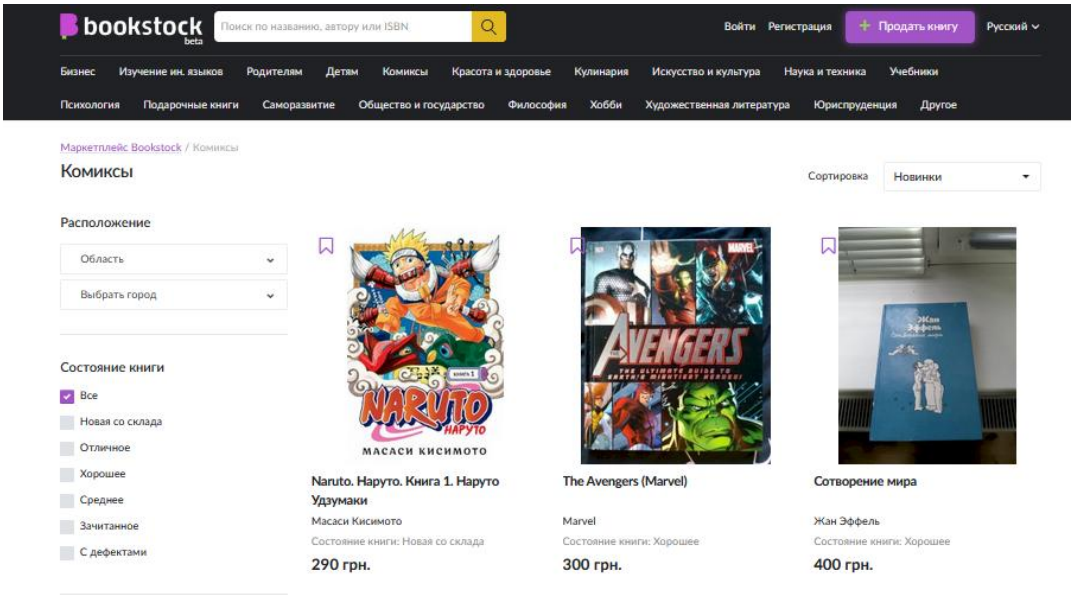

Рисунок 5 – Скріншот маркетплейсу «Bookstock»

1.3 Інструментальні засоби розробки

1.3.1 .NET Core – багатоплатформовий .NET

Версія Core – це модульна кроссплатформенная open source реалізація стандарту .NET загального призначення. Вона містить безліч API з .NET

Framework і включає середовище виконання, фреймворк, а також інструменти, які підтримують різні ОС і процесори.

Реалізація перебувала під управлінням напрацювань з ASP.NET Core, але вимагала більш сучасного підходу. Вона може використовуватися для різних пристроїв, в хмарі і для вбудованих IoT-сценаріїв.

До основних характеристик .NET Core слід віднести наступні:

1. Кросплатформеність: .NET Core надає ключову функціональність для реалізації потрібних особливостей вашого застосування, а також можливість повторного використання коду незалежно від вашої платформи. Вона підтримує Windows, Linux і macOS. Ви можете писати програми і бібліотеки, які запускаються на всіх підтримуваних ОС без змін;
2. Open source: .NET Core – один з багатьох проектів, який знаходиться під управлінням .NET Foundation і доступний на GitHub. Використання .NET Core як open source проекту сприяє прозорому процесу розробки і розвитку активного, залученого спільноти;
3. Модульність: .NET Core володіє модульностью. На відміну від однієї великої збірки з усією вбудованою функціональністю, .NET Core доступний у вигляді менших функціонально-орієнтованих пакетів. Це призводить до більш гнучкої моделі розробки, дозволяє оптимізувати додаток, включаючи тільки необхідні пакети NuGet.

Переваги меншого поля покриття додатки відображаються в підвищеної безпеки, менш вимогливому обслуговуванні, високої продуктивності і в знижених витратах за принципом «плати тільки за те, чим користуєшся» [4].

Нові .NET API, можливості виконуючого середовища і можливості мови як частини .NET 5 (рис. 6).

.NET 5 – наступний крок в .NET Core. Проект покликаний поліпшити .NET в декількох ключових аспектах:

- створити єдині виконуючу середу і фреймворк, які можна використовувати всюди, з однаковим поведінкою в runtime і досвідом розробки;

– розширити можливості .net за рахунок кращих напрацювань з .Net Core, .Net Framework, Xamarin і Mono;
– зібрати продукт з єдиної кодової бази, над якою розробники (з Microsoft і спільноти) можуть разом працювати і розширювати її, що дозволить поліпшити всі можливі сценарії.

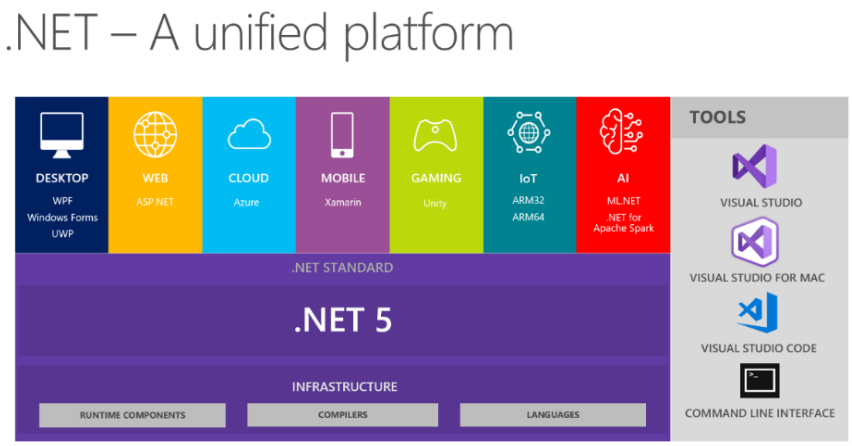

Рисунок 6 – Можливості.NET API

Цей новий проект і напрямок повністю змінять ситуацію з .NET. Завдяки .NET 5 ваш код і файли проектів будуть виглядати однаково, незалежно від типу створюваного додатка. З самого початку .NET спирався на JIT-компілятор для перетворення Intermediate Language коду в оптимізований машинний код. Ми створили кращу в галузі середу виконання з JIT, що володіє дуже високою продуктивністю, в той же час дозволяє розробникам писати код легко і швидко.

JIT-компілятори добре підходять для довго працюють хмар і клієнтських сценаріїв. Вони здатні генерувати код, що враховує особливості апаратної конфігурації, в тому числі специфічні процесорні інструкції. Також JIT може заново генерувати методи під час виконання, ця методика дозволяє компілювати з високою швидкістю, в той же час створюючи тонко налаштовану версію коду, якщо якісь методи використовуються часто [4].

1.3.2 Entity Framework Core 5.0

У більшості сучасних ASP NET Core додатків використовується Entity Framework Core. Entity Framework Core – це технологія для доступу до баз даних від Microsoft. Воно дозволяє взаємодіяти з СУБД (Система управління базами даних) за допомогою сутностей (entity), тобто класів і об'єктів NET, а не таблиць бази даних. Це найвідоміший і функціональний ORM-інструмент (Object-Relational Mapping) в C#.

ORM – це відображення даних на реальні об'єкти. Якщо традиційні засоби ADO.NET дозволяють створювати підключення, команди та інші об'єкти для взаємодії з базами даних, то Entity Framework являє собою більш високий рівень абстракції, який дозволяє абстрагуватися від самої бази даних і працювати з даними незалежно від типу сховища. Якщо на фізичному рівні йде оперування таблицями, індексами, первинними і зовнішніми ключами, але на концептуальному рівні, який пропонує Entity Framework, праця з об'єктами.

Наприклад, якщо розробник безпосередньо працює з базами даних, програміст повинен думати про підключення, підготовці SQL і параметрів SQL, як відправляти запити і про транзакції. А за допомогою Entity Framework Core все це робиться автоматично – розробник працює безпосередньо з класами NET [5].

Перший – Code First. Він має на увазі, що спочатку пишеться код на C#, а потім за цим кодом створюється база даних. Для цього підходу дуже важливо визначити класи моделі або entity, яка буде зберігатися в базі даних, описати її в класах C# у вигляді моделі, і написати клас контексту, який і буде працювати з використовуваної базою даних. Підхід Code First використовується найчастіше програмістами C#.

Другий підхід (Database-First) – підходить для тих, хто добре знає SQL, але в цьому випадку необов'язково добре знати C#. Насамперед створюється база даних, потім генерується EDMX-модель бази даних. У цьому XML у

файлі .edmx міститься інформація про структуру бази, модель даних і маппинг їх один на одного. У Visual Studio є графічний дизайнер, за допомогою якого можна працювати з .edmx

Model-First – це третій підхід ORM. Його часто використовують архітектори, так як при цьому підході можна не знати ні SQL, ні синтаксис C#. У цьому випадку спочатку створюється графічна модель EDMX, в цей час в фоновому режимі створюються класи C# моделі, а потім генерується база даних на основі діаграми EDMX. Всі таблиці бази даних визначаються в Entity Framework у вигляді класів моделей або сутностей (entity), як правило, за принципом 1 таблиця, наприклад users, – 1 клас в NET, наприклад, User. Такі пари називають умовностями, і вони визначені в класі контексту даних як набори DbSet і такий підхід працює за замовчуванням [6].

Центральної концепцією Entity Framework є поняття сутності або entity. Сутність представляє набір даних, асоційованих з певним об'єктом. Тому дана технологія передбачає роботу не з таблицями, а з об'єктами і їх наборами.

Будь-яка сутність, як і будь-який об'єкт з реального світу, має низку властивостей. Наприклад, якщо сутність описує людини, то ми можемо виділити такі властивості, як ім'я, прізвище, зріст, вік, вага. Властивості необов'язково представляють прості дані типу int, а й можуть представляти більш комплексні структури даних. І у кожної сутності може бути одна або кілька властивостей, які будуть відрізняти цю сутність від інших і будуть унікально визначати цю сутність. Подібні властивості називають ключами.

При цьому суті можуть бути пов'язані асоціативної зв'язком один-ко-многим, один-ко-одному і багато-до-багатьох, подібно до того, як в реальній базі даних відбувається зв'язок через зовнішні ключі. Відмінною рисою Entity Framework є використання запитів LINQ для вибірки даних з БД.

Іншим ключовим поняттям є Entity Data Model. Ця модель зіставляє класи сутностей з реальними таблицями в БД. Entity Data Model складається з трьох рівнів: концептуального, рівень сховища і рівень зіставлення (маппін-

га). На концептуальному рівні відбувається визначення класів сутностей, які використовуються в додатку [5].

Рівень сховища визначає таблиці, стовпці, відносини між таблицями і типи даних, з якими порівнюється використовувана база даних. Рівень зіставлення (маппінга) служить посередником між попередніми двома, визначаючи зіставлення між властивостями класу суті і стовпцями таблиць [7].

1.3.3 MS SQL Server

SQL Server є однією з найбільш популярних систем керування базами даних (СКБД) в світі. Дана СКБД підходить для самих різних проектів: від невеликих додатків до великих високонавантажених проектів. SQL Server довгий час був винятково системою управління базами даних для Windows, проте починаючи з версії 16 ця система доступна і на Linux. SQL Server характеризується такими особливостями як:

- продуктивність. SQL Server працює дуже швидко;
- надійність і безпека. SQL Server надає шифрування даних;
- простота. з даної СКБД відносно легко працювати і вести адміністрування.

Центральним аспектом в MS SQL Server, як і в будь-який СКБД, є база даних. База даних являє сховище даних, організованих певним способом. Нерідко фізично база даних представляє файл на жорсткому диску, хоча таке відповідність необов'язково. Для організації баз даних MS SQL Server використовує реляційну модель, яка передбачає зберігання даних у вигляді таблиць, кожна з яких складається з рядків і стовпців. Кожен рядок зберігає окремий об'єкт, а в стовпчиках розміщуються атрибути цього об'єкта.

Для ідентифікації кожного рядка в рамках таблиці застосовується первинний ключ (primary key). В якості первинного ключа може виступати один або декілька стовпців. Використовуючи первинний ключ, ми можемо поси-

латися на певну рядок в таблиці. Відповідно два рядки не можуть мати один і той же первинний ключ.

Через ключі одна таблиця може бути пов'язана з іншого, тобто між двома таблицями можуть бути організовані зв'язку. А сама таблиця може бути представлена у вигляді відносини ("relation"). Для взаємодії з базою даних застосовується мова SQL (Structured Query Language).

Клієнт (наприклад, зовнішня програма) відправляє запит на мові SQL за допомогою спеціального API. СКБД належним чином інтерпретує і виконує запит, а потім посилає клієнту результат виконання [8].

1.3.4 HTML5 і CSS3

HTML (HyperText Markup Language) представляє мову розмітки гіпертексту. Фактично з додаванням нових функцій HTML5 став не просто новою версією мови розмітки для створення веб-сторінок, але і фактично платформою для створення додатків, а область його використання вийшла далеко за межі веб-середовища інтернет: HTML5 застосовується також для створення мобільних додатків під Android, iOS, Windows Mobile і навіть для створення десктопних додатків для звичайних комп'ютерів

HTML 5 як потужна платформа для створення веб-додатків, яка включає не тільки безпосередньо мову розмітки гіпертексту, оновлений HTML, але і мова програмування JavaScript і каскадні таблиці стилів CSS 3 (Cascading Style Sheets). Каскадні таблиці стилів, які містять опис особливостей зовнішнього вигляду веб-документа, створеного на основі інструментів гіпертекстової розмітки. В основному CSS використовується для візуального оформлення веб-сторінок. Тобто, HTML задає структуру ресурсу, а таблиці стилів визначають то, як він буде виглядати [9].

Саме CSS відповідають за привабливість і дизайн сторінок. Ця технологія дозволяє «навести красу». Уже судячи з назви вона працює зі стилями (квітами, шрифтами, формами). CSS може застосовуватися не тільки в мовах

розмітки HTML і XHTML, але і працювати в XML-форматі (включаючи документи SVG і HUL).

CSS3 – третя версія таблиці каскадних стилів, яка значно розширила можливості попередніх поколінь. Особливість CSS3 полягає в тому, що з її допомогою можна створювати анімовані елементи без допомоги JS. Вона включає підтримку різних градієнтів і тіней, використовує нові форми згладжування [10].

Використання набору технологій HTML5 + CSS3 відкриває розробникам величезний арсенал програмних засобів, з якими значно легше стало створювати зручні, стильні, сучасні і функціональні веб-ресурси.

1.3.5 Фреймворк Bootstrap та діалогові окна SweetAlert

Це найпопулярніший в світі фреймворк для створення швидкодіючих мобільних сайтів з BootstrapCDN і шаблонами стартової сторінки. Швидке створення і настройка адаптивних мобільних сайтів за допомогою Bootstrap, що включає змінні Sass і міксини, адаптивну систему сіток і безліч готових компонентів і плагінів JavaScript.

Bootstrap має свою власну бібліотеку значків SVG з відкритим вихідним кодом, розроблену для кращої роботи з нашими компонентами і документацією. Значки Bootstrap призначені для кращої роботи з компонентами Bootstrap, також вони будуть прекрасно працювати в будь-якому проекті. Це SVG-файли, тому вони швидко і легко масштабуються, можуть бути реалізовані кількома способами і можуть бути стилізовані за допомогою CSS [11].

SweetAlert – добротне рішення, що дозволяє стильно показувати повідомлення користувачам в модальних вікнах. По суті, це елегантна заміну стандартним методом «Alert». Стандартне оформлення вікон виконано з урахуванням популярного нині тренда на плоскі інтерфейси. Все естетично і в більшості випадків цілком реально обійтися оформленням з коробки. Тим більше, всі вікна ще містять невелику анімацію (CSS3), яка підвищує наочність

повідомлень в залежності від їх типу. Вікна, створювані SweetAlert, адаптивні, тому на мобільних пристроях вони відображаються також прекрасно, як в десктопних браузерах.

Мікросервісом, як вище було сказано, виступає безпосередньо сайт, що напряму контактує з користувачем. Для створення даного сайту була задіяна велика кількість програмного забезпечення та сервісів. Основними з яких є система керування версіями GitHub, MS SQL, Visual Studio, SSMS та Visual Studio Code.

1.3.6 Система керування версіями GitHub

Система керування версіями (СКВ, англ. source code management, SCM) – програмний інструмент для керування версіями одиниці інформації: вихідного коду програми, скрипту, веб-сторінки, веб-сайту, 3D-моделі, текстового документу тощо.

СКВ зазвичай використовуються при розробці програмного забезпечення для відстеження, документування та контролю над поступовими змінами в електронних документах: у сирцевому коді застосунків, кресленнях, електронних моделях та інших документах, над змінами яких одночасно працюють декілька людей.

Кожна версія позначається унікальною цифрою чи літерою, зміни документу занотовуються. Зазвичай також зберігаються дані про автора зробленої зміни та її час. Інструменти для контролю версій входять до складу багатьох інтегрованих середовищ розробки.

Більшість систем контролю версії дозволяють кожному з об’єктів присвоювати теги, за допомогою яких можна формувати нові гілки та репозиторії. Використання системи контролю версії є необхідним для роботи над великими проектами, над якими одночасно працює велика кількість розробників.

Системи контролю версії надають ряд додаткових можливостей:

- можливість створення різних варіантів одного документу;
- документування всіх змін;
- коли ким було додано, хто який рядок змінив;
- реалізує функцію контролю доступу користувачів до файлів. Є можливість його обмеження;
- дозволяє створювати документацію проекту з поетапним записом змін в залежності від версії;
- дозволяє давати пояснення до змін та документувати їх.

1.3.7 Бібліотека jQuery

jQuery – бібліотека JavaScript, що містить в собі готові функції мови JavaScript, всі операції jQuery виконуються з коду JavaScript.

Синтаксис jQuery розроблений, щоб зробити орієнтування у навігації зручнішим завдяки вибору елементів DOM, створенню анімації, обробки подій, і розробки AJAX-застосунків. jQuery також надає можливості для розробників, для створення плагінів у верхній частині бібліотеки JavaScript. Використовуючи ці об’єкти, розробники можуть створювати абстракції для низькорівневої взаємодії та створювати анімацію для ефектів високого рівня. Це сприяє створенню потужних і динамічних веб-сторінок.

Бібліотека jQuery проводить маніпуляції з html-елементами, керуючи їх поведінкою і використовуючи DOM для зміни структури веб-сторінки. При цьому вихідні файли HTML і CSS не змінюються, зміни вносяться лише в відображення сторінки для користувача.

Бібліотека jQuery є JavaScript файлом, яка включає всю його DOM, події (events), ефекти (effects), і Ajax функції. Вона може бути додана до web-сторінки посиланням на локальну копію, або на одну з копій доступних на публічному сервері (наприклад Google або Microsoft CDN).

2 ПРОЕКТНА ЧАСТИНА

2.1 Особливості розробки проекту

Основною метою розробки даного проекту було створення самописного двигуна CMS для роботи бази даних з мікросервісами. CMS – це система керування контентом, набір скриптів для створення, редагування і управління контентом сайту. Прикладами CMS є WordPress, Joomla, OpenCart.

Якщо раніше більшість сайтів були статичними і вимагали внесення правок в їх вміст вручну, зараз динаміка розвитку проектів вимагає готовності швидко реагувати на зміни і впроваджувати їх з максимальною оперативністю. При цьому не усі користувачі хочуть або можуть собі дозволити звертатися до розробників, особливо якщо сайт вимагає постійної роботи над ним.

У свою чергу, системи керування контентом дозволяють користувачам, які не володіють навичками розробки сайтів і знаннями мов програмування, самостійно працювати над створенням і зміною сайту.

Тобто, в даному проекті системою CMS є панель адміністратора для додавання, видалення, редагування інформації, яка на кінцевому етапі буде відображатися користувачеві. Мікросервісом виступає безпосередньо сайт, що надає необхідну інформацію. В даному випадку, предметною областю є управління контентом для інтернет магазину.

Важливо зазначити, що CMS роблять набагато більше, ніж допомагають керувати текстовим та графічним вмістом, що відображається на веб-сторінках. Вони розвинулися, щоб допомогти розробити вигляд веб-сайтів, відстежувати сеанси користувачів, обробляти пошуки, збирати коментарі відвідувачів, форуми хостів та багато іншого.

Як правило, CMS використовують для таких сайтів:

- блог, форум (WordPress, phpBB, vBulletin);
- інтернет магазин (Magento, OpenCart, osCommerce);
- соціальні мережі (InstantCMS, Social Engine);

- персональні сайти (WordPress, Monstra);
- корпоративні сайти (Joomla, Drupal);
- портали (DLE, Drupal).

Проте, більшість CMS можливо гнучко налаштовувати і використовувати для розробки сайтів різної спрямованості. Наприклад, найбільш популярним і універсальним варіантом є WordPress, на якому можливо створити практично будь-який проект: від особистого сайту до великого порталу або інтернет-магазину. Підсумуємо усі характеристики CMS-систем:

Переваги CMS:

- просто та зручно використовувати;
- багато функцій за рахунок доповнень, шаблонів, розширень;
- сайт можна створити за короткий проміжок часу;
- наявність документації.

Недоліки CMS:

- не підходять для нетипових завдань;
- популярні CMS уразливі;
- необхідно стежити за оновленням CMS і сумісністю версій доповнень;
- підвищене споживання ресурсів, особливо при використанні плагінів.

Принцип роботи CMS полягає у схемі поділу наповнення сайту та його дизайну. Користувачеві надається можливість вибрати шаблон – заготовку, яка визначає оформлення сторінки. Залишається тільки заповнити його інформацією.

Більшість додатків мають певну форму функціональності CRUD. Насправді, кожному програмісту в якийсь момент доводилося мати справу з CRUD. Не кажучи вже про те, що програма CRUD – це форма, яка використовує форми для отримання та повернення даних із бази даних. CRUD також може описати конвенції користувальницького інтерфейсу, які дозволяють

переглядати, шукати та змінювати інформацію за допомогою комп'ютерних форм та звітів.

По суті, сутності читаються, створюються, оновлюються та видаляються. Ці самі об'єкти можуть бути змінені, взявши дані сервісу та змінивши властивості налаштування, перш ніж надсилати дані назад в службу для оновлення.

Парадигма CRUD є поширеною у створенні веб-додатків, оскільки вона забезпечує пам'ятну основу для нагадування розробникам про те, як створювати повноцінні, корисні моделі:

- процедури create: виконує оператор insert для створення нових записів;
- процедури read: читає записи таблиць на основі первинних ключів у вхідному параметрі;
- процедури update: виконує операцію update на таблиці на основі визначеного первинного ключа для запису в пункті where заяви;
- delete процедури: видаляє вказаний рядок у пункті where.

Акронім CRUD визначає всі основні функції, притаманні реляційним базам даних та програмам, що використовуються для управління ними, включаючи базу даних Oracle, Microsoft SQL Server, MySQL та інші.

Панель адміністрування (або адмін-панель) – це основний інструмент для роботи з будь-яким інтернет-магазином. За допомогою неї можна керувати продуктами та замовленнями, пропонувати знижки, приймати замовлення, спілкуватися зі своїми клієнтами, змінювати вигляд вашого магазину та робити багато іншого.

Панель адміністратора буде адаптивна до розміру екрана пристрою, з якого ви переглядаєте його. Таким чином, буде можливість керувати власним магазином з мобільних пристроїв.

Динамічна веб-сторінка (англ. dynamic web page) – веб-сторінка, вміст якої може змінюватись. В початковому варіанті гіпертекстова навігація від-

бувалася між «статичними» документами. Однак з часом до веб-сторінок було додано інтерактивності, і такі сторінки почали називати динамічними.

Наповнення (контент) такої веб-сторінки може замінюватися залежно від певних умов або дій. ASP.NET Core Razor Pages – це орієнтований на сторінки фреймворк для створення динамічних веб-сайтів, керованих даними, з чітким розділенням проблем.

На основі останньої версії ASP.NET від Microsoft – ASP.NET Core, Razor Pages підтримує розробку крос-платформ і може бути розгорнутий в операційних системах Windows, Unix та Mac.

Razor Pages – рекомендована основа для генерації HTML на стороні сервера на платформі .NET Core. Razor Pages використовує популярну мову програмування C# для програмування на сервері, а також простий у вивченні синтаксис шаблону Razor для вбудовування в розмітку HTML, щоб динамічно створювати вміст для браузерів.

Існує два шляхи для створення динамічних сторінок. По-перше, використання скриптів, що виконуються в браузері користувача (англ. client-side scripting) для зміни вмісту сторінки залежно від певних дій користувача. Для зміни не потрібно повного перезавантаження сторінки.

По-друге, використання програм, що виконуються на сервері (англ. server-side scripting) для зміни наповнення сторінки, що передається браузеру користувача. Інформація може змінюватися залежно від даних, відправлених у HTML формі, параметрів в URL, типу браузера, дати або часу доби та інших умов.

В даному випадку, для оптимізації запитів будуть використані обидва способи, бо для деяких ситуацій виконання запитів зі сторони клієнта буде більш ресурсовитратне ніж зі сторони сервера. Тому в цьому випадку буде використано змішаний вид будови динамічних сторінок.

2.2 Трьохрівнева архітектура веб-додатків

Трьохрівнева архітектура додатків – це модульна клієнт-серверна архітектура, яка складається з рівня уявлення, рівня додатки і рівня даних. Рівень даних забезпечує зберігання інформації, рівень додатків обробляє логіку, а рівень представлення являє собою графічний інтерфейс користувача (GUI), який взаємодіє з двома іншими рівнями. Ці три рівня є логічними, а не фізичними, і можуть працювати як на одному фізичному сервері, так і на різних машинах.

Рівень представлення. Цей рівень, створений з використанням HTML5, JavaScript і графічний інтерфейс користувача (GUI), розгортається на обчислювальному пристрої через веб-браузер або веб-додаток. Рівень представлення зв’язується з іншими рівнями за допомогою викликів інтерфейсу прикладних програм (API).

Рівень додатки, який також можна назвати логічним рівнем, написаний на мові програмування, такій як Java, Python або Ruby, і містить бізнес-логіку, яка підтримує основні функції програми. Базовий рівень додатків може бути розміщений на розподілених серверах в хмарі або на виділеному внутрішньому сервері, в залежності від того, скільки обчислювальної потужності потрібно з додатком (рис. 7).

Рівень даних складається з бази даних і програми для управління доступом для читання і запису в базі даних.

Цей рівень також може називатися рівнем зберігання і може бути розміщений локально або в хмарі. Популярні системи баз даних для управління доступом для читання або запису включають MySQL, Oracle, PostgreSQL, Microsoft SQL Server і MongoDB.

Переваги використання 3-рівневої архітектури включають поліпшену масштабованість, продуктивність і доступність. При використанні підходу до розробки додатків з трьома рівнями або частинами, всі ці частини можуть розроблятися одночасно декількома командами програмістів, які кодують

різними мовами, при цьому кожна з команд не залежить від інших розробників, які займаються створенням іншого рівня.

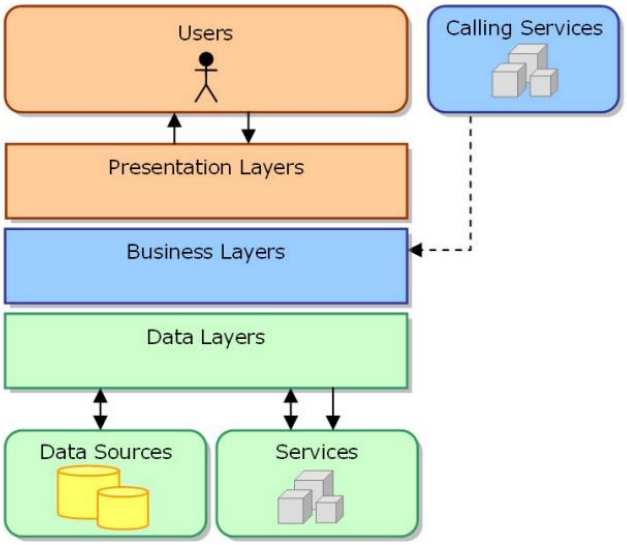

Рисунок 7 – Трирівнева архітектура додатків

Оскільки процес створення програмного коду для кожного рівня може зазнавати зміни, не зачіпаючи інші рівні, 3-рівнева модель полегшує безперервний розвиток програми для підприємства або програмного пакету в міру появи нових потреб і можливостей. Існуючі програми або критичні частини можуть бути постійно або тимчасово збережені і вміщені в новий рівень, компонентом якого вони стають.

2.3 Моделювання системи UML-засобами

UML (англ. Unified Modeling Language – уніфікована мова моделювання) – мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення, моделювання бізнес-процесів, системного проектування та відображення організаційних структур.

В процесі проектування система розглядається з різних точок зору за допомогою моделей, різні уявлення яких подаються у формі діаграм. Діагра-

ма – це графічне представлення безлічі елементів. Зазвичай зображується у вигляді графа з вершинами (сутностями) і ребрами (відносинами).

За допомогою діаграм можна візуалізувати систему з різних точок зору. Use Case Diagram – це найбільш загальне уявлення функціонального призначення системи. Даний процес діє як метод моделювання програмного забезпечення, що визначає функції, які повинні бути реалізовані, і вирішення будь-яких помилок, які можуть виникнути.

На діаграмі використання застосовуються два типу основних сутностей: варіанти використання і дійові особи , між якими встановлюються такі основні типи відносин:

- асоціація між дійовою особою і варіантом використання;
- узагальнення між діючими особами;
- узагальнення між варіантами використання;
- залежності (різних типів) між варіантами використання [12].

На рисунку 8 наведено приклад діаграми Use-Case для нашої системи:

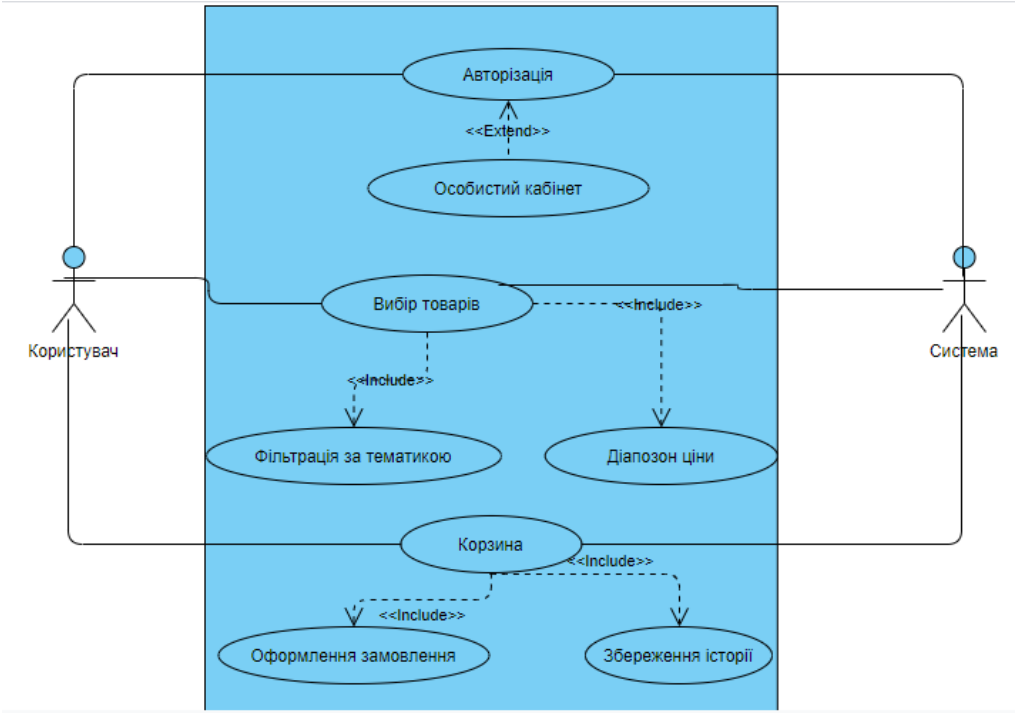

Рисунок 8 – Діаграма варіантів використання для користувача

Змоделюємо ще одну діаграму з точки зору адміністратору системи. Приклад представлено на рисунку 9:

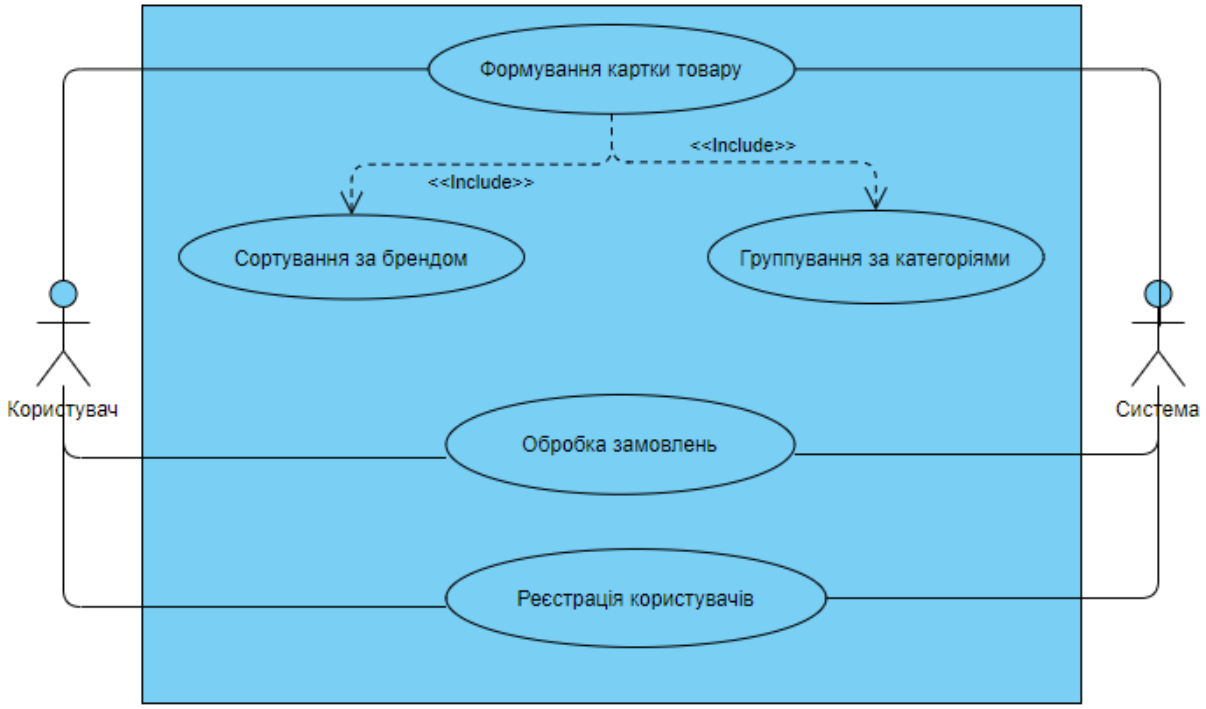

Рисунок 9 – Діаграма варіантів використання для адміністратору

3 ОПИС РЕАЛІЗАЦІЇ ПРОЕКТУ

3.1 Проектування моделі даних

Перед безпосередньою реалізацією проекту необхідно розробити макет моделі даних, який дозволить більш продуктивно та продумано розробляти систему споживання контента.

По-перше, визначимо список об’єктів системи:

– BaseEntity – базовий клас для всіх об’єктів системи;

– Product та його характерисити – набір об’єктів для визначення атрибутів товара, таких як: зображення, ціна, специфакції, тощо;

– Order та його характеристи – набір сутностей для збереження інформації щодо замовлень, типів оплати та статусів;

– Category – категорія в яку можна додати товар;

– Brand – бренд за яким товар закріпляється;

– User – користувач або адміністратор системи;

– Role – роль, яка визначає права користувача системи;

– SupportMessage – контактна форма для збереження запитів;

– Cart – віртуальна корзина, яка довзоляє створювати замовлення.

Нижче зазначимо перелік основних функцій цього інтерфейса, спроектованого в системі. Назви методів:

– FindAll;

– FindById;

– Find;

– Insert;

– Update;

– Delete;

– Restore.

Як бачимо з наведеного інтерфейса маємо розширену імплементацію, яка додатково к чотирьом базовим функціям дає можливість, відновлювати

видалені записи, знаходити об’єкти за ідентифікатором або за іншими ознаками.

Необхідно визначити алгоритм роботи кожного з них, а також відтворити сигнатури цих методів для подальшого розгляду.

```
BaseDao<T> BaseDao(string, System.Data.IDbConnection)
BaseDao<T> Count(bool)
BaseDao<T> Delete(int)
BaseDao<T> Delete(T)
BaseDao<T> DeleteOrRestore(T)
BaseDao<T> Find(System.Func<T, bool>, bool)
BaseDao<T> FindAll(bool)
BaseDao<T> FindById(int?)
BaseDao<T> FirstOrDefault(System.Func<T, bool>, bool)
BaseDao<T> FirstOrDefaultRandom(bool)
BaseDao<T> GetEnumerator()
BaseDao<T> Insert(T)
BaseDao<T> Merge(T)
BaseDao<T> Query(string)
BaseDao<T> QueryFirstOrDefault(string)
BaseDao<T> Random(bool)
BaseDao<T> Random(int, bool)
BaseDao<T> Restore(int)
BaseDao<T> Restore(T)
BaseDao<T> IEnumerable<T> GetEnumerator()
BaseDao<T> Take(int, int, bool)
BaseDao<T> Update(T)
BaseDao<T> SelectFromString
BaseDao<T> TableName
```

Основним методом пошуку є Find():

```
public virtual IEnumerable<T> Find(Func<T, bool> predicate,
bool withDeleted = false) =>
Connection.Query<T>($"{SelectFromString}{(withDeleted ?
string.Empty : " where IsDeleted = 0")}").Where(predicate);
```

Для відтворення алгоритму створення нових елементів був реалізовано метод з однойменною назвою Insert().

Приклад даного методу розглянуто нижче.

```
public virtual int Insert(T item)
{
            var properties =
item.GetType().GetProperties(System.Reflection.BindingFlags.
Public | System.Reflection.BindingFlags.Instance | Sys-
tem.Reflection.BindingFlags.DeclaredOnly);
            var sql = $"insert into {TableName}
({string.Join(",", properties.Select(x => x.Name))}) values
({string.Join(",", properties.Select(x => "@" + x.Name))})
SELECT CAST(SCOPE_IDENTITY() as int)";
            return item.Id = Connection.Query<int>(sql,
item).Single();
}
```

Для оновлення запису потрібно буде використовувати функцію Update:

```
public virtual bool Update(T item)
{
   var properties =
item.GetType().GetProperties(System.Reflection.BindingFlags.
Public | System.Reflection.BindingFlags.Instance | Sys-
tem.Reflection.BindingFlags.DeclaredOnly);
   var sql = $"update {TableName} set {string.Join(",",
properties.Select(x => x.Name + " = @" + x.Name))} where Id
= @Id";
   return Connection.Execute(sql, item) > 0;
}
```

Також є декілько реалізацій видалення та відтворення об’єктів. Уся логіка реалізована на логічному видалені, що дозволяє в свою чергу уникати фізичного видалення, що дозволяє зберігати навіть помилково видалені об’єкти системи:

```
public virtual bool Delete(int id) =>
Connection.Execute($"update {TableName} set IsDeleted = 1,
DateModified = getdate() where Id = {id}") > 0;
public virtual bool Delete(T item)
{   item.IsDeleted = true;
    return Delete(item.Id);
 }
```

```
public virtual bool Restore(int id) =>
Connection.Execute($"update {TableName} set IsDeleted = 0,
DateModified = getdate() where Id = {id}") > 0;
public virtual bool Restore(T item)
{
    item.IsDeleted = false;
    return Restore(item.Id);
}
public virtual bool DeleteOrRestore(T item) =>
item.IsDeleted ? Restore(item) : Delete(item);
```

3.2 Опис інтерфейсу користувача

Безкоштовний хостинг – послуга (інтернет-сервіс), що дозволяє користувачеві безкоштовно розмістити веб-сайт або іншу інформацію користувача (текст, зображення, відео) в мережі Інтернет на сервері хостинг-провайдера. Для тестування маркетплейсу його розміщено на хостінгу за посиланням «http://flea-market.somee.com/» (рис. 10):

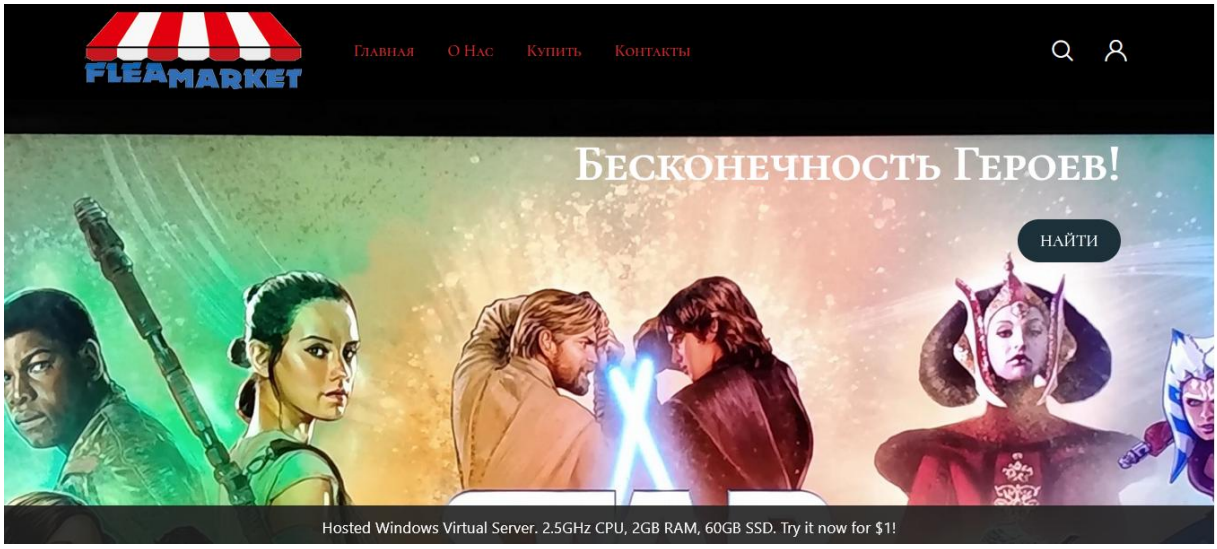

Рисунок 10 – Головна сторінка маркетплейсу «FleaMarket»

У разі якщо відвідувач маркетплейсу хоче замовити лот, йому потрібна авторизація чи регістрація (рис. 11) (якщо він робить покупку на цій площадці вперше).

Для входу під своїм акаунтом необхідно ввести логін та пароль, у разі проблем з останнім користувач може перейти за посиланням «Забыли пароль?» (рис. 12).

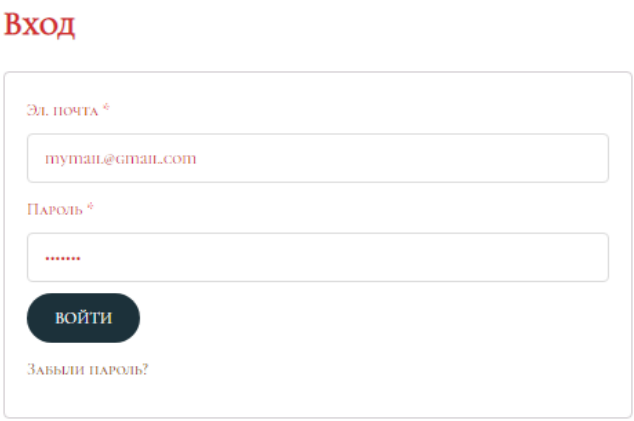

Рисунок 11 – Вхід до особистого кабінету клієнта «FleaMarket»

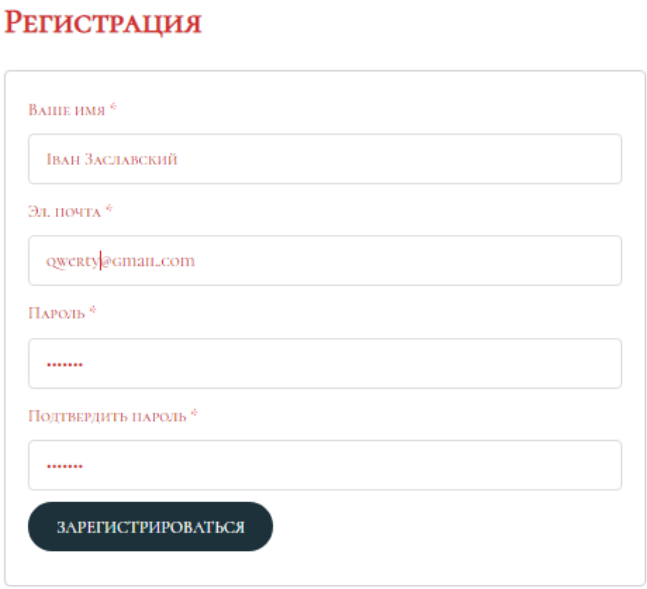

Рисунок 12 – Регістрація нового клієнта «FleaMarket»

При регістрації необхідно ввести свої дані та обов’язково пароль. Він повинен відповідати умовам системи, а саме бути не менше 8 символів. У разі некоректного паролю система висвічує відповідне повідомлення для кори-

стувача, як представлено на рисунку 13. Після успішної реєстрації користувачу надається доступ до персонального акаунту (рис. 14).

Рисунок 13 – Повідомлення про некоректний пароль

З панелі керування обліковим записом є можливість легко переглядати свої останні замовлення, управляти адресами доставки і виставлення рахунків, а також редагувати свій пароль і дані облікового запису.

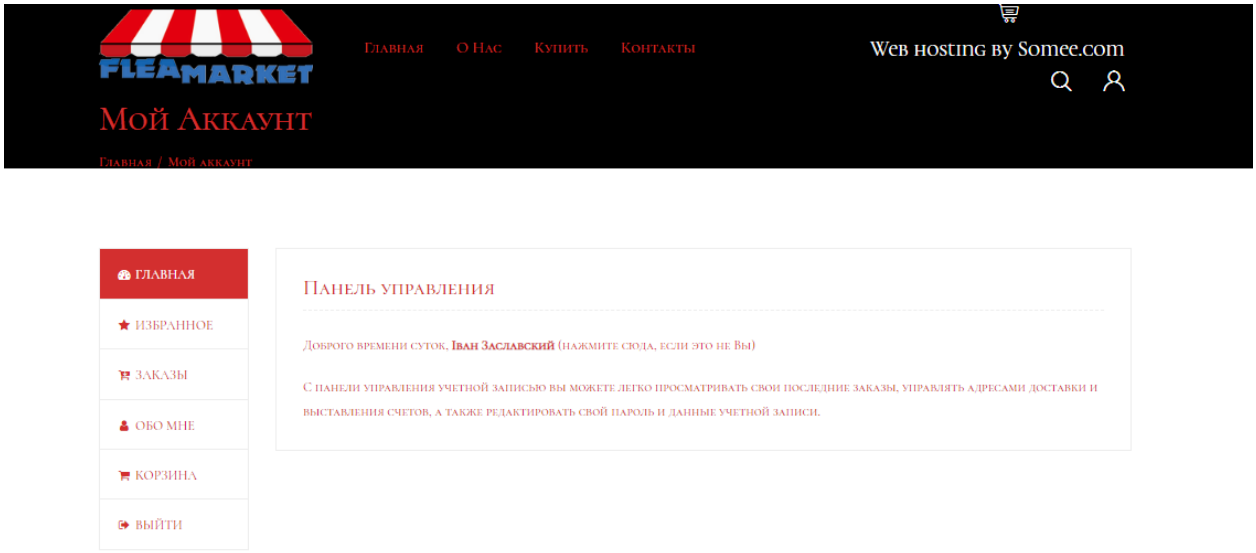

Рисунок 14 – Сторінка персонального аканту

На сторінці доступні наступні вкладки: «Главная», «Избранное», «Заказы», «Обо мне», «Карзина» та кнопка виходу з акаунту. Для заповнення кор-

зини необхідно обрати лоти для покупки. Так, якщо перейти до головної сторінки у користувача є можливість обрати категорії, за якими він хоче зробити замовлення, наприклад комікси та фігурки героїв.

Крім цього, є функція завдання цінового діапозону (рис. 15). Для пошукових критеріїв є можливість водночасно обирати декілька значень.

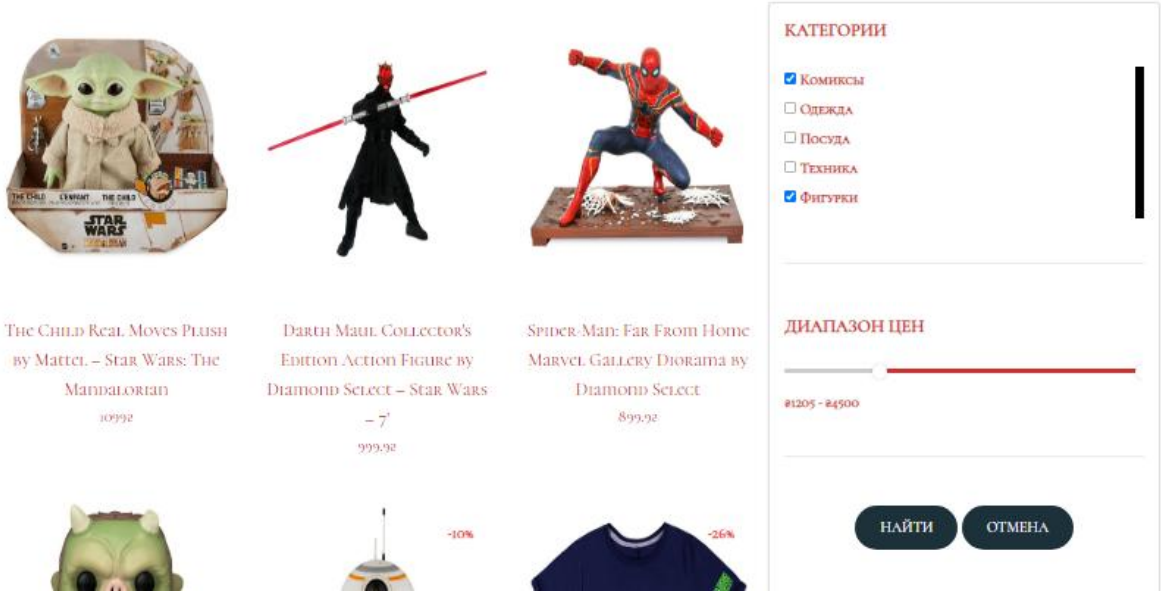

Рисунок 15 – Налаштування пошуку товару

Результат роботи пошукової функції представлено на рисунку 16.

Для кожного товару при наведені миші на його зображення з’являються дві іконки:

– для додавання до категорії улюблених;

– переміщення у корзину.

При наведені на іконку корзини у верхньому правому куті сайту з’являється перелік товарів, який користувач обрав до даної категорії (рисунку 17).

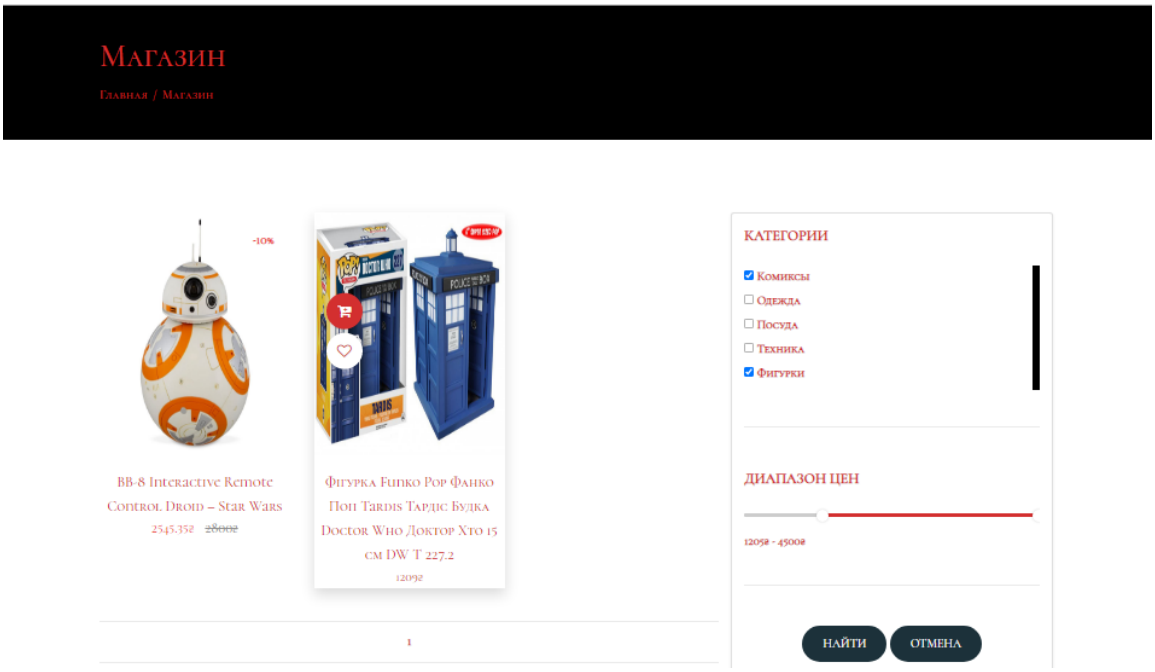

Рисунок 16 – Результати пошуку товару за параметрами

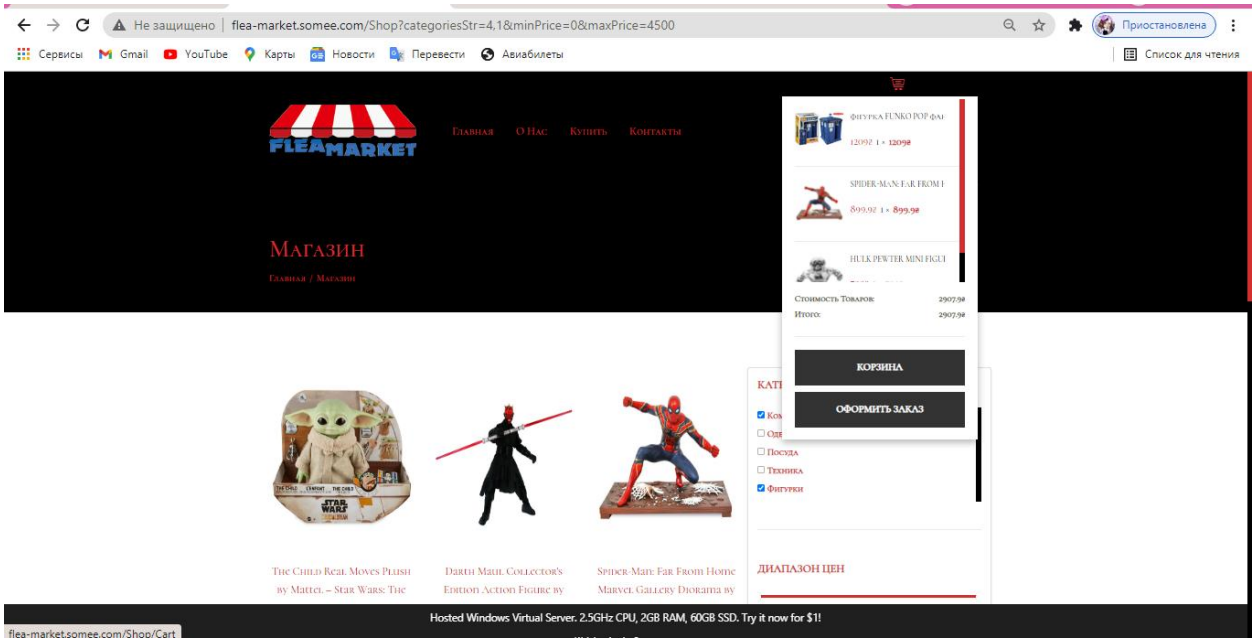

Рисунок 17 – Вікно відображення стану корзини

На даній вкладці є кнопки для переходу до самої корзини і для оформлення замовлення. Кошик замовлень в інтернет-магазині – це інтерфейс, куди користувач може додавати товари, які він збирається купити. Буває, що від-

відувач приходить в інтернет-магазин, але йде без покупок, часто кидаючи наповнену інтернет-кошик. Причина цього часто криється в незрозумілих її функціях і складнощі при оформленні.

Для того щоб відвідувач міг зробити замовлення через веб-кошик, її слід відображати звично для нього:

- розташування – в шапці сайту, ідеально – в правому верхньому куті;
- іконка – інтуїтивно зрозуміла, схожа на кошик або сумку;
- іконка не повинна бути схожа на банер – подібні зображення "фільтруються" поглядом користувача;
- у верхньому кутку інтернет-магазину поруч з іконкою відображається кількість товарів, які лежать в кошику, і їх загальна вартість;
- якщо в інтернет-кошика додано хоча б один товар – її значок змінюється на інший, який повідомляє, що в ній є товар.

Вкладка корзини представлена у вигляді таблиці, де перелічені всі обрані товари (назва і фото) та кількість-ціна для кожного лоту. Наприкінці можна побачити загальну суму замовлення (рис. 18).

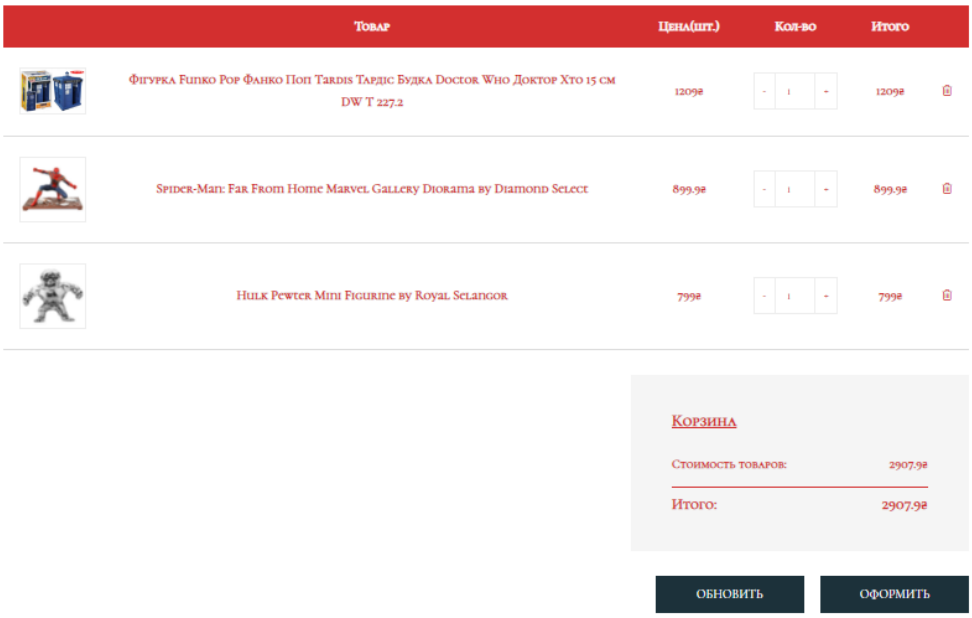

Рисунок 18 – Корзина покупця

До підтвердження своєї покупки користувач має можливість ввести поправки або додати ще декілька позицій товарів.

Для оформлення замовлення слід заповнити форму та обрати тип доставки та вид сплати за замовлення (рис. 19).

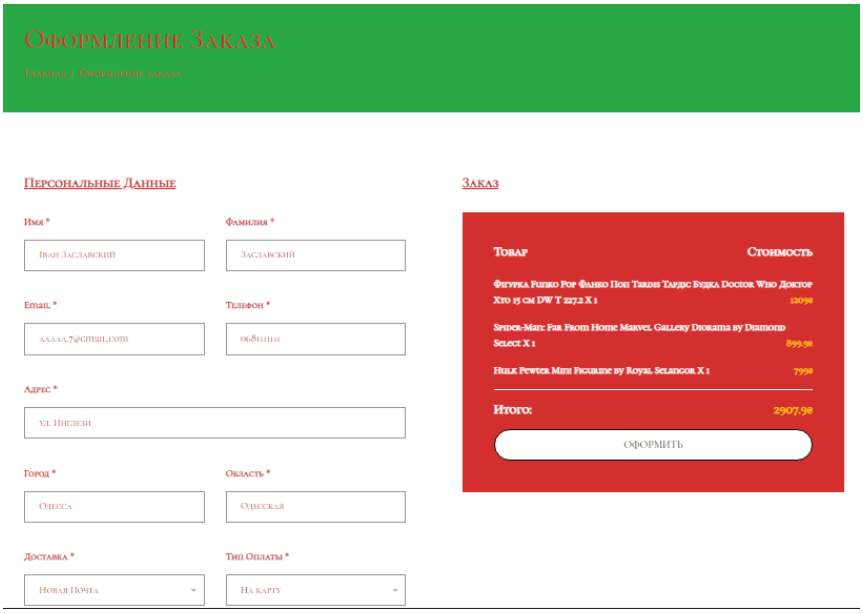

Рисунок 19 – Форма для оформления замовлення

Після підтвердження замовлення його дані автоматично передаються до вкладки «Заказы», як показано на рисунку 20:

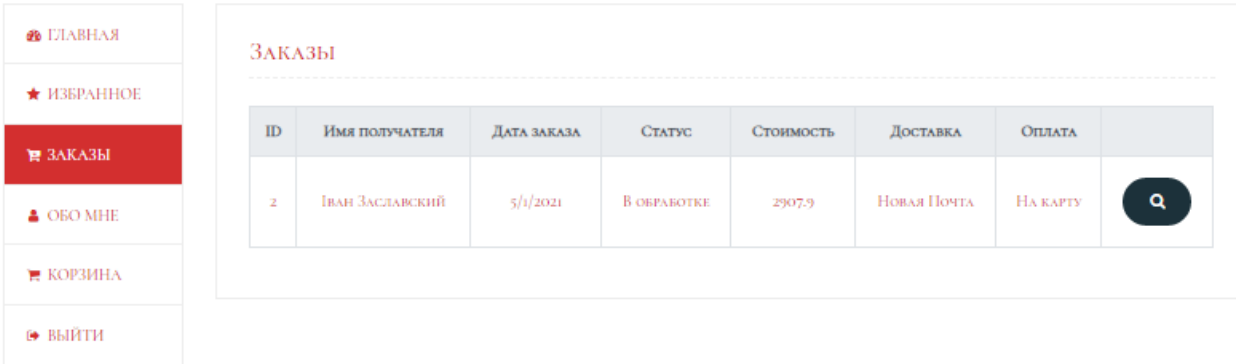

Рисунок 20 – Вкладка «Заказы»

3.4 Опис кабінету адміністратору

Розглянемо кабінет для адміністратору маркетплейса. Для авторизації необхідно ввести логін та пароль, як показано на рисунку 21. При підтвердженні авторизації система висвічує відповідне повідомлення (рис. 22). На наступному етапі йде завантаження кабінету для адміністратора.

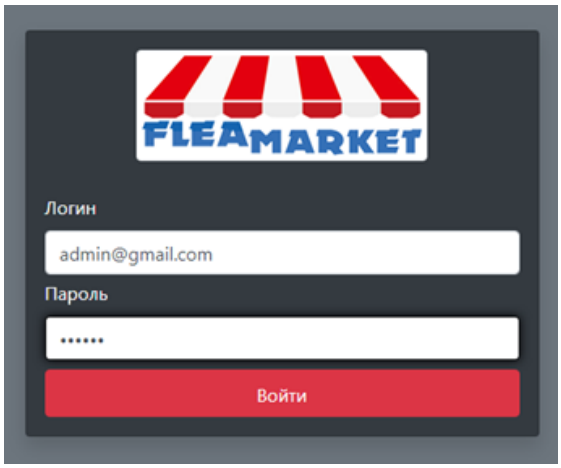

Рисунок 21 – Авторизація адміна

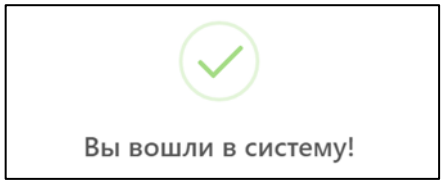

Рисунок 22 – Підтвердження авторизації

Весь функціонал розподілений за вкладкам:

– «Товари»;
– «Закази»;
– «Контакти»;
– «Вихід».

Приклад вкладок представлено на рисунку 23.

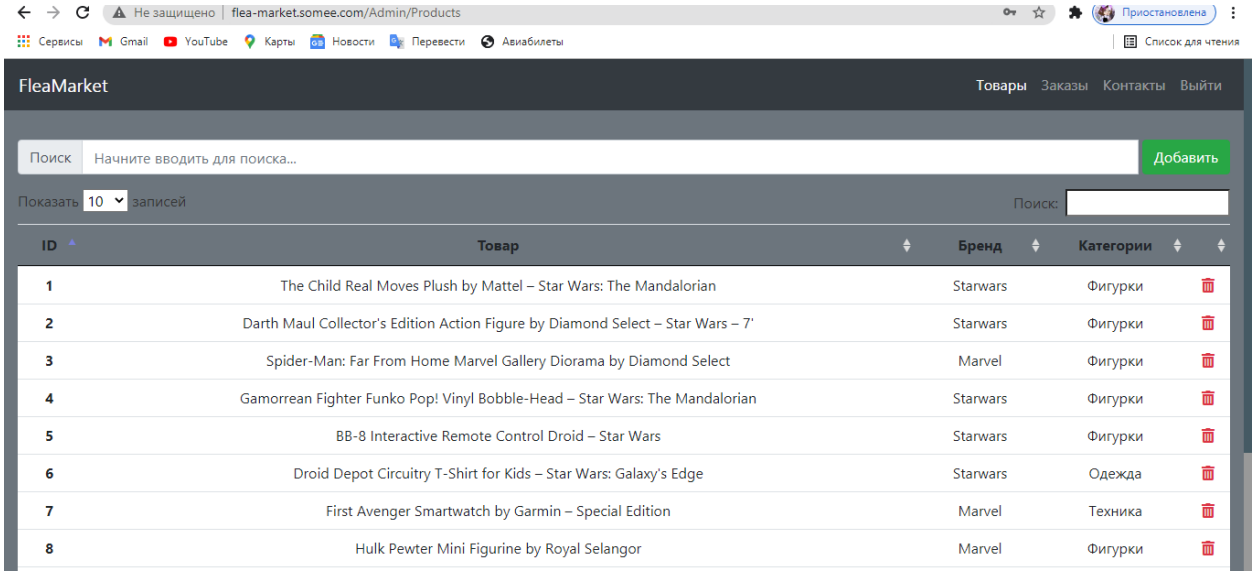

Рисунок 23 – Сторінка адміністратора «FleaMarket»

На головній вкладці «Товари» відображається зміст у вигляді таблиці по товарам маркетплейсу, а саме:

– товари;

– бренд;

– категорії товарів (фігурки, комікси, одяг, посуд та інше).

Адміністратор має можливість редагувати дані чи сортувати їх за абеткою або за полем «id». Розглянемо обробку інформації по товарам більш детально.

При натисканні на назву товару, відкривається картка для редагування інформації (рис. 24):

– назва (обов’язкове поле);

– опис лота (обов’язкове поле);

– бренд (обирається із списку);

– обов’язкове фото (при завантаженні автоматично йде налаштування розмирів);

– категорія лота («фигурки», «одежда», «посуда», «комиксы», «техника»), крім цього є можливість додавати нові категорії;

– специфікації, де можна обрати одну із перелічених чи додати нову (рис. 25):

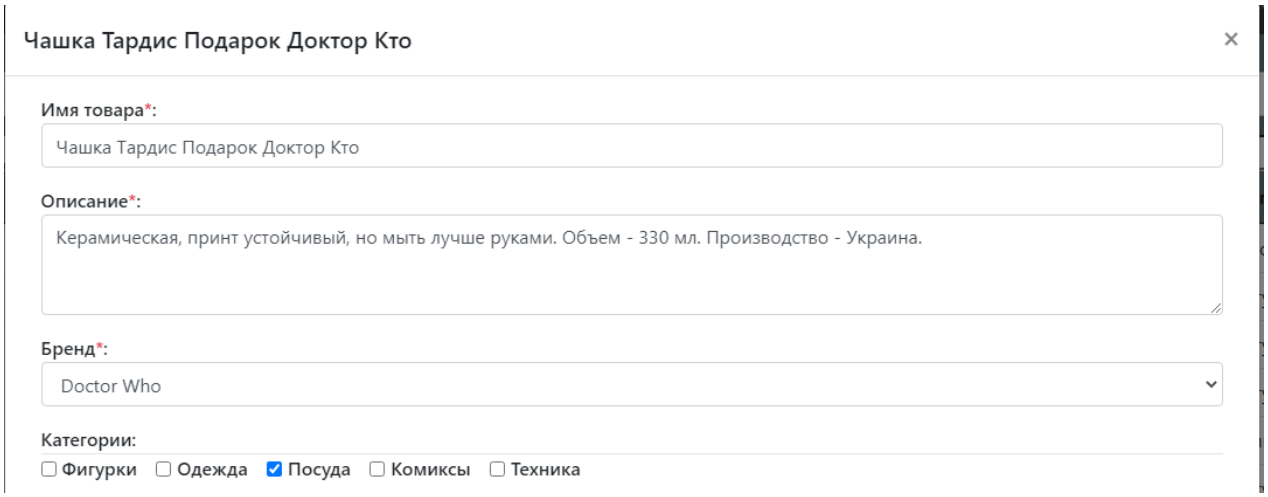

Рисунок 24 – Редагування картки товару

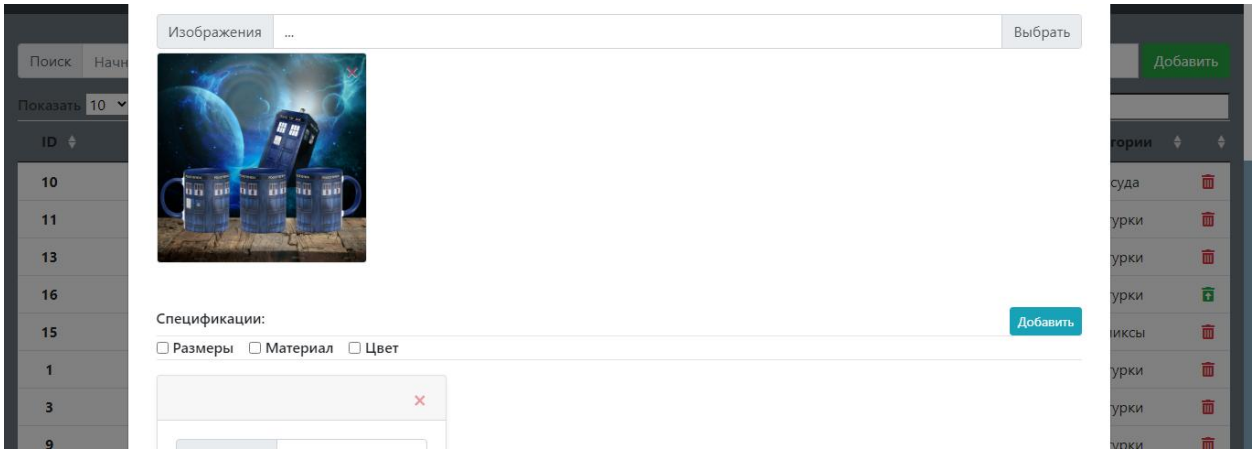

Рисунок 25 – Налаштування картки лоту

Поле «Специфікації» (рис. 26) необхідно для роботи пошукових запитів користувача. Так, наприклад є можливість вказати колір матеріалу, чи ціну за даний лот.

Якщо проходять розпродажі, то вказується як стара, так і нова ціна – це допомогає покупцю орієнтуватися у вигідних пропозиціях маркетплейсу.

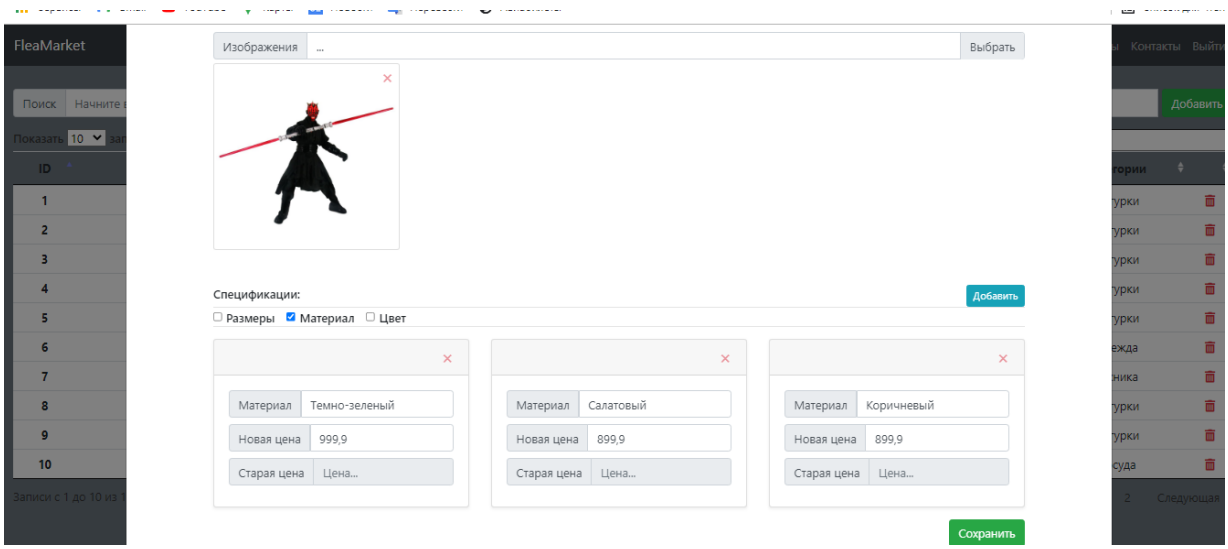

Рисунок 26 – Заповнення полів специфікацій лота

Для зручності відображення товарів у адміністратору є можливість обрати з списку кількість записів для однієї сторінки.

Пошукова функція допомагає знаходити інформацію за описом чи її деякою частиною (рис. 27).

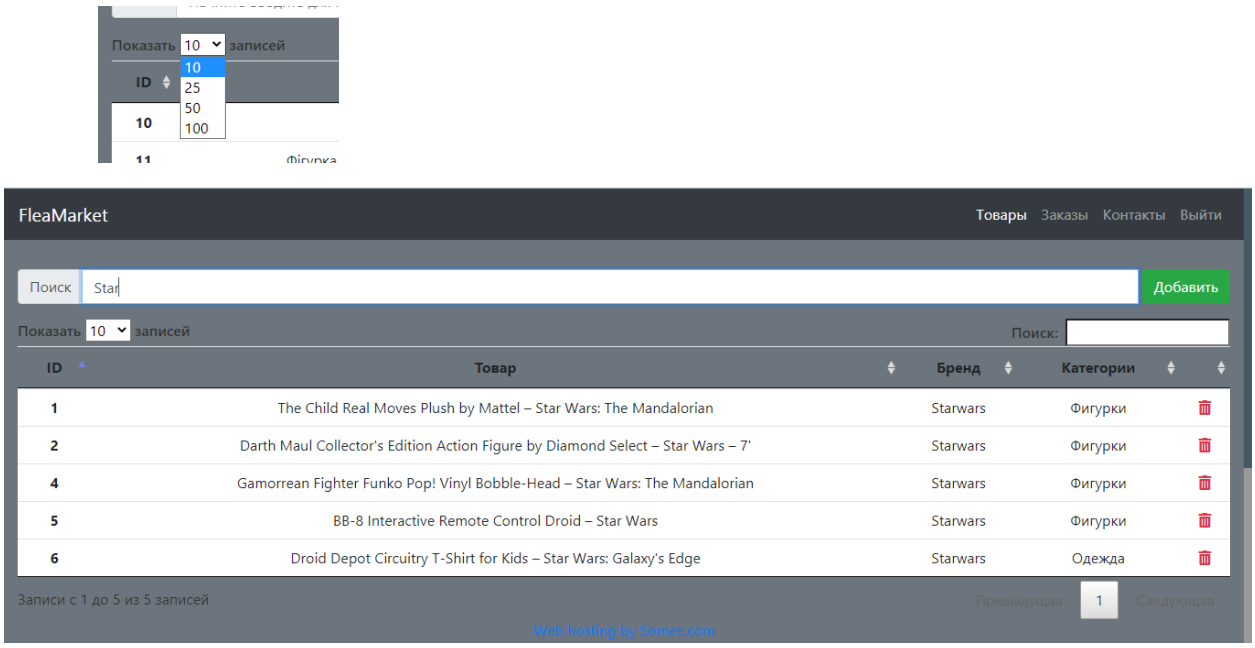

Рисунок 27 – Результат пошукової функції для адміністратору

Наприклад, при введені слова «Star» відображаються всі записи , в яких є така комбінація символів.

Наступна вкладка адміністратору – це «Замовлення». Тут представлено список замовлень з можливістю огляду деталей та функціями редагування інформації, якщо виникає на те потреба (рис. 28):

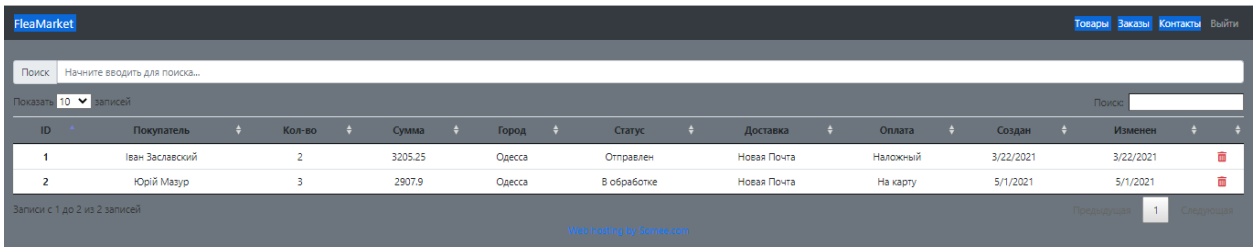

Рисунок 28 – Перелік замовлень маркетплейсу

3.5 Подальший розвиток проекту

Розглянемо просування бізнесу в інтернеті більш детально. Просування товарів в Інтернеті кардинально відрізняється від просування послуг. Причин тому проста: онлайн оплата і доставка товару. Тобто сучасні технології дають можливість організувати бізнес-процеси таким чином, що користувач не виходячи з браузера зможе придбати товар, без будь-яких контактом з офф-лайн-департаментами компанії. Для цього і створюються інтернет-магазини.

Це дозволяє досягти відразу кілька бізнес-цілей:

– оптимізація відділу продажів (в крайньому випадку заміна його колл-центром на аутсорсингу або скорочення до мінімуму – кількох людей для підтвердження замовлень, в залежності від їх кількості);
– додатковий канал продажів (для багатьох власників оффлайн-бізнесів, просування товарів в інтернеті досить швидко стає каналом продажів, що генерує 30-50% прибутку вже в перші два роки;
– диверсифікація джерел доходу;

– автоматизація бізнес-процесів (за допомогою налаштувань онлайн-оплати, підключення модулів доставок – можна автоматично передавати всю інформації у внутрішні інструменти обліку, наприклад, CRM /1C, що також досить спрощує процес обліку складу і процес покупки);
– програма лояльності, повернення клієнтів (в офлайні процес повторного просування товарів клієнту складний, в онлайні – досить створити особистий кабінет, забрати дані користувача і за допомогою інструментів інтернет-маркетингу повертати клієнтів за новими покупками пропонуючи знижки і бонуси, повідомляючи про нові акції);
– збір особистих даних (просувати товари в інтернеті – означає (якщо читати між рядків) збирати дані користувачів, що заходять на ваш сайт, залишають заявки і оформляють кошика з метою подальшого використання в цілях вашого бізнесу) [13].

Інтернет-магазин, це не зовсім "канал" просування в Інтернеті. Це кінцева точка для збору даних користувачів і акумуляція трафіку з Інтернету в одному місці, на майданчику належить власнику.

Практично у всіх сучасних інтернет-магазинах для просування товарів в Інтернеті використовуються внутрішні програми лояльності, реалізовані за допомогою особистих кабінетів і знижок. Наприклад якщо працює програма лояльності з кешбек. Частина суми замовлення повертається в особистий кабінет у вигляді бонусів, які потім можна використовувати для отримання знижки. Реалізувати такий функціонал на чужому майданчику в Україні, на даний момент – неможливо.

Розробивши власний інтернет-магазину, ви можете самостійно управляти знижками і акціями. Наприклад виставляти знижку за відсотком або ставити фіксовану, робити акції на певні товари або групи товарів, сортувати товари в каталозі по знижкам, розміром знижки або обсягу. Також, зараз дуже популярні "дисконтні купони" в обмін на контактні дані користувача або реєстрацію.

Плюси просування через маркетплейс: відсутність великих вкладень для "заходу" на маркетплейс. Можливість зареєструватися на великій кількості маркетплейсов для просування товарів в Інтернеті і отримувати замовлення за рахунок заниження ціни.

На маркетплейсах, як правило, величезна кількість товарів. Але всі вони +/- знаходяться в одному ціновому діапазоні. Як на ринку. Ті, хто ближче до виходу – дорожче, ті, хто в глибині – дешевше. Тільки замість "входу" на ринок – топ видачі по фільтру. Проте є найпопулярніша кнопка, на яку тиснуть понад 80% користувачів – сортування за зростанням ціни.

Відсутність необхідності в рекламному бюджеті і самостійному просуванні товарів в Інтернеті. Звичайно, можна дати рекламу всередині великого маркетплейса, проте це все одно буде значно дешевше, ніж працювати з google або facebook.

Мінуси просування через маркетплейс:

– відсутність можливості просувати бренд в повній мірі.
– ви не забирати дані користувачів і не збираєте аудиторію.
– ви прив'язані до форматів і інструментів власника маркетплейса.
– незручність в роботі з великим асортиментом.

Тут самим можна контролювати потік клієнтів і конкуруєте в рамках поточної аудиторії ресурсу. Це не завжди погано, але частіше – заважає розвиватися. Особливо якщо розміщуватися на невеликій кількості маркетплейсів [15].

Об’єкт дипломного проекту являє собою торгову площадку, де крім власного продажу є можливість додавати інших продавців з товаром заявленої тематики. Це збільшить аудиторії користувачів, кількість та асортимент товару а значить, що і прибутки для власника такої площадки.

Крім цього, слід пам’ятати і про розміщення рекламних банерів на бокових панелях сайту, які також роблять свій внесок у збільшення прибутків власника торгової точки в інтернеті.

ВИСНОВКИ

Покупці все частіше шукає інформацію онлайн. Рроль маркетплейсів швидко зростає. Як для продавців, так і для покупців пошут на одній торговій площадці більш зручний та вигідний. При великих покупках є можливість отримати бонуси від самого мартекплейсу, чи додаткові знижки у його магазинах.

Об’єкт дипломної роботи – маркетплейс за тематикою «Коміксі та кінострічки». Під час аналізу предметної області в ході роботи над дипломом були оглянуті існуючі аналоги, проаналізовані їх функціонал та саму структуру площадок. Все це допомогло поставити задачі та вимоги до дипломної роботи.

Нап етапі проектування за допомогою UML-засобів були смодульовані діаграми варіантів використання, які відображають основні функції і задачі у дипломному проекті.

Для реалізації функціоналу вибір став на створенний самописний двигуна CMS для роботи бази даних з мікросервісами. Інструментальними засоби розробки були обрані: модульна кроссплатформенная open source реалізація стандарту .NET, технологія для доступу до баз даних Framework Core, MS SQL Server, HTML5 і CSS3, бібліотека jQuery та фреймворк Bootstrap.

СПИСОК ВИКОРИСТОВАНИХ ДЖЕРЕЛ

1. Маркетплейс с нуля: как запустить торговую площадку. URL: https://e-pepper.ru/news/marketpleys-s-nulya-kak-zapustit-torgovuyu-ploshchadku.html. (дата звернення 08.04.2021)
2. 7 преимуществ создания собственного интернет-магазина. URL: https://horoshop.ua/blog/7-preimushestv-sozdanija-sobstvennogo-internet-magazina/. (дата звернення 10.04.2021)
3. Top Comic Book Auction Websites. URL: https://www.liveabout.com/top-comic-book-auction-sites-804239 (дата звернення 12.04.2021)
4. Представляем .NET 5. URL: https://temofeev.ru/info/articles-predstavlyaem-net-5/. (дата звернення 13.04.2021)
5. Entity Framework Core. Вступление. URL: https://habr.com/ru/company-otus/blog/500012/. (дата звернення 13.04.2021)
6. Новые возможности EF Core 5.0. URL: https://docs.microsoft.com/ru-ru/ef/core/what-is-new/ef-core-5.0/whatsnew. (дата звернення 13.04.2021)
7. Введение в Entity Framework. URL: https://metanit.com/sharp-entityframework/1.1.php. (дата звернення 20.04.2021)
8. Введение в MS SQL Server. URL: https://metanit.com/sql/sqlserver-1.1.php. (дата звернення 21.04.2021)
9. Руководство по HTML5 и CSS3. URL: https://metanit.com/web/html5/. (дата звернення 23.04.2021)
10. HTML5+CSS3. URL: https://astwellsoft.com/ru/blog-tehnology/html5-css3.html. (дата звернення 25.04.2021)
11. Создавайте быстрые, отзывчивые сайты с помощью Bootstrap. URL: https://getbootstrap.ru/docs/4.5/. (дата звернення 28.04.2021)

12. USE CASE-діаграма. приклади використання. URL: https://ukr.kagutech.com/4133730-use-case-diagram-examples-of-using. (дата звернення 28.04.2021)
13. Як просувати товари в інтернеті. URL: https://digiants.agency/uk/blog/kak-prodvigat-tovary-v-intertnete (дата звернення 02.05.2021)
14. Продвижение на маркетплейсах: как получить більше прибиль. URL: https://netology.ru/blog/09-2020-marketpleisy (дата звернення 02.05.2021)