Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Розробка web-сайту інтернет-магазину для продажу спортивного одягу на базі eCommerce платформи Magento 2

Виконав(ла): студент(ка) 6 курсу, групи СПм
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) Простяк В.М. (прізвище та ініціали)

Керівник (підпис) Цуприк Г.Б. (прізвище та ініціали)

Нормоконтроль (підпис) Стоянов Ю.М (прізвище та ініціали)

Завідувач кафедри (підпис) Петрик М.Р. (прізвище та ініціали)

Рецензент (підпис) Стадник Н.Б. (прізвище та ініціали)

Тернопіль
2022

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ _____
(підпис) (прізвище та ініціали)
« » 20___р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр
(назва освітнього ступеня)

за спеціальністю 121 інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Простяку Володимиру Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка web-сайту інтернет-магазину для продажу спортивного одягу на базі eCommerce платформи Magento 2

Керівник роботи Цуприк Галина Богданівна, доцент кафедри програмної інженерії
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «___»_____20____року № _____
2. Термін подання студентом завершеної роботи _____
3. Вихідні дані до роботи вимоги до функціоналу системи, інтернет-магазин, використання Magento 2

4. Зміст роботи (перелік питань, які потрібно розробити)
Спроектувати і розробити систему для продажу спортивного одягу з використанням Magento 2. Система повиннавідповідати архітектурному шаблону MVVM. Код програми повинен бути об'єктно орієнтованим. Дослідити і розробити можливість інтеграції з кастомізатором продуктів.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Діаграма взаємодії підсистем, діаграма варіантів використання, діаграма класів, діаграма послідовностей, слайди презентації

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Галина Михайлівна	30.11.2022	
Безпека в надзвичайних ситуаціях	Клепчик Василь Михайлович		

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	19.09.2022	
2	Огляд існуючих методів	27.09.2022	
3	Основна частина	15.10.2022	
4	Підготовка пояснювальної записки	25.11.2022	
5	Спецчастина	27.11.2022	
6	Підготовка презентації та доповіді	28.11.2022	
7	Попередній захист	30.11.2022	
8	Нормоконтроль, рецензування		
9	Занесення диплома в електронний архів		
10	Допуск до захисту у зав. кафедри		

Студент _____ Простяк В.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

Анотація

Проект на тему «Розробка web-сайту інтернет-магазину для продажу спортивного одягу на базі eCommerce платформи Magento 2».

Простяк Володимир Михайлович. Тернопільський національний технічний університет імені Івана Пулюя, Факультет комп'ютерно-інформаційних систем і програмної інженерії, Кафедра програмної інженерії, група СПм–61, Тернопіль – 2022.С. – 83, рисунків – 49, додатків – 6.

Мета проекту: розробити сайт, який взаємодіє з базою даних для можливості перегляду інформації про товари, відгуки користувачів, покупку та продаж власної продукції. Для розробки даного продукту було використано PHP, JS, розробка користувацького інтерфейсу була здійснена за допомогою мову розмітки HTML, CSS, системою керування базами даних обрано MySQL, робота була здійснена на основі eCommerce платформи з відкритим кодом Magento 2. Користь проекту полягає у великому попиті сучасного населення на покупки різних товарів в інтернеті. У пошуках бажаного до товару тратиться велика кількість часу. Розроблений сайт допомагає зекономити досить велику кількість часу за допомогою звичайного перегляду списку, відгуків і опису товарів, покупки та доставки у ваше місто. Проблему зберігання доволі великих обсягів інформації вирішує розроблена база даних, яка допомагає набагато швидше відшукати потрібну інформацію ніж в паперових аналогах. Та й в загальному підвищує зручність та ефективність роботи продукту.

Abstract

Project on " Development of a website for an online store for the sale of sportswear based on the Magento 2 eCommerce platform".

Prostiak Volodymyr Mikhailovich. Ternopil Ivan Pulyuy National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Software Engineering, SPm-61 Group, Ternopil – 2022.C. – 83, drawings – 49, applications – 6.

The purpose of the project: to develop a site that interacts with the database to view information about products, user reviews, buy and sell their own products. PHP, JS was used to develop this product, the user interface was developed using HTML markup language, CSS, the database management system selected MySQL, the work was done on the basis of the Magento 2 open source eCommerce platform. The benefit of the project is the high demand of the modern population to buy various goods online. A lot of time is spent searching for the desired product. The developed site helps to save a considerable amount of time by means of usual viewing of the list, responses and the description of the goods, purchase and delivery to your city. The problem of storing quite large amounts of information is solved by a developed database, which helps to find the necessary information much faster than in paper counterparts. And in general increases the convenience and efficiency of the product.

Зміст

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ І ТЕРМІНІВ

Magento 2 - це програмне забезпечення створене з використанням Zend Framework.

GIT - розподілена система керування версіями файлів та спільної роботи.

MVC - архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення. Цей шаблон передбачає поділ системи на три взаємопов'язані частини: модель даних, вигляд та модуль керування.

MVVM (Model-View-ViewModel) - шаблон проєктування, що застосовується під час проєктування архітектури застосунків. MVVM полегшує відокремлення розробки графічного інтерфейсу від розробки бізнес логіки (бек-енд логіки), відомої як модель (можна також сказати, що це відокремлення представлення від моделі).

Репозиторій - патерн, який розділяє рівні джерела даних і логіки програми. Рівень абстракції між Business Logic рівнем та Data Access рівнем

СУТНІСТЬ-АТРИБУТ-ЗНАЧЕННЯ (Entity–attribute–value model (EAV)) – патерн для проектування бази даних.

Дані записуються у вигляді трьох стовпців:

- Сутність: предмет, що описується.
- Атрибут або параметр: зазвичай реалізується як зовнішній ключ у таблиці визначень атрибутів.
- Таблиця визначень атрибутів може містити такі стовпці: ідентифікатор атрибута, ім’я атрибута, опис, тип даних і стовпці, які допомагають перевірити вхідні дані, наприклад, максимальна довжина рядка та регулярний вираз, набір допустимих значень тощо. Значення атрибута.

ВСТУП

Останніми роками в усьому світі все більшого поширення набувають методи ведення бізнесу в Інтернеті, зокрема. інтернет-торгівля. Застосування Інтернету дозволяє швидко і з незначними витратами вивести і просувати продукцію на національний і міжнародні ринки. Торгівля через Інтернет дозволяє істотно знизити вартість продукції, оскільки відпадають потреби в утриманні торгових площ, придбанні торгівельного обладнання, не потрібно утримувати торгівельний персонал тощо. Крім того, споживач у реальному масштабі часу може переглянути номенклатуру продукції, що реалізується, швидко знайти потрібний товар, вияснити його характеристики, ознайомитися з відгуками інших споживачів, обрати зручний спосіб і час доставки товару, провести платежі через Інтернеті і т. ін.

Дана робота є актуальною і корисною, оскільки інтернет-торгівля набуває широкого поширення. Цей факт дає підставу для проведення такого роду модифікацій для подальшого розвитку галузі і збільшення зручності користування продуктом.

Метою роботи є дослідження і глибше вивчення можливостей розробки інтернет-магазинів на основі Magento 2. Ознайомлення з новою версією системи, а саме Magento 2.4.3.

Об’єктом дослідження роботи є інтернет торгівля та системи розробки інтерне-магазинів. Предметом дослідження вибрано систему Magento 2, як одну із найпопулярніших у цій сфері.

Ціллю даної роботи є створення інтернет-магазину на основі eCommerce платформи з відкритим кодом Magento 2. У ході дослідження і розробки повинен бути створений готовий до експлуатації магазин, з можливістю розширення та високим рівнем надійності.

1. ОГЛЯД ПЕДМЕТНОЇ ОБЛАСТІ РОЗРОБКИ ІНТЕРНЕТ-МАГАЗИНІВ

1.1 Огляд конкурентів

В наш час існує багато площадок для покупки речей в інтернеті, різні форуми для обміну враженнями. Метою моєї роботи є розробити подібний сайт для перегляду інформації, відгуків та покупки різного роду товарів. Для розробки свого продукту я проаналізував ринок і вибрав декілька прикладів конкурентів, які будуть наведені нижче.

Першим розглянутим сайтом був магазин де продаються різного роду інструменти (див. рисунок 1.1).

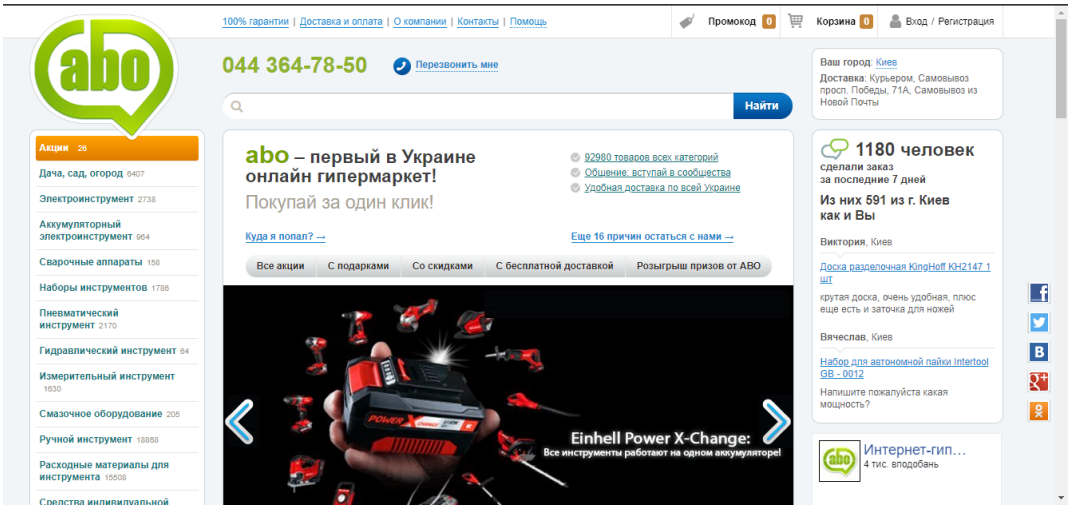

Рисунок 1.1 – Сайт для покупки інструментів

Недоліками даного сайту є відсутність товарів, які не стосуються будівництва. Також мінусом є відсутність налаштувань мови сайту.

Інтерфейс користувача є доволі простим і зрозумілим, що є доволі великим плюсом цього сайту.

Наступним був розглянутий магазин з продажу музики «Sony music» написаний на основі WordPress (див. рисунок 1.2). Даний сайт є гарно оформленим але з невеликою кількістю можливостей. Технологія WordPress створена для розробки різного сайтів і не зовсім підходить для створення вебсайтів з великою

кількістю магазинів. Також недоліком є те, що функціонал самого магазину розроблений на основі цієї технології є доволі невеликим та сам магазин не може містити велику кількість товарів, зміни в атрибутах продуктів важко здійснювати. Головною перевагою таких магазинів є простота у зміні зовнішнього вигляду сайту.

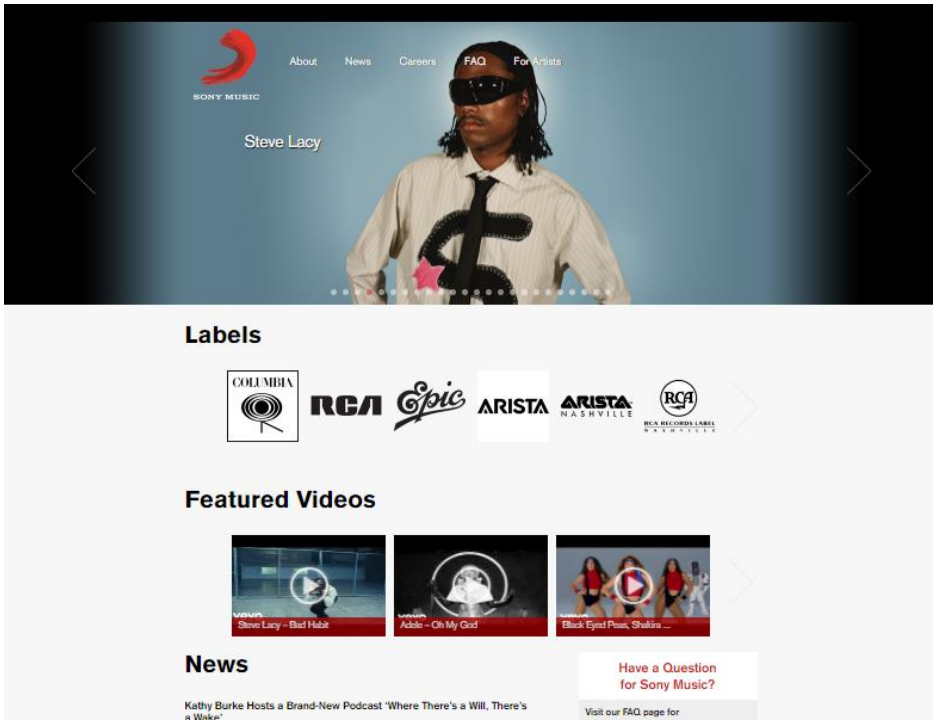

Рисунок 1.2 – Інтернет-магазин «Sony music»

Наступний розглянутий магазин був створений на основі технології Opencart, це площадка продажу лікарських засобів «Біосфера»(див. рисунок 1.3).

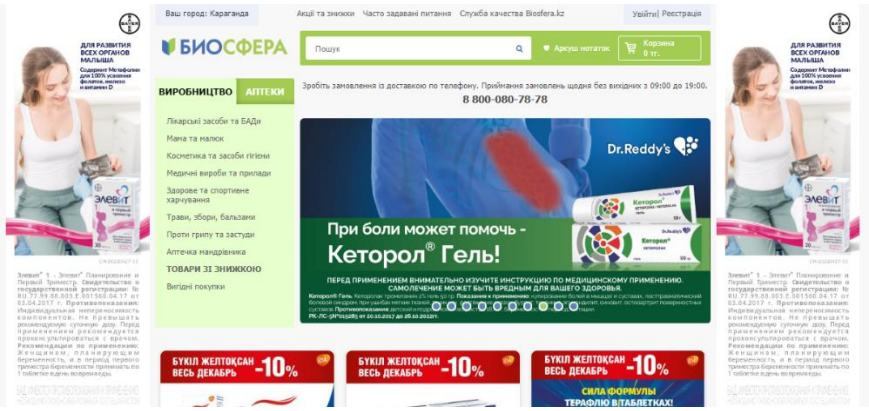

Рисунок 1.3 – Інтернет-магазин «Біосфера»

Недоліками магазинів на основі цього CMS можна вважати доволі малі можливості в розробці графічного інтерфейсу користувача. Також в ході дослідження було виявлено, що у Opencart досить погана SEO частина, що значно погіршує роботу сайтів на основі цієї технології у ролі торгівельних площадок.

Також в ході аналізу конкурентів було розглянуто магазин «Ideal Sport». Даний інтернет-магазин написаний на основі Magento 2(див. рисунок 1.4). Перевагою над іншими є те, що на цьому вебсайті представлено два магазини для різних носіїв мови. Також функціонал магазину є доволі широким, а оскільки він написаний на основі Magento 2, то має багато можливостей для розширення в разі потреби.

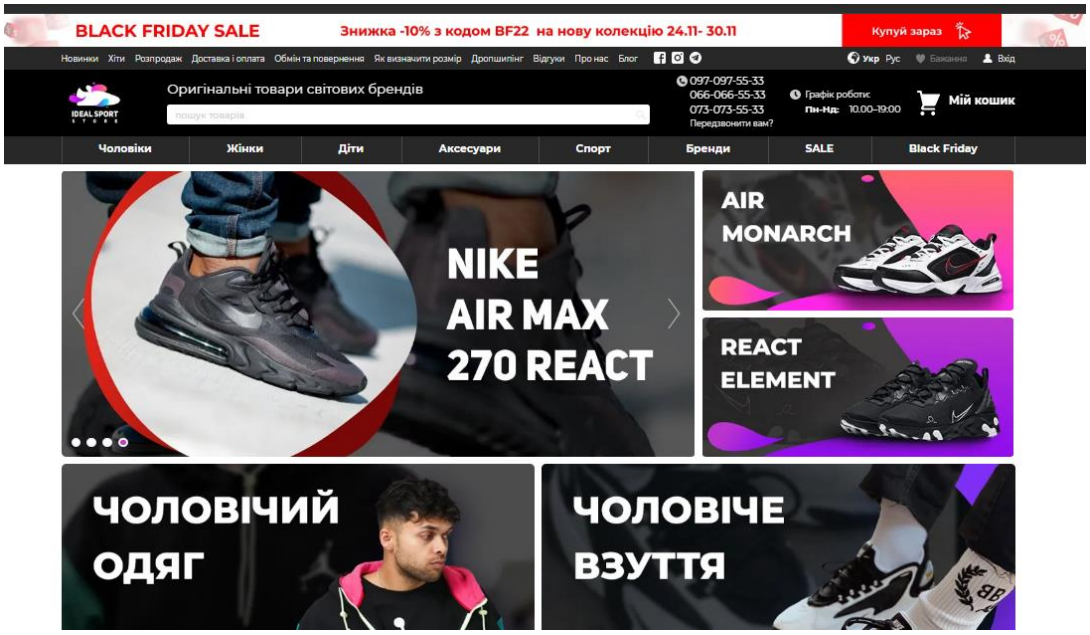

Рисунок 1.4 – Інтернет-магазин «Ideal Sport»

Даний сайт є чудовим прикладом інтернет магазину. В ході досліджень цей сайт був обраний як основний конкурент. В подальшій роботі буде проведено розробку функціоналу, який відсутній в оглянутих магазинах для покращення користування покупцями торгівельної площадки, тим самим отримати перевагу над вище описаними магазинами.

1.2 Обґрунтування вибору напрямку дослідження

Для початку роботи над проектом необхідно чітко розуміти які саме технології використовувати. В наш час є доволі велика кількість систем для створення інтернет-магазинів, деякі з них були розглянуті у розділі 1.1. Розробку самого магазину було вирішено зробити на основі Magento 2.4.3, оскільки це найновіша стабільна версія цієї CMS системи і в ході досліджень було виявлено значну кількість переваг над іншими схожими системами. Також вагомим аргументом у виборі цієї CMS стала висока популярність на ринку розробки інтернет-магазинів.

Magento 2 – це eCommerce платформа з відкритим кодом, програмне забезпечення якої створене з використанням Zend Framework [4]. Основними перевагами цієї системи є:

- Орієнтація на e-commerce
- Пошукова оптимізація
- Маркетингові інструменти
- Аналітика та звітність
- Можливості каталогу
- Стійкість до вірусів
- Простота адміністрування
- Професійна підтримка

Вибрана система має клієнт-серверну архітектуру. Backend(серверна частина) сайту написана з використанням PHP. В якості системи керування базами даних використовується MySQL. Frontend(клієнтська частина сайту) розроблена з використанням HTML, CSS та JavaScript. У нових версіях також була добавлена можливість розробки клієнтської частини сайту з використанням PWA(Progressive Web Application), а обмін даними ведеться за допомогою GraphQl API. Станом на 2022 рік популярність розробки сайтів з використанням технології прогресивної веб-сторінки(PWA) значно зросли, що є хорошою тенденцією, оскільки сайти розроблені таким чином працюють значно швидше і виглядають набагато краще.

Хоча такий спосіб розробки значно збільшує вартість проекту та кількість необхідних спеціалістів, оскільки одній людині дуже важко одночасно працювати з клієнтською і серверною частинами сайту у разі використання PWA на проекті.

Для розробки сайту для цієї роботи було вирішено використовувати звичайний спосіб розробки Frontend частини Magento 2, для розробки серверної частини було обрано PHP, оскільки ці всі технології є найбільш оптимальними для роботи з Magento 2.

1.3 Технічний аспект проблеми

Для початку роботи над магазином необхідно розгорнути проект на пристрої. У ході дослідження виявлено, що є три основних способи розгортання такого роду проектів:

- LAMP
- LXS
- Docker

LAMP – це абревіатура до Linux, Apache, MySQL і PHP. Разом вони надають перевірений набір програмного забезпечення для створення веб-додатків.

LXC – це інтерфейс, який дозволяє користувачам Linux легко створювати та керувати системними контейнерами або контейнерами програм.

Docker – це відкрита платформа для розробки, доставки та запуску програм. Він надає можливість пакувати та запускати програму в слабко ізольованому середовищі, яке називається контейнером.

Для роботи було вибрано систему Ubuntu(сучасна операційна система з відкритим кодом для Linux), оскільки вона є доволі зручною для розробки ПЗ та роботи у консолі. Перечислені вище системи керування контейнерами ідеально підходять для роботи з вибраною ОС.

У всіх зазначених середовищ, які були зазначені вище є свої переваги та недоліки. В ході досліджень було вирішено відмовитися від використання LAMP через велику складність встановлення і підтримки системи та через незручності в роботі з Magento 2. Найбільш зручним варіантом на перший погляд здавалися LXC контейнери, вони зручні у роботі на ОС Linux та доволі зручні для роботи з вибраною CMS. Хоча недоліком цих контейнерів виявилося саме розгортання проекту, яке є доволі складним через невелику кількість літератури. Тому було вирішено обрати Docker контейнери через їхню простоту у встановленні та користуванні. Головною перевагою цієї технології стало те, що для Magento 2 було розроблено спеціальне середовище на базі Docker, а саме Dockergento.

Dockergento – це просто bash скрипт, готовий до використання в Linux і Mac, щоб мати можливість використовувати Docker із найкращою продуктивністю при роботі з Magento 2.

Також у ході проектування сайту виникла проблема з вибором методології розробки програмного забезпечення. Оскільки PHP дозволяє писати як функціональний, так і об’єктно-орієнтований код, то вибір постав між методологією функціонального програмування та методологією об’єктно-орієнтованого програмування. У ході дослідження було виявлено, що платформа Magento 2 надає перевагу ООП. Тож саме ця методологія була обрана.

Об'єктно-орієнтоване програмування (ООП) – парадигма програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основних принципи, а саме: інкапсуляція, наслідування і поліморфізм [13].

Інкапсуляція – це принцип при якому клас містить не тільки дані, а і методи для їх обробки. Доступ до стану об’єкта напряму заборонено, зовні робота з ним дозволена через інтерфейси.

Наслідування – принцип, який дає можливість класу наслідувати код іншого (батьківського) класу, та доповнювати, або перевизначати певні його методи. У PHP не дозволено множинне наслідування. Клас нащадок також успадковує тип класу предка.

Поліморфізм – це принцип в основі якого лежить використання єдиного інтерфейсу для різних сутностей або використання одної змінної для маніпуляції даними різного типу (виклику різних методів класу змінної).

2 РОЗРОБКА МОДЕЛІ ТА ПРОГРАМНОГО КОМПЛЕКСУ

2.1 Розробка моделі предметної області та бізнес моделі

2.1.1 Розробка моделі предметної області

Для розробки програмного забезпечення було обрано шаблон модель-вид-контролер(MVC)(див. рисунок 2.1). Перевагами цієї архітектури є:

- Розробка програми стає швидкою.
- Декільком розробникам легко співпрацювати та працювати разом.
- Простіше оновити програму.
- Легше налагоджувати, оскільки ми маємо кілька рівнів, належним чином написаних у програмі.

Недоліки архітектури MVC:

- Важко зрозуміти архітектуру MVC.
- Повинні бути суворі правила щодо методів.

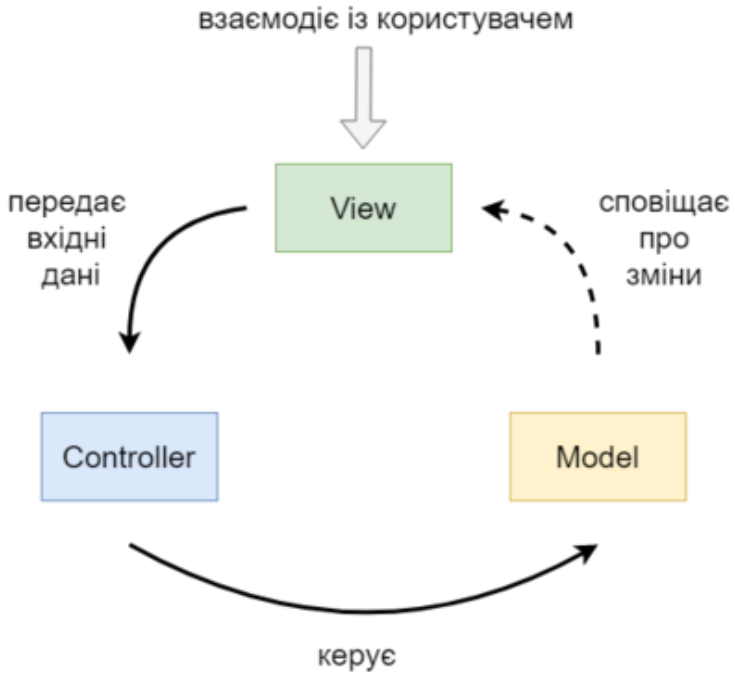

Рисунок 2.1 – Графічне зображення шаблону MVC

У ході вивчення архітектури Magento 2 було виявлено, що кращим варіантом архітектурного шаблону для розробки на базі цієї платформи буде Model-View-ViewModel(MVVM) [5] (див. рисунок 2.2).

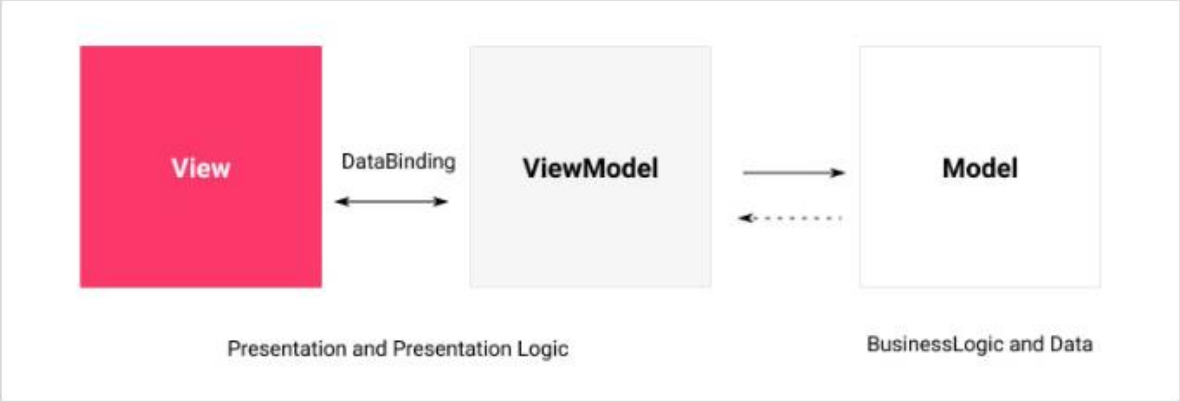

Рисунок 2.2 – Графічне зображення шаблону MVVM

MVVM має три рівні:

- Model

  Модель містить бізнес-логіку програми та залежить від пов’язаного класу - ResourceModel – для доступу до бази даних. Моделі залежать від контрактів на обслуговування, щоб надати свою функціональність іншим рівням програми.

- View

  Вид – це і структура, і макет того, що видно на екрані - фактичний HTML. Це досягається у файлах PHTML, що поширюються з модулями. Такі файли пов’язані з кожною ViewModel у XML-файлах - Layout, які іноді називають сполучними. Файли макета також можуть прив’язувати файли JavaScript для використання на сторінці.

- ViewModel

  ViewModel працює разом із шаром Model і надає лише необхідну інформацію шару View. У Magento 2 це обробляється класами блоків модуля. Зверніть увагу, що зазвичай це була частина ролі контролера системи MVC. У MVVM контролер відповідає лише за обробку потоку користувача, тобто він отримує

запити та повідомляє системі відобразити перегляд або перенаправити користувача на інший маршрут.

Архітектура Magento 2 складається з 4 рівнів [5]:

- Рівень презентації

  Презентаційний рівень – це верхній рівень, який містить елементи користувацької частини(макети, блоки, шаблони) і контролери. Презентаційний рівень зазвичай викликає сервісний рівень за допомогою сервісних контрактів. Але, залежно від реалізації, це може перетинатися з бізнес-логікою.

- Сервісний рівень

  Сервісний рівень – це рівень між презентаційним і доменним рівнями. Він виконує сервісні контракти, які реалізовані як інтерфейси PHP. Сервісні контракти дозволяють додавати або змінювати модель ресурсу бізнес-логіки за допомогою файлу ін’єкції залежностей (di.xml). Сервісний рівень також використовується для надання доступу до API (REST / SOAP або інші модулі). Інтерфейс сервісів оголошено в просторі імен / API модуля. Інтерфейс даних (сутності) оголошено всередині /Api/Data.

  Сутності даних – це структури даних, що передаються до інтерфейсів служби та повертаються з них.

- Доменний рівень

  Доменний рівень відповідає за бізнес-логіку, яка не містить інформації про ресурси та базу даних. Крім того, доменний рівень може включати реалізацію контрактів на обслуговування. Кожна модель даних на рівні домену залежить від ресурсної моделі, яка відповідає за доступ до бази даних.

- Рівень постійності

  Рівень постійності описує модель ресурсів, яка відповідає за отримання та зміну даних у базі даних за допомогою запитів CRUD. Він також реалізує додаткові функції бізнес-логіки, такі як перевірка даних і реалізація функцій бази даних.

2.1.2 Розробка бізнес моделі взаємодії користувачів з магазином

При розробці бізнес моделі важливо чітко знати ролі користувачів системи та їх взаємодію із сайтом, проаналізувати всі варіанти використання магазину користувачами. Для кращого розуміння моделі краще розробити UML діаграми, які в подальшому буде реалізована у вигляді програми.

Розробку бізнес моделі варто почати із загальної UML діаграми зовнішньої взаємодії. Тому першою розробленою діаграмою буде діаграма варіантів використання інтернет-магазину [5] (див. рисунок 2.3).

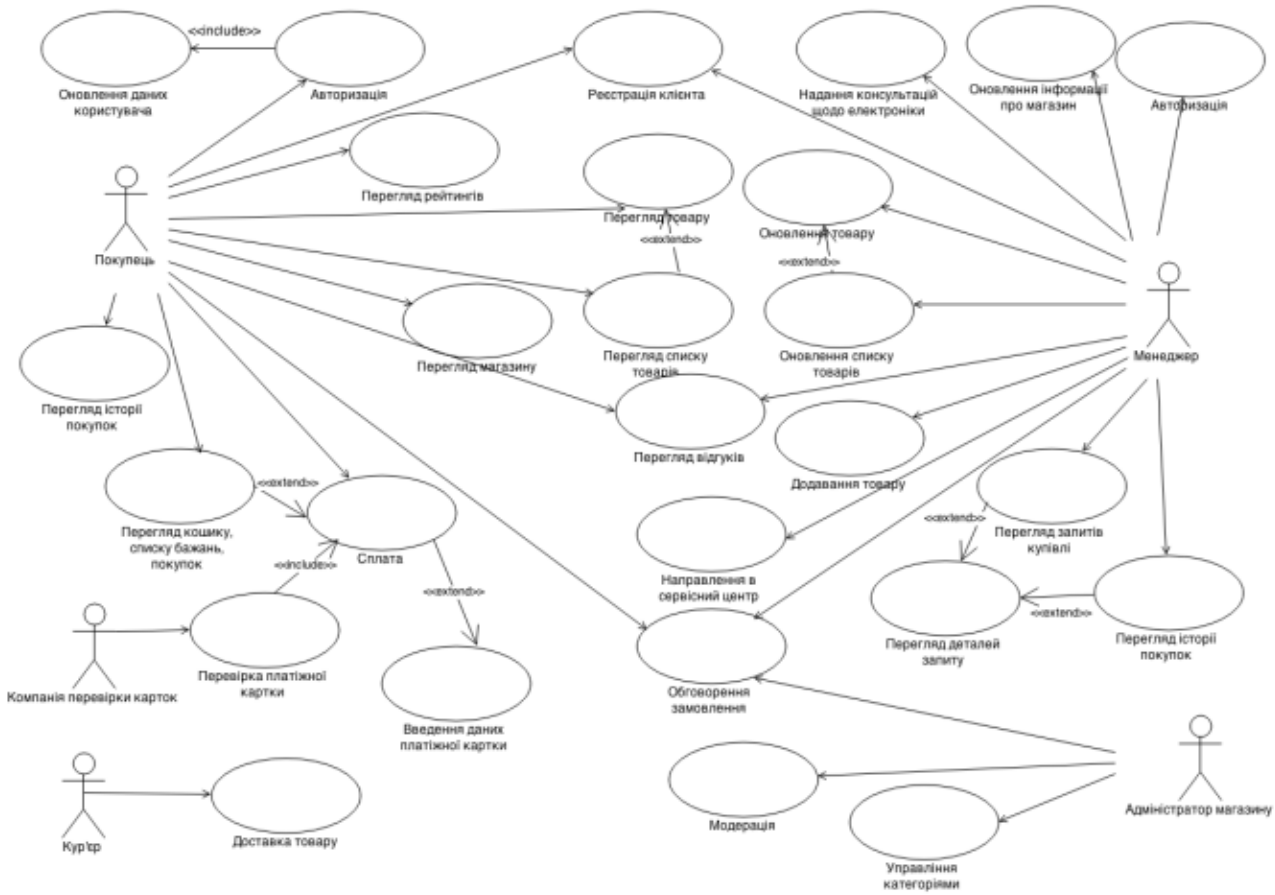

Рисунок 2.3 – Діаграма варіантів використання інтернет-магазину

В ході розробки бізнес моделі було розглянуто три категорії користувачів: покупець, менеджер і адміністратор. Вибір такого набору дійових осіб був

зумовлений різними можливостями для цих груп користувачів. Після цього можна створити UML діаграми активності(діяльності) [7].

Для побудови діаграм діяльності необхідно проаналізувати можливі потоки подій. Результати аналізу наведені у таблицях потоків подій(див. табл. 2.1).

Таблиця 2.1 – Потоки подій

№	Потоки подій	Опис
1	Пошук товарів	• клієнт за допомогою пошукового поля створює запит на пошук товару в таблиці «Товари»; • СУБД звертається до таблиці «Товари», після чого починається пошук товару за його унікальним кодом або співпадіння ключових слів; • результати запиту передаються клієнту; • користувачу відкривається сторінка зі списком товарів або товаром.
2	Вибір товару	• клієнт створює запит на вибір товару з таблиці «Категорії»; • СУБД звертається до таблиці «Категорії», починається пошук категорій товарів за ідентифікатором категорії; • таблиця «Категорії» передає результат запиту СУБД, результати виводяться користувачу.

Продовження таблиці 2.1

№	Потоки подій	Опис
3	Замовлення товару	• клієнт додає товар у кошик; • СУБД звертається до таблиці «Товари», у ній відбувається пошук товару та додавання ідентифікатора товару у таблицю «Квот»; СУБД повертає результат пошуку, який відображається у корзині користувача.
4	Оплата товару	• клієнт створює запит на оплату товару; • залежно від доступних у магазині і вибраних користувачем способів оплати формується запит до стороннього ПЗ для верифікації та оплати товару; • клієнт вводить дані свого банківського рахунку(якщо це необхідно) та дані про адрес доставки товару, після цього натискає на кнопку «Оплатити рахунок»; • клієнт оплачує рахунок.

Продовження таблиці 2.1

№	Потоки подій	Опис
5	Залишити відгук про товар	• клієнт створює запит до бази даних на перегляд товару; • СУБД звертається до таблиці «Товари», пошук товару відбувається за його унікальним кодом або співпадіння ключових слів; • результат пошуку передаються СУБД, відкривається сторінка товару на стороні клієнта; • клієнт вписує свій відгук у відповідне поле, після цього створюється запит на запис коментаря у таблицю «Товари»; • після успішного запису у таблицю результат відправляється клієнту у вигляді повідомлення про успіх.

Діаграми активності було вирішено побудувати на основі трьох варіантів використання, а саме: «Вхід користувача в систему»(див. рисунок 2.4), «Пошук товарів у системі», «Оцінка товару користувачем». Також варто проаналізувати і створити діаграми діяльності для альтернативного ходу подій.

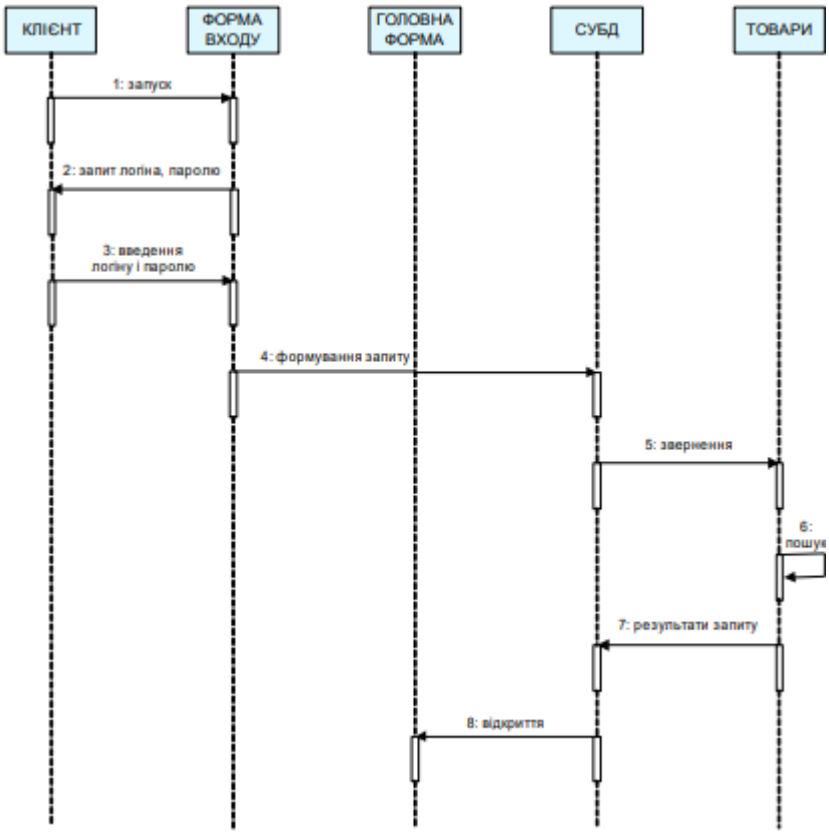

Рисунок 2.4 – Діаграма активності для варіанту використання «Вхід користувача в систему»

Аналіз альтернативних варіантів необхідний для того, щоб визначити більшість варіантів подій, а саме коли основний потік не може пройти успішно. Для варіанту «Вхід користувача в систему» альтернативний потік буде використаний, якщо одне з полів вводу заповнене неправильно(див. рисунок 2.5). Для варіанту використання «Пошук товарів у системі» альтернативним потік буде застосовано, якщо товар не буде знайдено у базі даних. При варіанті використання «Оцінка товару користувачем» цей потік буде застосований коли користувач спробує повторно оцінити уже оцінений товар. Створені діаграми діяльності наведені у додатку В.

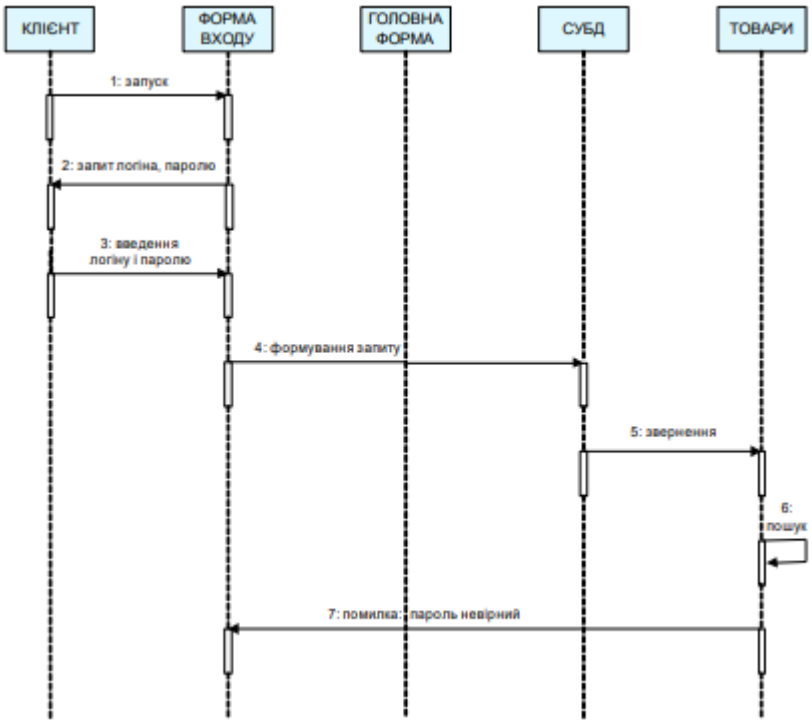

Рисунок 2.5 – Діаграма активності для альтернативного потоку варіанта використання «Вхід користувача в систему»

Таким чином в ході аналізу було розроблено UML діаграму варіантів використання інтернет-магазину, розглянуто потоки подій та побудовано діаграми діяльності для трьох варіантів використання, з урахуванням альтернативних потоків подій для кожного з варіантів використання.

2.2 Проектування класів інтернет-магазину

Проаналізувавши основні функції магазину та створивши діаграму варіантів використання варто приступити до розробки діаграми класів, та дослідити їх взаємодію між собою.

Діаграма класів – це тип діаграми статичної структури, який описує структуру системи, показуючи класи системи, їхні атрибути, операції (або методи) і зв’язки між об’єктами.

Перед тим як розробити діаграму класів системи із всіма зв’язками потрібно розглянути кожен клас окремо. Система Magento 2 є доволі великою тому доцільно буде оглянути основний функціонал та спроектувати діаграму основних класів. Важливо також взяти до уваги те, що у обрана CRM використовує EAV патерн для побудови структури бази даних.

Сутність-атрибут-значення (EAV) – це патерн в якому структуру таблиць бази даних ділять на три види: сутність(запис основних сутностей моделей, наприклад, продукт), атрибут(атрибути сутності, наприклад, ціна) та значення(фактичне значення атрибута). Через це класи системи будуть відправляти запити до різної кількості таблиць у базі даних. Такий підхід значно ускладнює процес побудови запитів але є більш зручним для додавання нових атрибутів до продукта.

Клас Product (див. рисунок 2.6) відповідає за продукти. Він містить в собі деталі про назву продукту, його інформацію, ціну, категорію, фото. Адміністратор може створювати, редагувати та видалити продукти, а також змінювати його стан, якщо у цьому є необхідність.

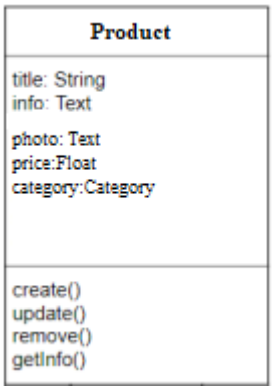

Рисунок 2.6 – Клас Product

Клас Order (див. рисунок 2.7) відповідає за здійснені покупки. Він містить в собі такі деталі: продукт, який купляють, користувач, який здійснив покупку, розмір оплати. В цьому класі можна отримати інформацію про всі закази.

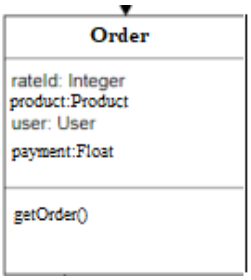

Рисунок 2.7 – Клас Order

Клас Category (див. рисунок 2.8) відповідає за категорії продуктів. Він містить у собі наступні деталі про катеорії: назва категорії, нік категорії (необов'язковий параметр, при його опускані він транслітерує назву категорії), та його опис (необов'язковий параметр). Адміністратор може добавляти категорію, редагувати, або видалити її. Також він може приховати категорію в меню.

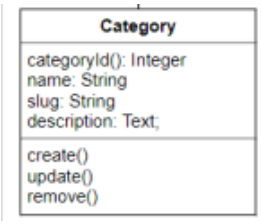

Рисунок 2.8 – Клас Category

Клас User (див. рисунок 2.9) відповідає за всіх користувачів в системі. Він містить у собі наступні деталі про користувачів: їх ідентифікатор, прізвище, ім'я та ім'я користувача (логін). Самі коритсувачі можуть редагувати профіль у випадку якщо їхня інформація про себе змінилася, а адміністоатор може отримати інформацію про них.

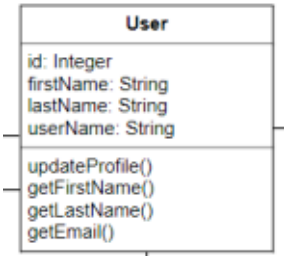

Рисунок 2.9 – Клас User

Клас Admin (див. рисунок 2.10) відповідає за адміністратора в системі. Унаслідується від класу User. Не містить в собі додаткових атрибутів. Напряму з цього класу можна змінювати стан продукту, якщо в цьому є необхідність.

Рисунок 2.10 – Клас Admin

Клас Guest (див. рисунок 2.11) відповідає за гостей в системі. Цей клас є дочірнім від батьківського класу User. По суті це є користувач, який тільки зайшов на сайт, він не має ще ніяких атрибутів і не може виконувати особливих функцій в системі. За допомогою цього класу можна зареєструватись та авторизуватись в системі.

Рисунок 2.11 – Клас Guest

Клас Member (див. рисунок 2.12) відповідає за учасників тендерів в системі. Цей клас є дочірнім від батьківського класу User. Містить у собі контактну інформацію про учасників, а також тільки у них є можливість голусування.

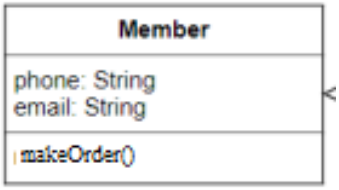

Рисунок 2.12 – Клас Member

Описавши основні класи системи, можемо побудувати діаграму класів, до якої можна звертатись в подальшій розробці (див. рисунок 2.13).

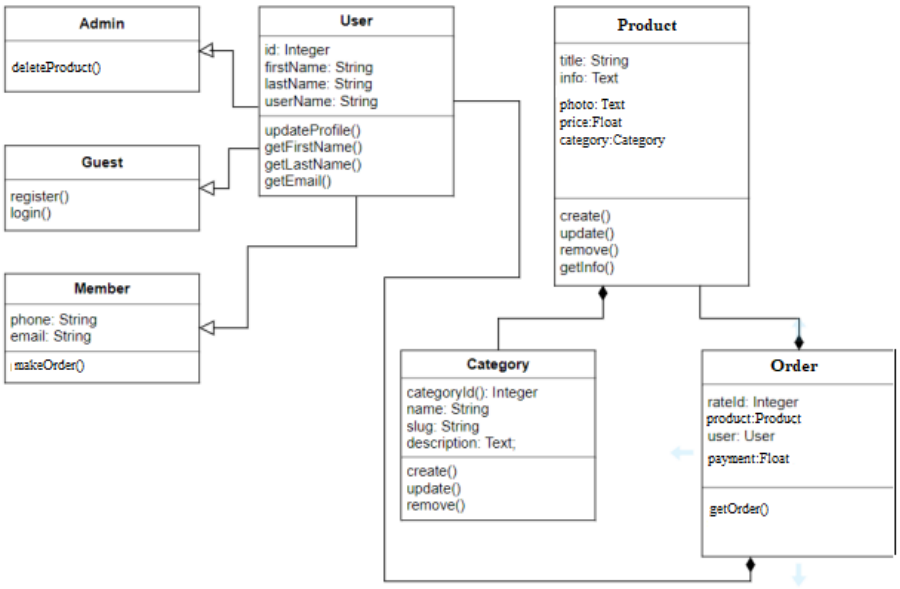

Рисунок 2.13 – Загальна діаграма класів

3 КОНСТРУЮВАННЯ ІНТЕРНЕТ-МАГАЗИНУ НА ОСНОВІ MAGENTO 2

3.1 Реалізація ключових класів

Оскільки Magento 2 це система з великим функціоналом, який наявний в усіх інтернет-магазинах, то було прийняте рішення звернути увагу саме на реалізацію класів функціоналу, які будуть вирізняти розроблений магазин над іншими. Усі фрагменти коду модулів наведено у додатку Г.

Першим розробленим модулем системи [4], який значно покращить досвід взаємодії з нашим сайтом у користувача буде «Out Of Stock Notification» модуль. Його основним завданням є надсилання листа підписаному користувачу при надходженні товару на склад магазину.

В першу чергу потрібно створити табличку в базі даних для збереження всіх підписників і даних на продукт, на який були створені підписки. Для цього у файлі db_schema.xml було описано структуру таблиці(див. рисунок 3.1).

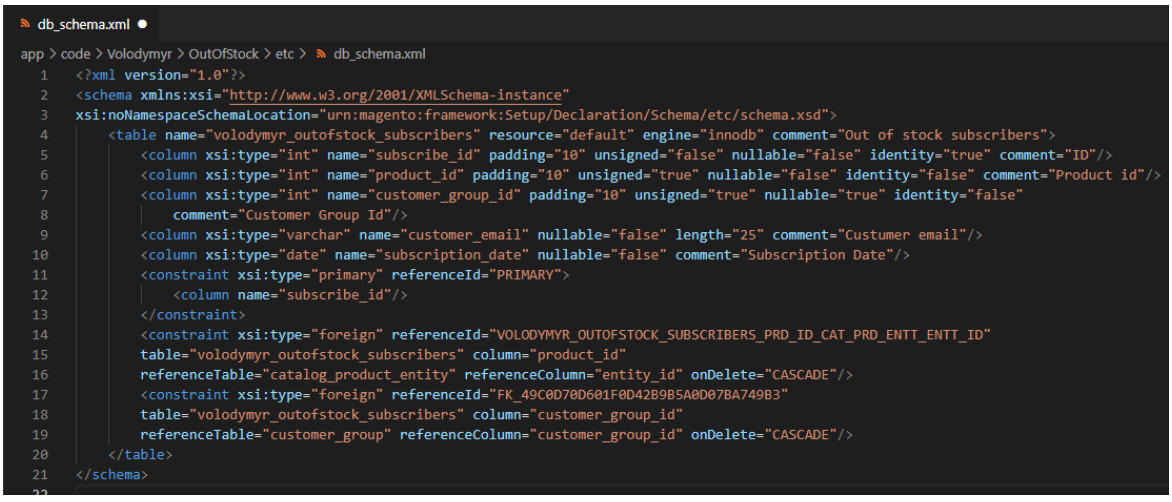

Рисунок 3.1 – Файл db_schema.xml

У цьому файлі описано поля ідентифікатора продукту, ідентифікатор користувача, поштовий адрес, дата підписки та група користувача. Також у db_schema.xml вказані зовнішні ключі, які пов’язують цю таблицю з таблицею продуктів та груп користувачів.

Наступним кроком стала розробка контролера для сторінки з таблицею на стороні адміністрації сайту, для контролю і моніторингу підписок. Оскільки Magento 2 використовує MVVM шаблон проектування, то у контролері буде описано лише один метод, який зрендерить потрібну сторінку і відобразить заголовок сторінки витягнувши його з таблиці конфігурацій(див. рисунок 3.2).

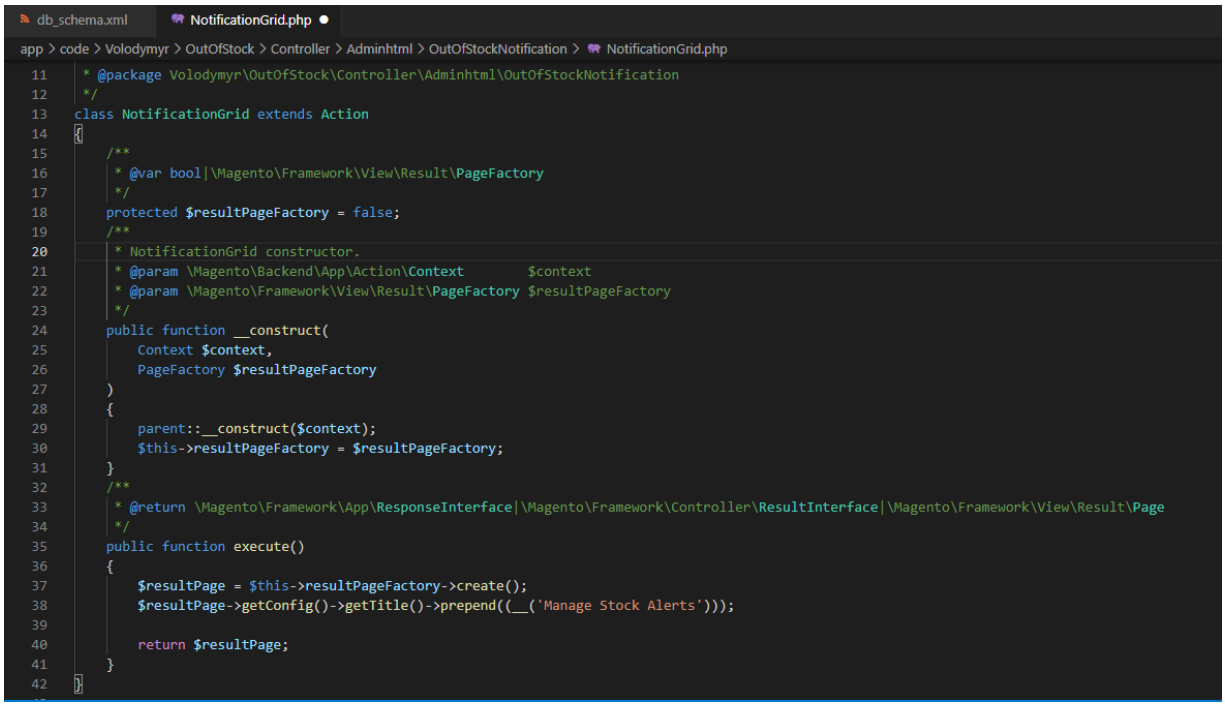

Рисунок 3.2 – Клас NotificationGrid

Також необхідно розробити модель для опису нашої сутності та клас ресурс моделі який надасть зв’язок нашої моделі із базою даних. Клас моделі складається з get та set методів для витягу і додавання даних сутності, а ресурс модель ініціалізує дані про таблицю і первинний ключ(див. рисунок 3.3).

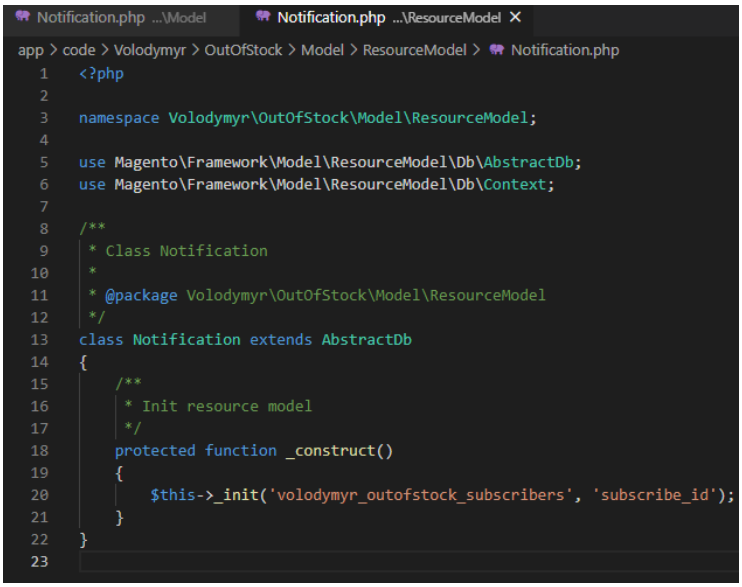

Рисунок 3.3 – Клас ресурс моделі

Також необхідно було розробити репозиторій який би надав можливість виконувати CRUD(збереження, читання, оновлення, видалення) операції над даними, які ми будемо зберігати(див. рисунок 3.4-3.5).

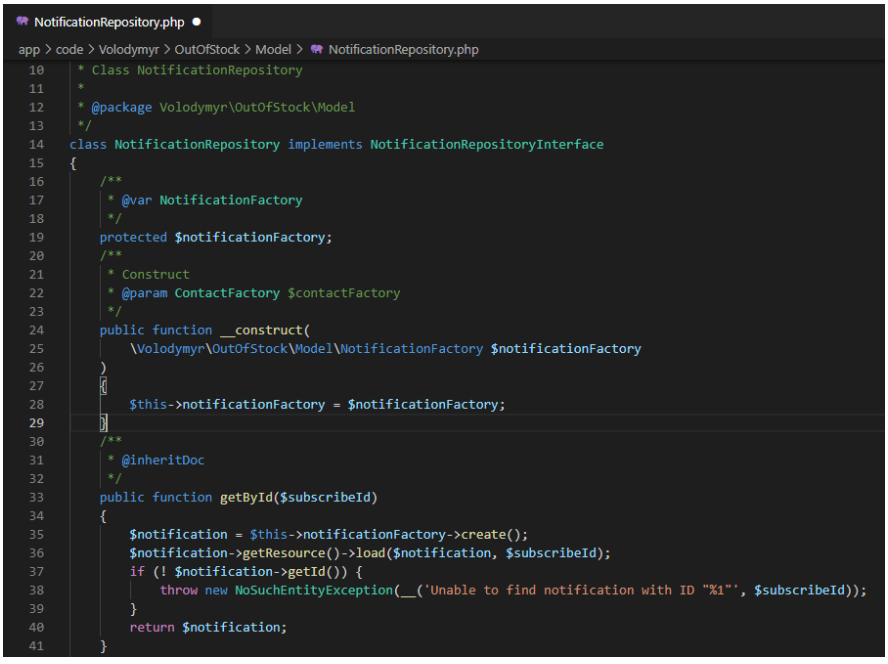

Рисунок 3.4 – Клас репозиторія

```php
    /**
     * @inheritDoc
     */
    public function save(NotificationInterface $notification)
    {
        $notification->getResource()->save($notification);
        return $notification;
    }

    /**
     * @inheritDoc
     */
    public function delete(NotificationInterface $notification)
    {
        $notification->getResource()->delete($notification);
    }

    /**
     * @inheritDoc
     */
    public function deleteById($subscribeId)
    {
        $notification = self::getById($subscribeId);
        $notification->getResource()->delete($notification);
    }
}
```

Рисунок 3.5 – Клас репозиторія

Схожий функціонал контролера був розроблений для сторінки на стороні клієнта у його акаунті. Важливим також був клас який би надав можливість користувачу відписуватись від продукту, якщо він його більше не цікавить. Цей клас також буде реалізований як контролер(див. рисунок 3.6-3.7).

Unsubscribe.php

app > code > Volodymyr > OutOfStock > Controller > Account > Unsubscribe.php

```php
/**
 * Class Unsubscribe
 *
 * @package Volodymyr\OutOfStock\Controller\Account
 */
class Unsubscribe extends Action
{
    /**
     * @var Magento\Framework\Controller\ResultFactory;
     */
    protected $pageFactory;

    public function __construct(
        NotificationRepositoryInterface $notificationRepository,
        ResultFactory $resultRedirectFactory,
        ManagerInterface $messageManager,
        Context $context
    )
    {
        $this->notificationRepository = $notificationRepository;
        $this->resultRedirectFactory = $resultRedirectFactory;
        $this->messageManager = $messageManager;
        parent::__construct($context);
    }
```

Рисунок 3.6 – Клас відписки від продукту

```php
    /**
     * @inheritDoc
     */
    public function execute()
    {
        $id = $this->getRequest()->getParam('id');

        if (!$id) {
            $this->messageManager->addError(__('Unable to proceed. Please, try again.'));
            $resultRedirect = $this->resultRedirectFactory->create();

            return $resultRedirect->setPath('volodymyr_outofstock/account/grid', array('_current' => true));
        }
        try{
            $this->notificationRepository->deleteById($id);
            $this->messageManager->addSuccess(__('Your stock alert has been deleted !'));
        } catch (Exception $e) {
            $this->messageManager->addError(__('Error while trying to delete stock alert: '));
            $resultRedirect = $this->resultRedirectFactory->create();

            return $resultRedirect->setPath('volodymyr_outofstock/account/grid', array('_current' => true));
        }

        $resultRedirect = $this->resultRedirectFactory->create();

        return $resultRedirect->setPath('volodymyr_outofstock/account/grid', array('_current' => true));
    }
}
```

Рисунок 3.7 – Клас відписки від продукту

Наступним необхідним функціоналом був модуль інтеграції з кастомізатором продуктів Zakeke. Сайти з подібним функціоналом зустрічаються доволі рідко, особливо в межах нашої країни, що виділить розроблений магазин із сотень інших.

Першим необхідним функціоналом буде файл web.api для визначення класів, які будуть взаємодіяти з стороннім сервісом(див. рисунок 3.8).

webapi.xml ●

vendor > futurenext > zakeke > etc > webapi.xml

```xml
<routes xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Webapi:etc/webapi.xsd">
    <route url="/V1/zakekeEnabled/:sku" method="GET">
        <service class="Futurenext\Zakeke\Api\ZakekeEnabledRepositoryInterface" method="checkSku"/>
        <resources>
            <resource ref="Futurenext_Zakeke::customization" />
        </resources>
    </route>
    <route url="/V1/zakekeEnabled" method="POST">
        <service class="Futurenext\Zakeke\Api\ZakekeEnabledRepositoryInterface" method="enableBySku"/>
        <resources>
            <resource ref="Futurenext_Zakeke::customization" />
        </resources>
    </route>
    <route url="/V1/zakekeEnabled/:sku" method="DELETE">
        <service class="Futurenext\Zakeke\Api\ZakekeEnabledRepositoryInterface" method="deleteBySku"/>
        <resources>
            <resource ref="Futurenext_Zakeke::customization" />
        </resources>
    </route>
    <route url="/V1/zakekeSettings/keys" method="POST">
        <service class="Futurenext\Zakeke\Api\ZakekeSettingsRepositoryInterface" method="setZakekeApiKeys"/>
        <resources>
            <resource ref="Futurenext_Zakeke::customization" />
        </resources>
    </route>
    <route url="/V1/zakekeColor" method="GET">
        <service class="Futurenext\Zakeke\Api\ZakekeColorRepositoryInterface" method="getColors"/>
        <resources>
            <resource ref="Futurenext_Zakeke::customization" />
        </resources>
    </route>
```

Рисунок 3.8 – Файл web.api

Також необхідними файлами будуть config.xml для запису конфігураційних даних для інтеграції та api.xml для опису доступів у нашій системі для інтеграції(див. рисунок 3.9-3.10).

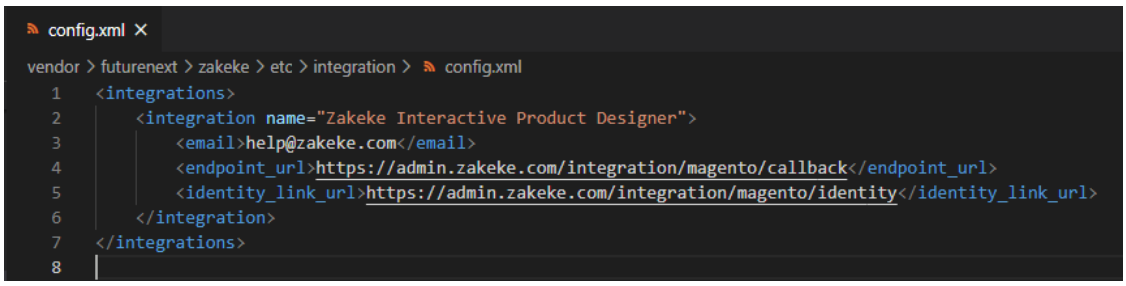

Рисунок 3.9 – Файл config.xml

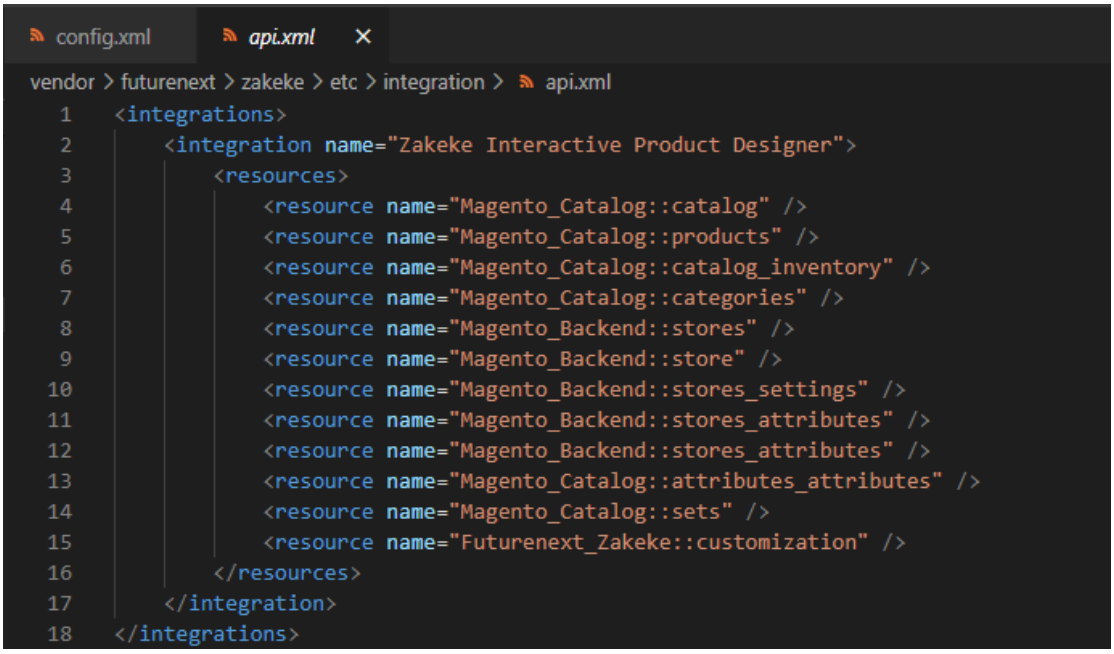

Рисунок 3.10 – Файл api.xml

Інтеграція буде здійснювати CRUD операції, для цього необхідно розробити три репозиторії для роботи з кастомізованими опціями продукту, налаштування доступності продукту для кастомізації та встановлення ключа для інтеграції. Доступи до класів цих репозиторіїв здійснюються через інтерфейси класів(див. рисунок 3.11-3.13).

ZakekeColorRepositoryInterface.php

vendor > futurenext > zakeke > Api > ZakekeColorRepositoryInterface.php

```php
<?php

namespace Futurenext\Zakeke\Api;

use Futurenext\Zakeke\Api\Data\ZakekeColorInterface;
use Magento\Framework\Exception\NoSuchEntityException;

interface ZakekeColorRepositoryInterface
{
    /**
     * Retrieve configurable product data.
     *
     * @return ZakekeColorInterface[]
     * @throws NoSuchEntityException
     */
    public function getColors(): array;
}

```

Рисунок 3.11 – Інтерфейс роботи з кастомізованими опціями продукту

ZakekeEnabledRepositoryInterface.php

vendor > futurenext > zakeke > Api > ZakekeEnabledRepositoryInterface.php

```php
interface ZakekeEnabledRepositoryInterface
{
    /**
     * Check if a product the with given SKU is customizable.
     *
     * @param string $sku
     * @return bool
     * @throws InputException
     */
    public function checkBySku(string $sku): bool;

    /**
     * Remove a product as customizable
     *
     * @param string $sku
     * @return void
     * @throws InputException
     */
    public function deleteBySku(string $sku);

    /**
     * Set product as customizable
     *
     * @param string $sku
     * @return bool
     * @throws InputException
     * @throws StateException
     * @throws CouldNotSaveException
     */
    public function enableBySku(string $sku): bool;
}

```

Рисунок 3.12 – Інтерфейс налаштування доступності продукту для кастомізації

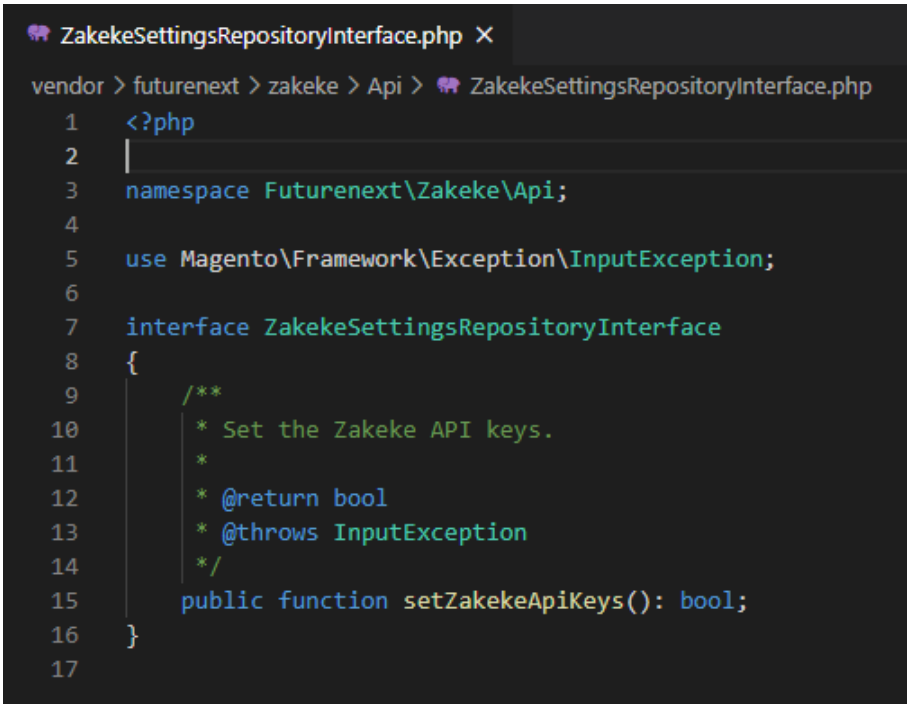

Рисунок 3.13 – Інтерфейс встановлення ключа для інтеграції

Також в ході роботи було створено всі необхідні класи для функціонування описаних вище модулів та декілька інших модулів, а саме: Customer, Brand, Rules, модуль інтеграції з системою оплати Fondy. Фрагменти коду будуть наведені у додатку Г.

3.2 Розробка GUI інтернет-магазину

Magento 2 дає можливість розробляти дизайн сайту на основі вже розроблених, стандартних дизайнів. Таким чином новий вигляд сайту був розроблений на основі стандартного вигляду Block. Усі файли дизайнів розміщуються за шляхом app/design/frontend, а також файли, які відносяться до розроблених вище модулів знаходяться безпосередньо у папках модуля.

В першу чергу варто розглянути користувацьку частину модуля «Out Of Stock Notification». Для цього був розроблений клас блоку з методами для отримання необхідних даних таблиці(див. рисунок 3.14-3.15).

OutOfStockNotification.php

app > code > Volodymyr > OutOfStock > Block > OutOfStockNotification.php

```php
/**
 * Class OutOfStockNotification
 *
 * @package Volodymyr\OutOfStock\Block
 */
class OutOfStockNotification extends View
{
    /**
     * @var Data
     */
    protected $configHelper;
    protected $stockItemRepository;
    protected $customerSession;
    /**
     * Construct
     * @param Data $configHelper
     */
    public function __construct(
        Data $configHelper,
        StockItemRepository $stockItemRepository,
        Context $context,
        EncoderInterface $urlEncoder,
        JsonEncoderInterface $jsonEncoder,
        StringUtils $string,
        Product $productHelper,
        ConfigInterface $productTypeConfig,
        FormatInterface $localeFormat,
        Session $customerSession,
        ProductRepositoryInterface $productRepository,
        PriceCurrencyInterface $priceCurrency,
        array $data = []
```

Рисунок 3.14 – Клас блоку таблиці підписок користувача

OutOfStockNotification.php

app > code > Volodymyr > OutOfStock > Block > OutOfStockNotification.php

```php
    }
    /**
     * @return $productId
     */
    public function getCurrentProductId()
    {
        $productId = parent::getProduct()->getData('entity_id');

        return $productId;
    }
    /**
     * @return $option->getIsInStock()
     */
    public function getStockOption()
    {
        $option = $this->stockItemRepository->get($this->getCurrentProductId());

        return $option->getIsInStock();
    }
    /**
     * @return $this->customerSession->getCustomer()->getEmail()
     */
    public function getCustomerSessionInfo()
    {
        if($this->customerSession->isLoggedIn())
        {
            return $this->customerSession->getCustomer()->getEmail();
        }

        return;
    }
}
```

Рисунок 3.15 – Клас блоку таблиці підписок користувача

Також для функціонування таблиці було додано out_of_stock.phtml файл для відображення на стороні клієнта(див. рисунок 3.16).

out_of_stock.phtml

app > code > Volodymyr > OutOfStock > view > frontend > templates > out_of_stock.phtml

```
<?php
/**
 * @var  $block \Volodymyr\OutOfStock\Block\OutOfStockNotification
 */
if($block->getConfigValue()) {
    if($block->getStockOption() == false) { ?>
        <div >
            <form id = "notification" action = <?= $block->getAction() ?>  method="post">
                <label for = "email" > Subscribe to back in stock notification </label >
                    <br >
                        <input form = "notification" type = "email" id = "email" name = "email"
                            placeholder = "some-email@smile.com"
                            required value = <?= $block->getCustomerSessionInfo(); ?>
                        >
                    <br >
                <input form = "notification" id="productId" name = "productId" type = "hidden"
                    value = <?= $block->getCurrentProductId();?>
                >
                <div style = "margin: 5px 0px 5px 0px" >
                    <input form = "notification"  type = "submit" value = "Submit" style = "font-size: 1.8vh"  >
                </div >
            </form >
        </div >
    <?php }
} ?>

```

Рисунок 3.16 – Файл out_of_stock.phtml

Для дизайну таблиці необхідно створити файл grid.css з описом стилів, кольорів і відступів(див. рисунок 3.17).

grid.css

app > code > Volodymyr > OutOfStock > view > frontend > web > css > grid.css > .admin__field.field-roleLabel

```
.gallery .image .action-make-base > span, .images .image .action-make-base > span {
    border: 0 none;
    clip: rect(0px, 0px, 0px, 0px);
    height: 1px;
    margin: -1px;
    overflow: hidden;
    padding: 0;
    position: absolute;
    width: 1px;
}
.gallery .image .action-make-base:before, .images .image .action-make-base:before {
    color: #9e9e9e;
    content: "\e63b";
    display: inline-block;
    font-family: "Admin Icons";
    font-size: 1.8rem;
    font-weight: normal;
    line-height: 22px;
    margin: 0;
    overflow: hidden;
    text-align: center;
    vertical-align: top;
}
.gallery .image .action-make-base:hover:before, .images .image .action-make-base:hover:before {
    color: #858585;
}
.gallery .image .action-make-base:active:before, .images .image .action-make-base:active:before {
    color: #858585;
}
.gallery .image .action-make-base:focus, .images .image .action-make-base:focus, .gallery .image .action-make-base:active, .images .image
    background: transparent none repeat scroll 0 0;
    border: medium none;
```

Рисунок 3.17 – Файл grid.css

Ідентичний функціонал було розроблено для інших модулів системи. Код загального дизайну сайту були розміщені за шляхом app/design/frontend, до них входять шаблони, js та css файли, які використовуються на сторінках. Фрагменти коду цих файлів наведено у додатку Д.

3.3 Тестування програмного забезпечення та оцінка якості

Після розробки і запуску модулів системи їх необхідно протестувати. Найпершим і найпростішим елементом який потрібно перевірити це дизайн сайту. Дизайн сайту повинен бути виконаним у чорних і оранжевих тонах, у шапці сайту повинен бути розміщений унікальний алгоритм(див. рисунок 3.18).

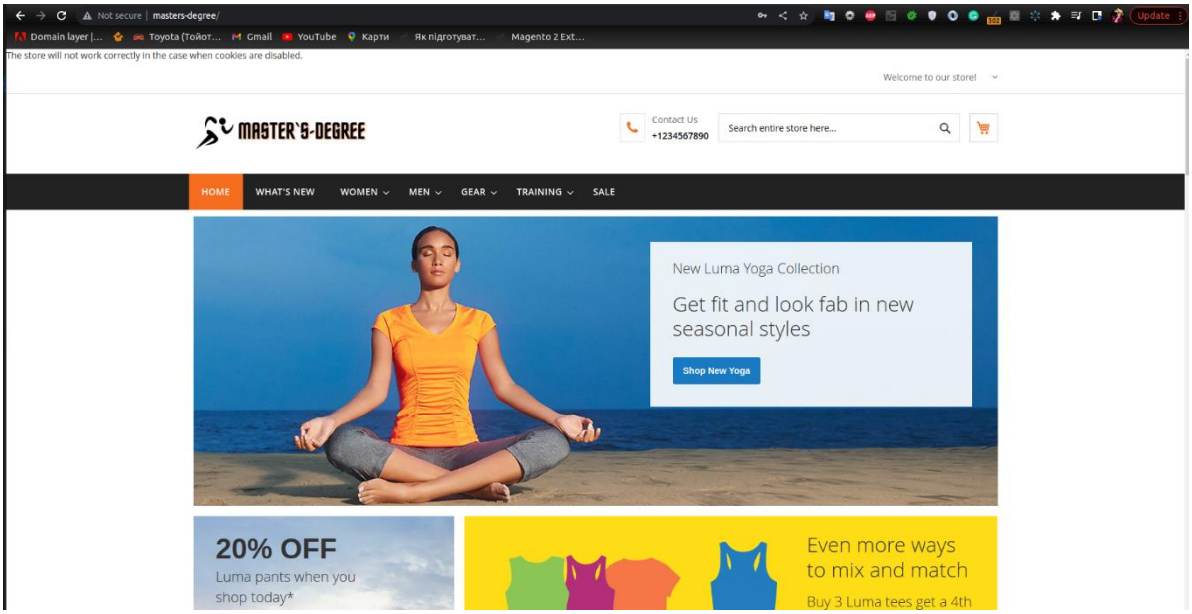

Рисунок 3.18 – Головна сторінка сайту

Також у футері сайту були розміщені всі необхідні посилання, розроблений дизайн для прокруту продуктів(див. рисунок 3.19).

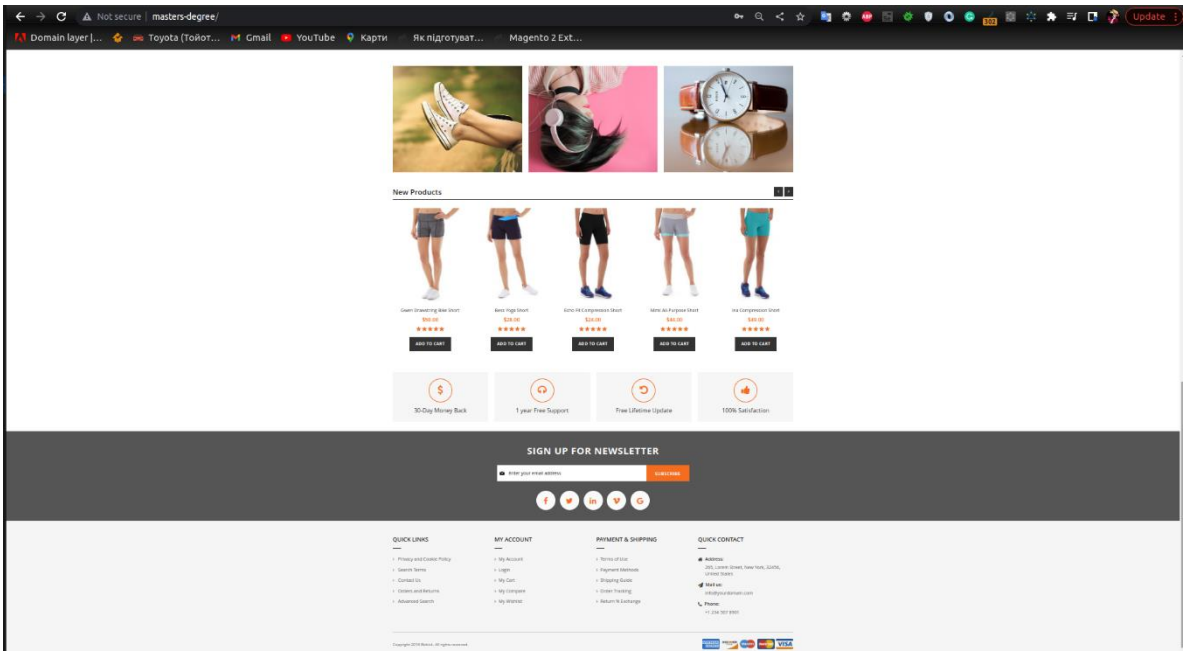

Рисунок 3.19 – Футер сайту

Наступним функціоналом який необхідно перевірити буде модуль «Out Of Stock Notification». У цьому модулі розроблена таблиця у панелі адмінінстратора та CRUD операції для цієї таблиці(див. рисунок 3.20).

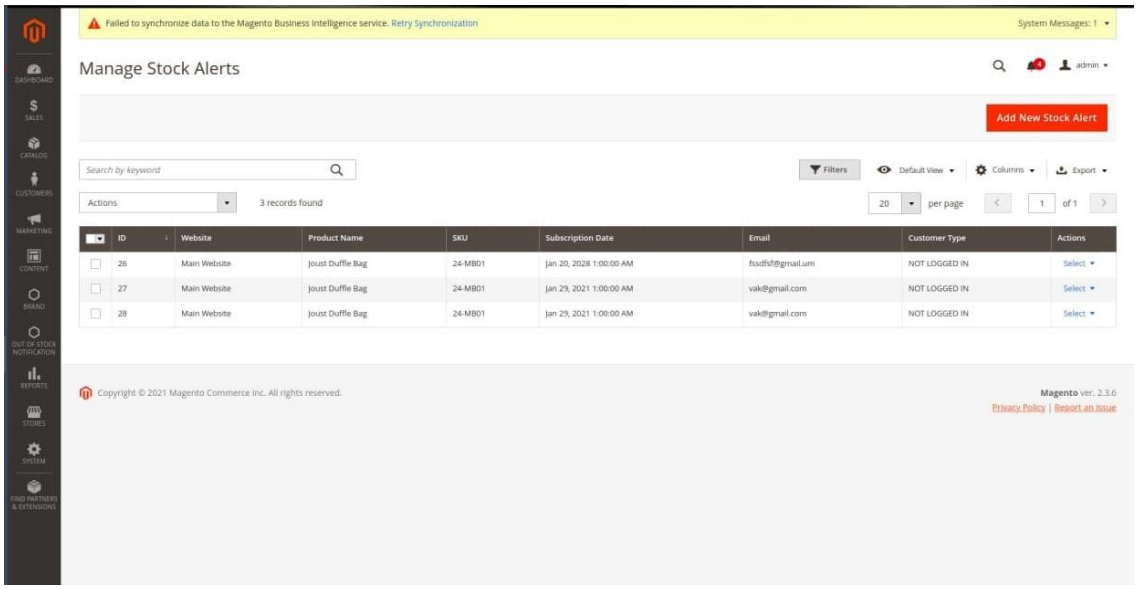

Рисунок 3.20 – Таблиця підписок на сповіщення

Також створено розділ з конфігурацією для модуля, де можна включити або відключити його, обрати шаблон для електронного листа(див. рисунок 3.21).

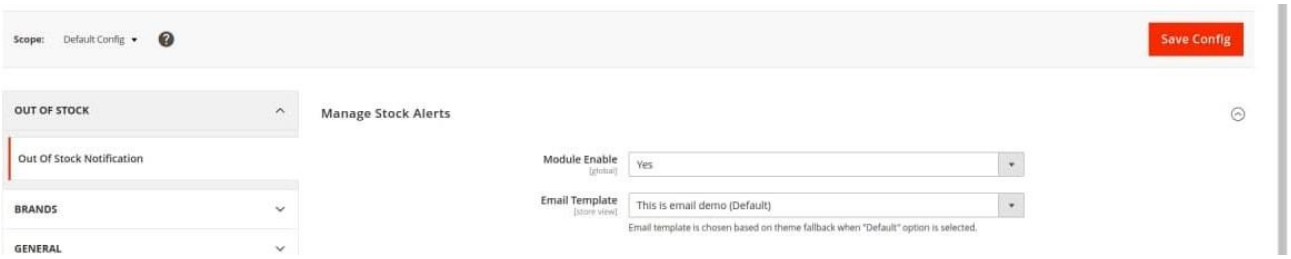

Рисунок 3.21 – Вибір шаблону для електронного листа

На стороні користувача повинна бути створена формочка для підписки на продукти, які вийшли з наявності в даному магазині(див. рисунок 3.22), та табличка для перегляду своїх підписок у профілі користувача з можливістю відписки(див. рисунок 3.23).

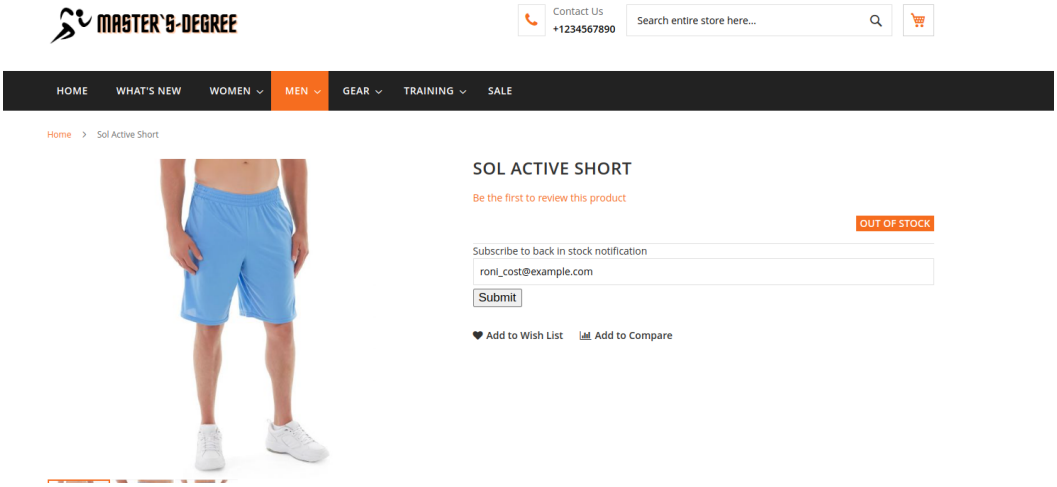

Рисунок 3.22 – Форма для підписки на продукт

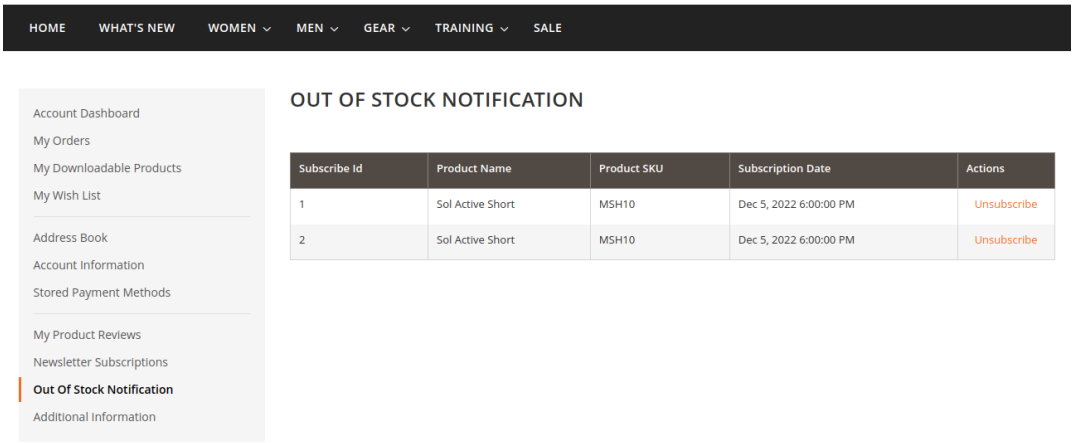

Рисунок 3.23 – Таблиця підписок у акаунті користувача

Також для акаунту користувача було добавлено функціонал для редагування фото та певної особистої інформації(див. рисунок 3.24).

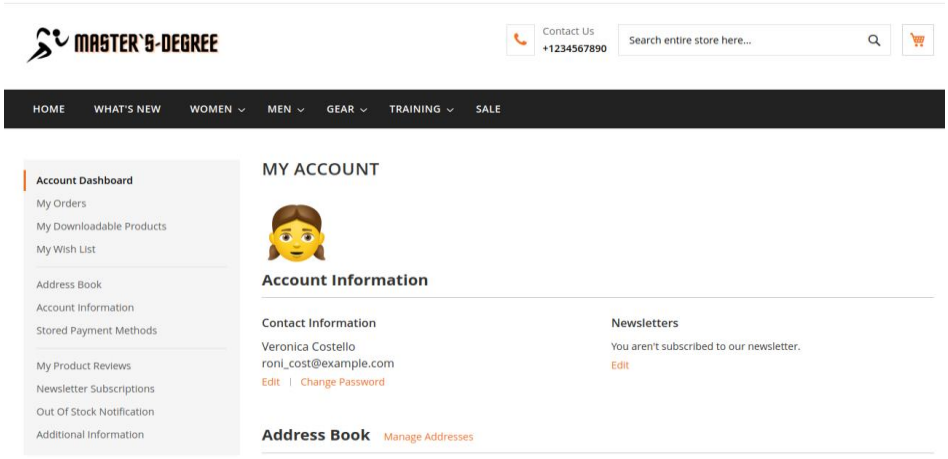

Рисунок 3.24 – Акаунт клієнта з фото

Для цього функціоналу було розроблено також можливість редагування на стороні адміністратора(див. рисунок 3.25).

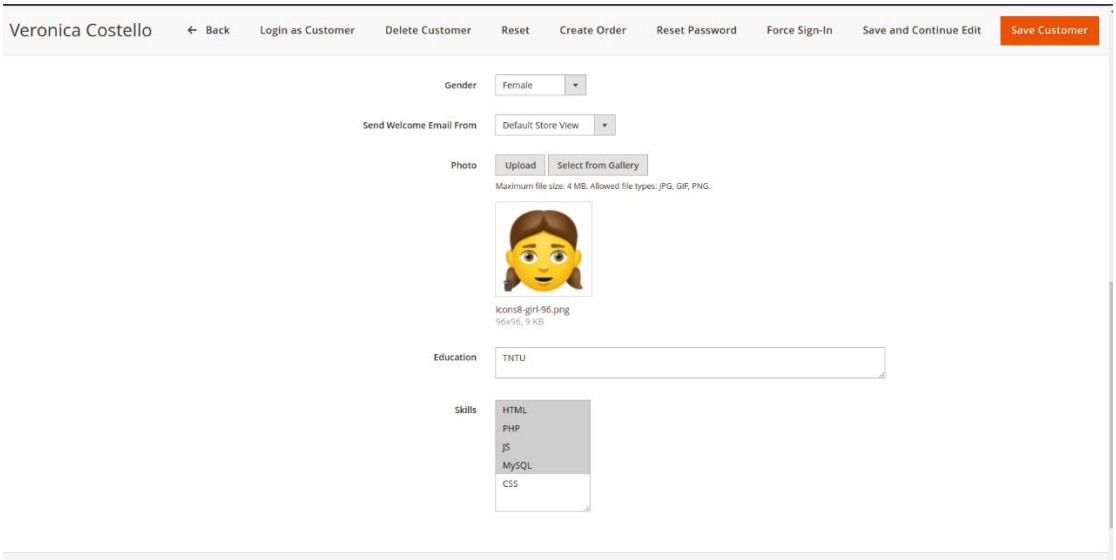

Рисунок 3.25 – Редагування користувача на стороні адміністратора

Головним функціоналом який був розроблений це інтеграція з кастомізатором продуктів. При вході на сторінку продукту необхідно відобразити

кнопку для входу в редактор та підібрати дизайн згідно нашої кольорової гами сайту(див. рисунок 3.26).

Рисунок 3.26 – Сторінка кастомізованого продуку

Сам редактор повинен мати можливість додавати зображення чи надписи(див. рисунок 3.27) на продукт, зберігати та ділитися своїми дизайнами.

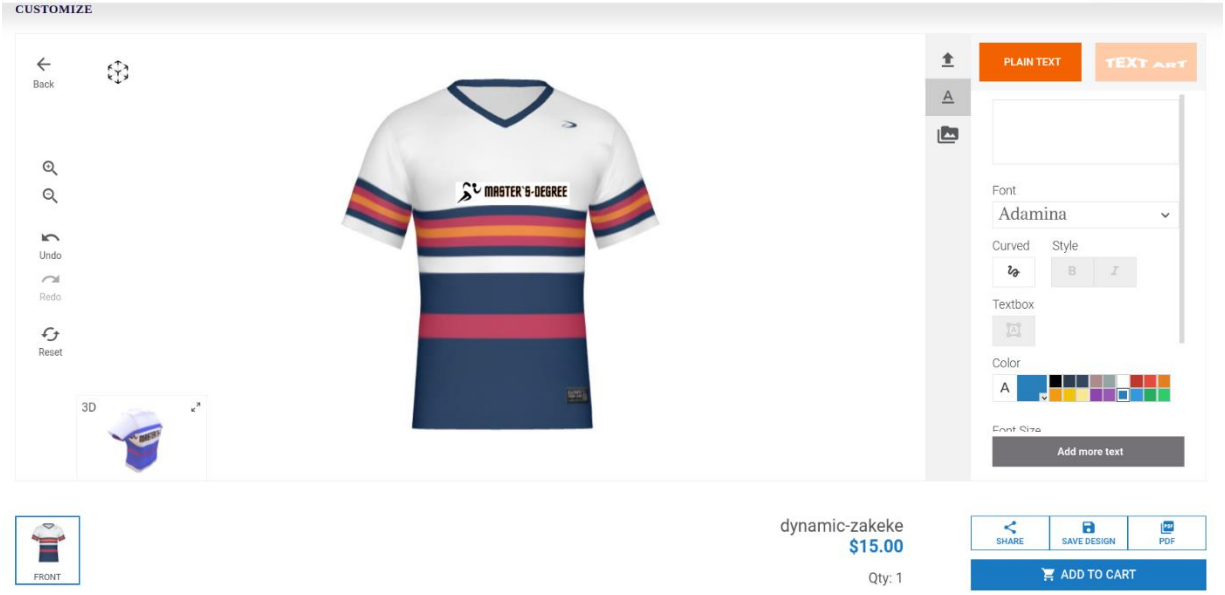

Рисунок 3.27 – Редактор продукту

Також у редакторі наявна можливість відкрити 3D модель продукту та кастомізовувати його(див. рисунок 3.28).

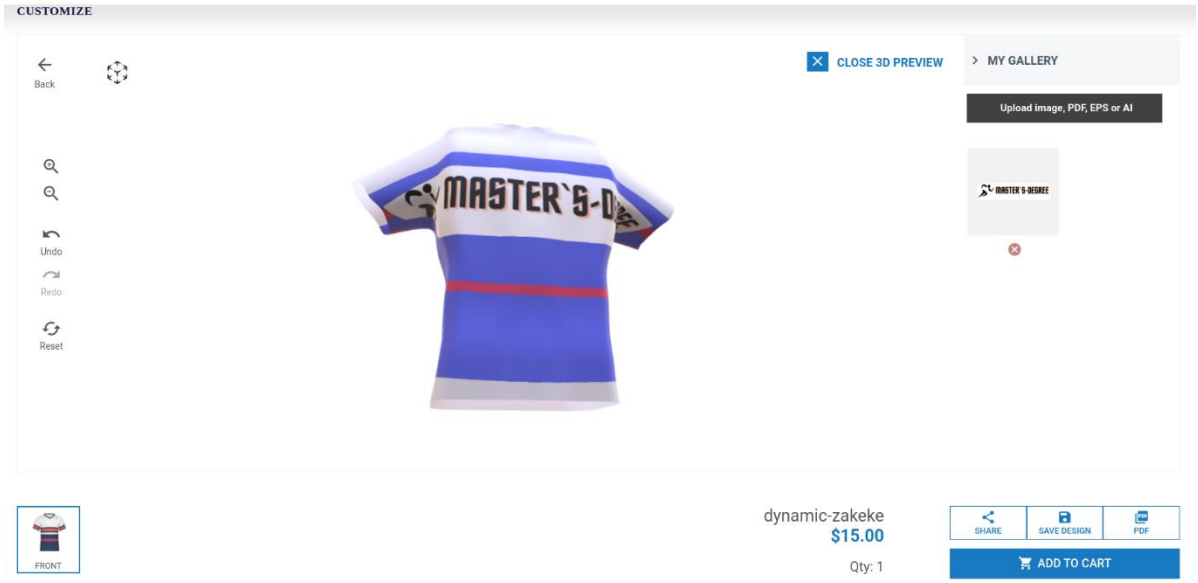

Рисунок 3.27 – 3D редактор продукту

Отже, в ході тестування не було виявлено проблем у роботі інтернет-магазину, необхідний функціонал було виконано. Особливо хорошу якість має модуль інтеграції з редактором продукту. Функціонал сповіщень про надходження продукту теж працює на достойному рівні. За рахунок кешування сторінок сайт працює швидко. При тестуванні системи проблем не виявлено.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТАЦІЯХ

4.1 Охорона праці

Зазвичай процес розробки інтернет-магазину є довготривалим та виснажливим, так як приходиться працювати з багатьма технологіями та великим обсягом даних, через це існує ймовірність припуститися помилки. Усім добре відомо, що одним із головних методів забезпечення ефективної роботи з комп'ютером є забезпечення належних умов праці, так як при недостатньому освітлені, просторі чи неякісному дисплеї, людина відчуватиме фізичний та з часом психологічний дискомфорт, який обов'язково відобразиться на результатах роботи. Для того щоб запобігти цьому, потрібно дотримуватись ряду наказів та стандартів.

В наказі № 207 від 14.02.2018 НПАОП 0.00-7.15-18 [15], якраз описується частина вимог, яких потрібно дотримуватись при роботі з екранними пристроями.

Відповідно до третьої частини наказу [15], робоче місце працівника має відповідати наступним вимогам:

1. Робоче місце має мати достатні розміри, щоб працівник мав простір для зміни робочого положення та рухів.
2. Усе випромінювання від екранних пристроїв має бути знижене до гранично допустимого рівня з погляду безпеки та охорони здоров'я працівників.
3. Усі елементи робочого місця та їх розташування мають відповідати ергономічним, антропологічним, психофізіологічним вимогам, а також характеру виконуваних робіт.
4. Освітлення робочого місця має створювати відповідний контраст між екраном і навколишнім середовищем та відповідати вимогам ДСанПІН 3.3.2.007- 98.
5. Мікроклімат виробничих приміщень має підтримуватись на постійному рівні та відповідати вимогам Санітарних норм мікроклімату виробничих приміщень ДСН 3.3.6.042-99 [16].

6. Робочий стіл чи поверхня повинні бути достатнього розміру та мати поверхню з низькою відбивною здатністю, бути гнучкою під час розміщення екрана, клавіатури, документів чи устаткування.
7. Робоче крісло має бути стійким і дозволяти працівнику легко рухатися та займати зручне положення. Сидіння має регулюватися по висоті, спинка сидіння – по висоті, та з можливістю нахилу. Для зручності слід передбачати підніжку для тих, кому це необхідно.

Також потрібно пам’ятати про безпеку, відповідно до четвертої частини наказу [15], потрібно дотримуватись наступних мінімальних вимог безпеки:

1. Перед початком роботи необхідно очищати екранні пристрої від пилу та інших забруднень.
2. Після закінчення роботи пристрої слід відключати від живлення.
3. При виникнення аварійної ситуації необхідно в той же час відключити пристрій від електричної мережі.
4. Не допускається:

   – Ремонтувати чи виконувати технічне обслуговування, і налагодження екранних пристроїв на робочому місці працівника під час роботи з екранними пристроями;

   – Вимикати захисні пристрої чи проводити зміни у конструкції та складі екранних пристроїв або їх технічне налагодження;

   – Працювати з несправними екранними пристроями, у яких під час роботи виникають нехарактерні сигнали, мигання та інші несправності.

5. Під час виконання робіт з комп’ютером, пов’язаних з нервово-емоційним напруженням мають дотримуватися оптимальні умови мікроклімату відповідно до вимог ДСН 3.3.6.042-99 [16].

Для забезпечення безпеки відповідно до п’ятої частини наказу «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями» [15], усі екранні пристрої повинні відповідати наступним мінімальним вимогам безпеки:

Відповідно до третьої частини наказу [15], робоче місце працівника інтернетмагазину, який працює над контентним наповненням сайту має відповідати наступним вимогам:

1. Екранні пристрої не мають бути джерелом ризику для працівників.
2. Усе випромінювання має бути зведене до мінімального рівня з погляду безпеки і охорони здоров’я працівників.
3. Символи на дисплеї мають бути чіткими, відповідного розміру. Між символами і рядками символів повинна бути правильна відстань.
4. Зображення на дисплеї має бути стабільним, без миготінь або інших видів несправності.
5. Яскравість та контрастність символів має легко регулюватися, а також швидко адаптуватися до навколишніх умов.
6. Під час вибору монітора, слід надавати перевагу тим пристроям, які мають можливості повороту та нахилу екрану.
7. При потребі монітор може бути закріпленим на окремому столі чи підставці.
8. При виборі монітора надавайте перевагу дисплеям з матовим покриттям, щоб мінімізувати відблискування або відбивання світла.
9. При виборі клавіатури, слід надавати перевагу тій, яка відкидається і є автономною, щоб працівник міг вибрати зручну робочу позу й уникнути втоми рук.
10. Поверхня клавіатури має бути матовою, щоб уникнути віддзеркалювання.
11. Устаткування, яке входить до робочої станції, не повинно виділяти надлишкового тепла.
12. Під час розробки, вибору, замовлення та модифікації програмного забезпечення, а також під час розробки завдань, що передбачають використання устаткування з екранними пристроями, роботодавець має керуватися таким програмним забезпеченням, яке відповідає розв’язуваним завданням і є простим у використанні, а де необхідно - адаптованим до рівня знань і досвіду працівника.

Отже, для безпечної та ефективної роботи контент-менеджера інтернетмагазину забезпечено належні умови праці, починаючи від робочого місця та його оснащення, та закінчуючи мікрокліматом робочого середовища, відповідно до вимог чинного законодавства.

4.2 Безпека в надзвичайних ситуаціях

Ефективність роботи людини з комп'ютером значною мірою визначається функціональним станом людини. Психофізіологічні та емоційні перенапруження, втома людини-оператора можуть призвести в комп'ютеризованих системах керування до помилок і як наслідок – до значних економічних втрат.

Згідно зі статистичними даними від 40 до 75% аварій літаків зумовлено людським фактором [17]. Відмови комп'ютеризованої системи керування рухом залізничного транспорту, на гірничо-збагачувальних комбінатах з вини операторів становлять понад 50% їх загальної кількості, причому значна їх частина спричинена невідповідністю функціонального стану оператора складності виконуваної роботи.

Трудова діяльність користувачів комп'ютерів відбувається у певному виробничому середовищі, яке впливає на їх функціональний стан. Найбільш значимі – фізичні фактори виробничого середовища, до яких належать електромагнітні хвилі різних частотних діапазонів, електростатичні поля, шум, параметри мікроклімату та ціла низка світлотехнічних показників.

Трудовий процес суттєво впливає на психофізіологічні можливості користувачів комп'ютерів, оскільки їх діяльність характеризується значними статичними фізичними навантаженнями; недостатньою руховою активністю; напруженнями сенсорного апарату, вищих нервових центрів, які забезпечують функції уваги, мислення, регуляції рухів. Окрім того, трудовий процес

користувачів комп’ютерів відзначається значними інформаційними навантаженнями.

Професійні якості та виробничий досвід, які визначають внутрішні засоби діяльності, обумовлюють надійну та безпомилкову діяльність користувачів комп’ютерів, дозволяють знаходити безпечні методи розв’язання виробничих завдань навіть у нестандартних ситуаціях.

Зовнішні засоби діяльності, які в основному визначаються ергономічними показниками щодо організації робочого місця, формою та параметрами його елементів, просторового розташування основного і допоміжного устаткування, можуть суттєво знизити фізичні та психофізіологічні навантаження, що діють на користувачів комп’ютерів.

Оскільки робота користувачів комп’ютерів найчастіше проходить за активної взаємодії з іншими людьми, то виникають питання раціоналізації міжособистісних стосунків. Цей комплекс питань порушує як психологічні, так і соціально-психологічні аспекти трудових взаємовідносин, які також є факторами "ризику", що відчутно впливають на функціональний стан користувачів комп’ютерів.

Визначення та вивчення факторів, що впливають на функціональний стан користувачів комп'ютерів дозволить виділити основні причини виникнення станів напруженості, стомлення, стресу і здійснити відповідні профілактичні заходи.

Отже, до основних факторів, що впливають на функціональний стан користувачів комп’ютера належать:

1. середовище – характеризується такими шкідливими факторами:
   1.2 фізичні: електромагнітні хвилі різних частотних діапазонів, електростатичні поля, шум, параметри мікроклімату та ряд світлотехнічних показників;
   1.3 хімічні: пил, шкідливі хімічні речовини, які виділяються при роботі принтера і копіювальної техніки;
   1.4 біологічні: підвищений вміст в повітрі патогенних мікроорганізмів, особливо у приміщенні з великою кількістю працюючих, при недостатній вентиляції, особливо у період епідемій;

1.5 психофізіологічні: напруження зору та уваги, інтелектуальні та емоційні навантаження, тривалі статичні навантаження і монотонність праці.

2. трудовий процес - характеризується значними статичними фізичними навантаженнями; недостатньою руховою активністю; напруженнями сенсорного апарату, вищих нервових центрів, які забезпечують функції уваги, мислення, регуляції рухів. Окрім того, трудовий процес користувачів комп'ютерів відзначається значними інформаційними навантаженнями;
3. внутрішні засоби діяльності – це професійні риси та виробничий досвід, які обумовлюють надійну та безпомилкову діяльність користувачів комп'ютерів, дозволяють знаходити безпечні методи розв'язання виробничих завдань навіть у нестандартних ситуаціях;
4. зовнішні засоби діяльності - визначаються ергономічними показниками щодо організації робочого місця, форми та параметрів його елементів, просторового розташування основного і допоміжного устаткування, які можуть суттєво знизити фізичні та психофізіологічні навантаження, що діють на користувачів комп'ютерів;
5. соціально-психологічні фактори трудових взаємовідносин.

У професійних операторів частіше зустрічаються порушення органів зору, опорно-рухового апарату, центральної нервової, серцево-судинної, імунної та статевої систем, захворювання шкіри. Зафіксована значна кількість скарг операторського персоналу на загальне недомагання, передчасне стомлювання, головний біль, порушення функцій органів зору, які здійснювали несприятливий психофізіологічний вплив на самопочуття та працездатність операторів.

Сучасна професія користувача ВДТ належить до розумової праці, яка характеризується: високою напруженістю зорових функцій; одноманітною позою; великою кількістю стереотипних висококоординованих рухів, що виконуються лише м'язами кистей рук на фоні малої загальної рухової активності; значним нервовоемоційним компонентом, особливо в умовах дефіциту часу; роботою з великими масивами інформації, що викликає активізацію уваги та інших вищих

психічних функцій. Крім того, при роботі з дисплеями на електронно-променевих трубках виникає вплив на користувача цілої низки факторів фізичної природи — електростатичні поля, радіочастотне та рентгенівське випромінювання тощо.

Діяльність професіоналів можна поділити на три групи:

1. Діяльність, яка пов'язана з виконанням нескладних багаторазово повторюваних операцій, що не вимагають великого розумового напруження. Наприклад, робота операторів комп'ютерного набору, працівників довідкових служб.
2. Діяльність, яка пов'язана із здійсненням логічних операцій, що постійно повторюються. Це робота інженера-економіста, інженера-проектувальника, оператора автоматизованого виробництва.
3. Діяльність, коли в процесі роботи необхідно приймати рішення за відсутності заздалегідь відомого алгоритму. Наприклад, робота інженера програміста, диспетчерів руху залізничного транспорту, аеропортів тощо.

У користувачів, які інтенсивно використовують комп'ютер в умовах значних розумових напружень досить часто (40—70%) виникають психологічні та поведінкові порушення (нервозність, роздратування, тривога, нерішучість, замкнутість тощо). Серед користувачів ВДТ в США і Європі значного поширення набуло специфічне захворювання, яке отримало назву синдром комп'ютерного стресу (СКС). СКС супроводжується головним болем, запаленням очей, алергією, роздратованістю, млявістю і депресією. Інформаційне перевантаження користувачів ВДТ супроводжується низкою специфічних захворювань, які називають інформаційними. Першим симптомом їх є головний біль. Дослідження, проведені в США, Німеччині, Швейцарії та інших країнах, показали, що робота з обслуговування ВДТ супроводжується підвищеним напруженням зору, інтенсивністю і монотонністю праці, збільшенням статичних навантажень, нервово-психічним напруженням, впливом різного виду випромінювань та ін. Внаслідок цього серед операторів ВДТ, як зазначають фахівці Всесвітньої організації охорони здоров'я, частіше, ніж в інших групах працюючих, трапляються такі професійні захворювання, як передчасна стомлюваність, погіршення зору,

м'язові і головні болі, психічні й нервові розлади, хвороби серцево-судинної системи, онкологічні захворювання та ін. Вважається, що стан організму операторів ВДТ визначається комплексним впливом факторів трудового процесу і середовища, значення яких є неоднаковим. На операторів з малим стажем роботи на ВДТ домінуючий вплив чинять фактори середовища, а на операторів зі стажем понад 5 років - фактори трудового процесу.

Комп'ютерний зоровий синдром (КЗС) - комплекс порушень здоров'я, який може виникати у користувачів персональних комп'ютерів (ПК) [18]. У користувачів ПК дуже поширені кон'юнктивіти і блефарити, патогенетично пов'язані з КЗС. Синдром розвивається при умові, що робоче місце організовано неправильно - у користувача незручне крісло, відсутні пюпітри для паперів, підставки для ніг та кистей рук, не встановлена висота і нахил монітора відносно очей, відстань від очей до екрана. За таких умов тіло людини при роботі займає вимушене положення: спина статично напружена, шия витягнута, плечі жорстко фіксовані. Напружені м'язи погіршують кровотік у сонних артеріях, а недостатнє кровозабезпечення головного мозку веде до очманіння, появи головного болю. На фоні шийного остеохондрозу з'являється відчуття випирання очних яблук, туману в очах, мушок та райдужних кіл у полі зору. Розвитку КЗС сприяє поганий мікроклімат приміщення, значна загальна іонізація та мікробне забруднення, а також куріння.

Національною радою з наукових досліджень США для стану зорового дискомфорту був уведений термін "астенопія", який означає "будь-які суб'єктивні зорові симптоми чи емоційний дискомфорт, що є результатом зорової діяльності". Симптоми астенопії були класифіковані на "очні" (біль, печія та різь в очах, почервоніння повік та очних яблук, ломота у надбрівній частині тощо) та "зорові" (пелена перед очима, мерехтіння, швидка втома під час зорової роботи та ін.).

Таким чином, на користувача комп'ютера впливає комплекс факторів. Урахування ступеня та якості впливу цих факторів на функціональний стан дозволяють розробити заходи та засоби щодо забезпечення безпеки, підвищення працездатності та збереження здоров'я користувачів комп'ютерів.

ВИСНОВКИ

В ході роботи над кваліфікаційною роботою було досліджено CRM Magento 2, створено діаграми класів та діаграму варіантів використання. Для розробленого магазину було спроектовано та розроблено декілька модулів для покращення взаємодії користувача з сайтом. При тестуванні системи було встановлено, що якість розробленої системи доволі висока. Також в ході дослідження було виявлено, що платформа Magento 2 в більшості випадків підходить для розгортання саме великої мережі магазинів, що являється як і перевагою цього CRM, так і його недоліком у порівнянні з менш затратними системами для створення маленьких інтернет-магазинів. Якщо на вашому сайті потрібен лише один магазин з кількістю товарів у районі 100, то краще використовувати інші системи, наприклад, WordPress чи Joomla. Також одним із недоліків Magento 2 може бути відносна важкість у вивчені цієї технології, адже у цій платформі об’єднана велика кількість технологій і для написання хорошого програмного забезпечення потрібна практика і вивчення структури та можливостей CRM. Хоча в той же час при розробці і вивчені можна виявити велику кількість переваг. Архітектура Magento 2 гнучка, структура програми модульна, що дозволяє легко додавати новий функціонал та тестувати її. Також архітектура бази даних без проблем дозволяє додавати нові сутності у систему та змінювати вже існуючі. Це також спрощує розробку у великих командах розробників. При використанні технології PWA для розробки графічного інтерфейсу користувача система працює значно швидше та має більш чіткий розділ між фронтенді і бекенд розробкою, що значно спрощує розподіл роботи між спеціалістами.

В нас час розроблений продукт є доволі актуальним і корисними для розвитку свого бізнесу, оскільки останні роки в Україні склалися сприятливі передумови для розвитку електронної комерції, це і є основним чинником важливості цієї роботи. Розвиток власного інтернет-магазину допоможе злегкістю відкрити свій бізнес, це також дає можливість поширювати свої товари за межі країни, що значно збільшує

клієнтську базу. Такий хід спростить вибір і покупку різних товарів, які важко знайти у звичайних магазинах для багатьох людей. Мінусом роботи на даний момент є її вузьконаправленість. Даний проект розрахований на українську аудиторію, це обмежує можливість заробітку на даний момент. Хоча навіть з такими недоліками розроблене програмне забезпечення сильно вирізняється серед інших подібних собі за рахунок декількох розроблених модулів, які відсутні у інших системах.

Очікується, що цей продукт вирветься на високі позиції на ринку України та в подальшій перспективі зможе приносити не тільки матеріальну вигоду власнику, але і спростить життя людям, яких цікавить продукція магазину. В межах України не велика кількість спортивних торгових площадок, що робить розроблений інтернет-магазин цінним та корисним у наш час.

В подальших дослідженнях можна глибше ознайомитися з технологією PWA. Також є висока ймовірність, що нові версії системи Magento 2 повністю перейдуть на використання цієї технології. Тому доцільним буде в майбутньому оновити графічний інтерфейс користувача з використанням PWA. Також можна розширити даний сайт і розмістити на ньому декілька магазинів для виходу на міжнародний рівень продажів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. М.Р. Петрик, Д.М. Михалик, О.Ю. Петрик, Г.Б. Цуприк. Методичні вказівки до виконання атестаційної роботи магістра за спеціальністю 121 – “Інженерія програмного забезпечення” для усіх форм навчання [Текст] – Тернопіль : Тернопільський національний технічний університет імені Івана Пулюя – 2020 – 27 с.
2. Інформаційні технології видобутку даних (Data mining, високопродуктивні обчислення у складних системах): навчальний посібник ІВ Бойко, МР Петрик, Г Цуприк – 2020
3. Петрик М.Р. Проєктування програмного забезпечення на основі аналізу вимог та інструментальних засобів розробки IBM Rational Software Architect (від Вимог до коду) Науково-методичний посібник. Тернопіль: Вид-во ТНТУ ім. Івана Пулюя.-2022.- 560с.
4. Magento 2 Developer Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://devdocs.magento.com.
5. Magento 2 Certified Professional Developer Guide [Електронний ресурс] – Режим доступу до ресурсу: https://belvg.com/tutorial/magento-2-certified-professional-developer-guide-section-1.
6. Пасічник О. Г. Основи веб-дизайну [Текст]: навч. посібн. / О. Г. Пасічник, О. В. Пасічник, І. В. Стеценко. – К. : Вид. група BHV, 2009 – 336 с.
7. Пюрівал Сєммі. Основи розробки веб-додатків [Текст] / Пюрівал Сєммі. – СПб. : Питер, 2015. – 272 с.
8. Ніксон Робін. Створюємо динамічні веб-сайти з допомогою PHP, MySQL, JavaScript, CSS и HTML5 [Текст] / Робін Ніксон. – СПб. : Санкт-Петербург, 2016. – 768 с.
9. Колісніченко Денис. PHP и MySQL. Розробка Web-додатків [Текст] / Денис Колісніченко. – СПб. : БХВ-Петербург, 2015. – 593 с.

10. Хоган Б. HTML5 і CSS3. Веб-розробка по стандартам нового покоління [Текст] / Б. Хоган. – СПб. : Київ, 2014. –320 с.
11. Stefanov Stoyan. JavaScript for PHP Developers [Текст] / Stoyan Stefanov. – O'Reilly, 2013. – 144 с.
12. Фленаган Девід. JavaScript. Детальний посібник, 5-е видавництво [Текст] / Девід Фленаган. – М. : Символ-Плюс, 2009. – 992 с. 57
13. Prettyman Steve. Learn PHP 7: Object Oriented Modular Programming using HTML5, CSS3, JavaScript, XML, JSON, and MySQL [Текст] / Steve Prettyman. – Apress, 2015. – 316 с.
14. Пасічник В. В. Веб-технології [Текст]: підруч. / В. В. Пасічник, О.В. Пасічник, Д. І. Угрин. – Львів : Магнолія 2013. – 336 с.
15. Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/z0508-18#Text.
16. Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99 [Електронний ресурс]. – 1999. – Режим доступу до ресурсу: https://zakon.rada.gov.ua/rada/show/va042282-99#Text.
17. Основні причини аварій літаків [Електронний ресурс]. – 2020. – Режим доступу до ресурсу: https://suspilne.media/7904-letiti-bez-strahu-vidpovidaemo-na-najposirenisi-zapitanna-pro-aviakatastrofi/
18. Комп'ютерний зоровий синдром [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://linza.com.ua/uk/articles/blog/kompyuternyy-zritelnyy-sindrom-simptomy-i-lechenie/.

ДОДАТКИ

Додаток А

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

ІМЕНІ ІВАНА ПУЛЮЯ

КАФЕДРА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку проекту

«Розробка web-сайту інтернет-магазину для продажу спортивного одягу на базі eCommerce платформи Magento 2»

Виконавець:

ст. гр. СПм-61

Простяк В.М.

_____

Керівник проекту:

Цуприк Г.Б

_____

Тернопіль 2022

ЗМІСТ

1. ПІДСТАВИ ДО РОЗРОБКИ

Розробка проводиться у відповідності до графіку навчального плану підготовки магістрів за спеціальністю 121 «Інженерія програмного забезпечення», та згідно наказу на виконання дипломної роботи студента-магістра.

Тема проекту: «Розробка web-сайту інтернет-магазину для продажу спортивного одягу на базі eCommerce платформи Magento 2».

Термін виконання: до __.___._____р.

2 ПРИЗНАЧЕННЯ ПРОГРАМНОГО ПРОДУКТУ

Програмний продукт призначений для продажу товарів у мережі інтернет. Розроблена система буде корисною для тих, хто хоче розповсюдити свій товар на велику аудиторію осіб та збільшити кількість покупців. Даний інтернет-магазин дозволить значно покращити процес купівлі та продажу за рахунок розробленого функціоналу підписок та інтеграцією з редактором продуктів. Користувачі з легкістю зможуть розробляти свої дизайни та купляти саме такі речі, які їх цікавлять.

3 ВИМОГИ ДО ПРОГРАМНОГО ПРОДУКТУ

3.1Функціональні вимоги

У системі повинні бути реалізовані наступні можливості:

- Підписка на продукт, який вийшов з наявності;
- Моніторинг підписок;
- Видалення підписок;

- Можливість редагування дизайну продукту;
- Збереження власних дизайнів;
- Розширеня функціоналу акаунту користувача;
- ACL.

Для інтеграції з кастомізатором продукту повинно бути реалізоване REST API.

3.2 Технічні вимоги

Вимоги до серверної частини:

- ОС Linux;
- не менше ніж 16Гб ОЗП;
- SSD 240 GB.

Вимоги до клієнтської частини:

- iOS/Android/Windows/macOS.

Додаткові вимоги:

- наявне підключення до мережі Інтернет.

3.3 Програмні вимоги

БД: MySQL 5.6, 5.7.

Версія PHP: від 7.3.

Версія CRM: Magento 2.4.3.

Інструменти для управління контейнерами: Docker/LXC.

4 ЕТАПИ РОЗРОБКИ

Розробка інформаційної системи проводиться в наступному порядку:

- аналіз предметної області, виявлення акторів та варіантів використання системи;
- вибір засобів розробки та архітектури системи;
- проектування архітектури;
- розробка програмного забезпечення системи;
- тестування інформаційної системи на реальних даних;
- оформлення супровідної документації;
- здача проекту.

Результати виконання кожного етапу проекту погоджуються з керівником проекту.

5 СУПРОВІДНА ДОКУМЕНТАЦІЯ

Для інформаційної системи повинні бути розроблені наступні документи:

- пояснювальна записка до проекту;
- презентація проекту;
- рецензія на проект;
- диск з проектом.

Пояснювальна записка до проекту оформляється згідно діючих вимог до нормоконтролю проектів.

6 ПОРЯДОК ЗДАЧІ ПРОЕКТУ

Розроблена система повинна відповідати вимогам, що складаються з перерахованих у п.3.1 цього документу характеристик.

Для здачі проекту необхідно підготувати весь перелік документів зазначений у п.5 цього документу.

Технічне завдання оформляється у відповідності з загальними вимогами до текстових конструкторських документів за ГОСТ 2.105-95 на аркушах формату А4.

Приймання проекту проводиться спеціально створеною комісією в термін зазначені в п.1 цього документу.

7 ВІДМІТКИ ПРО ВИКОНАННЯ ЕТАПІВ ТА ЗМІНИ В ПРОЕКТІ

Назва етапу	Відмітка*
Аналіз предметної області	
Архітектура системи	
Проектування база даних	
Використання системи	
Супровідна документація	

* відмітки про виконання етапу ставляться керівником проекту

Додаток Б
Тези

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И
X НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

7–8 грудня 2022 року

ТЕРНОПІЛЬ
2022

УДК 004.41

В. Простяк
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

РОЗРОБКА WEB-САЙТУ ІНТЕРНЕТ-МАГАЗИНУ ДЛЯ ПРОДАЖУ СПОРТИВНОГО ОДЯГУ НА ОСНОВІ MAGENTO 2

UDC 004.41

V. Prostiak

WEB SITE DEVELOPMENT OF AN INTERNET STORE FOR THE SALE OF SPORTS CLOTHING BASED ON MAGENTO 2

Ключові слова: Magento 2, eCommerce, Zend Framework, менеджер об'єктів, одинак, ін’єкція залежностей, фабрика, об'єктно-орієнтований підхід (ООП)
Key word: Magento 2, eCommerce, Zend Framework, object manager, singleton, dependency injection, factory, object-oriented approach

Останніми роками в усьому світі все більшого поширення набувають методи ведення бізнесу в Інтернеті, зокрема. інтернет-торгівля. Застосування Інтернету дозволяє швидко і з незначними витратами вивести і просувати продукцію на національний і міжнародні ринки. Торгівля через Інтернет дозволяє істотно знизити вартість продукції, оскільки відпадають потреби в утриманні торгових площ, придбанні торгівельного обладнання тощо. Крім того, споживач може швидко знайти потрібний товар, вияснити його характеристики, ознайомитися з відгуками інших споживачів, обрати зручний спосіб і час доставки товару, провести платежі через Інтернеті і т. ін.

З роками розробка інтернет-магазинів зазнала значних змін. Через велику популярність цієї галузі почало з'являтися багато платформ на основі яких програмісти могли б легко створити унікальний інтернет-магазин. Великим проривом у цій сфері стала eCommerce платформа з відкритим кодом Magento.

Magento – це програмне забезпечення створене з використанням Zend Framework. Перша версія була випущена 31 березня 2008 року. Через велику кількість недоліків та вразливостей систему постійно оновлювали та у 2015 році відбувся випуск бета-версії Magento 2.

Magento 2 має багато нових і покращених функцій, інструментів розробника, а його архітектура значно відрізняється від усіх попередніх версій. Одною з основних відмінностей стало те, що у Magento 2 відмовилися від використання менеджера об'єктів та одинака, натомість у нових версіях використовується ін’єкція залежностей та фабрики для створення об’єктів. Для написання back-end частини сайту використано мову програмування php та об'єктноорієнтований підхід (ООП) в програмуванні.

Платформа надає можливість для різного роду ведення продаж, кількість магазинів легко розширити, тематику магазину можна швидко змінити відповідно до вимог замовника, система працює швидко та надійно.

Література

1. Докуменатція Magento 2. URL: https://devdocs.magento.com/.

Додаток В

Діаграми активностей

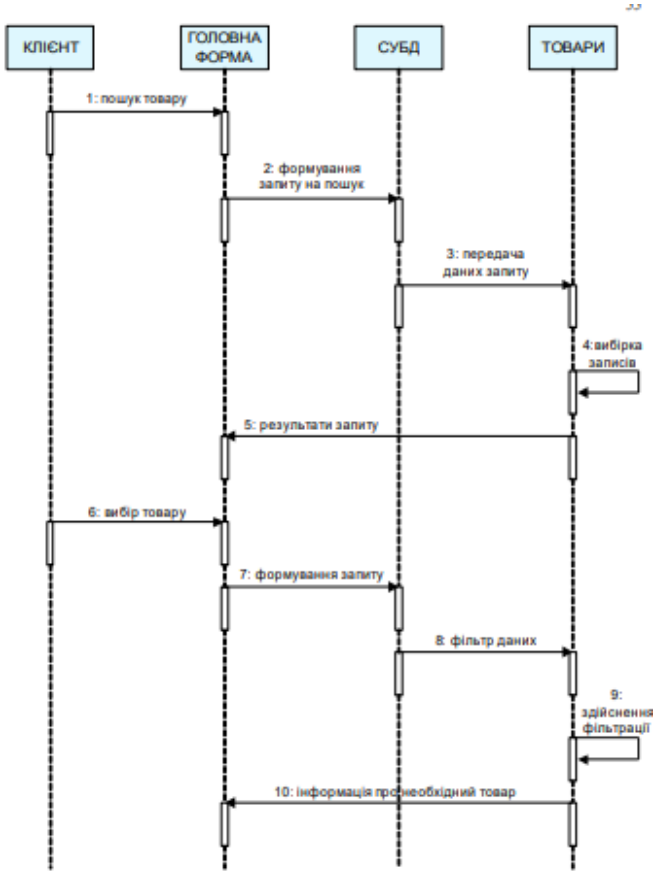

Рисунок В.1 – Діаграма активності для варіанту використання «Пошук товару»

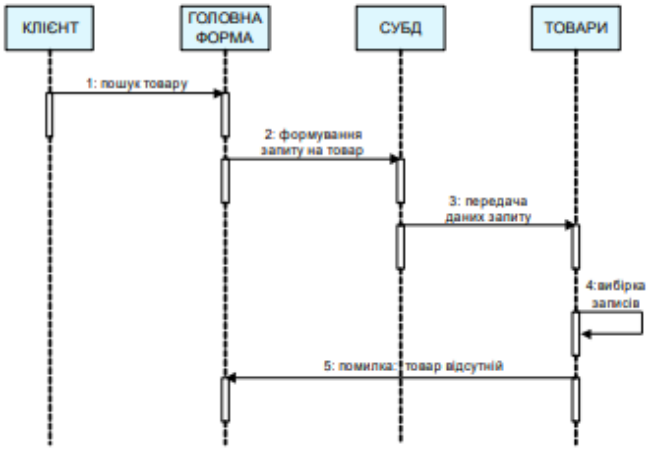

Рисунок В.2 – Діаграма активності для альтернативного потоку варіанта використання «Пошук товару»

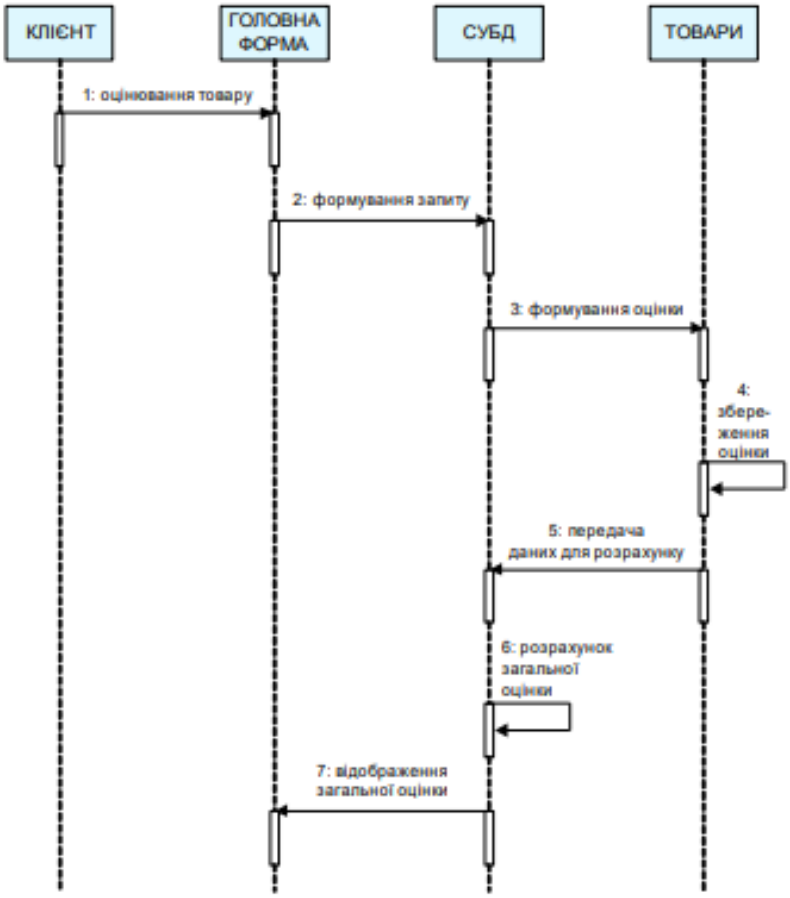

Рисунок В.3 – Діаграма активності для варіанту використання «Оцінювання товару»

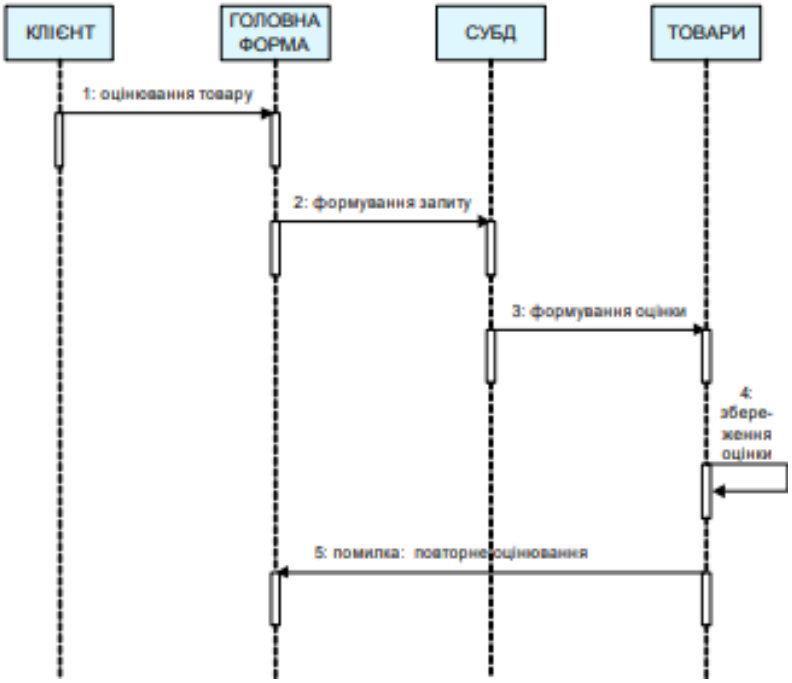

Рисунок В.2 – Діаграма активності для альтернативного потоку варіанта використання «Оцінювання товару»

Додаток Г
Фрагмент коду модуля “Out of Stock Notification”

Інтерфейс моделі

```
<?php

namespace Volodymyr\OutOfStock\Api\Data;

interface NotificationInterface
{
   /**
    * Get customer id
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function getSubscribeId();
   /**
    * Set customer id
    * @param int $subscribeId
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function setSubscribeId($subscribeId);
   /**
    * Get product id
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function getProductId();
   /**
    * Set product id
    * @param int $productId
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function setProductId($productId);
   /**
    * Get customer id
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function getCustomerGroupId();
   /**
    * Set customer group id
    * @param int $customerGroupId
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function setCustomerGroupId($customerGroupId);
   /**
    * Get customer email
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function getCustomerEmail();
   /**
    * Set customer email
```

```
    * @param string $customerEmail
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function setCustomerEmail($customerEmail);
   /**
    * Get customer subscription date
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function getSubscriptionDate();
   /**
    * Set customer subscription date
    * @param string $subscriptionDate
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function setSubscriptionDate($subscriptionDate);
}
```

Інтерфейс репозиторія

```
<?php

namespace Volodymyr\OutOfStock\Api;

use Magento\Framework\Api\SearchCriteriaInterface;
use Volodymyr\OutOfStock\Api\Data\NotificationInterface;

interface NotificationRepositoryInterface
{
   /**
    * @param int $subscribeId
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    * @throws \Magento\Framework\Exception\NoSuchEntityException
    */
   public function getById($subscribeId);
   /**
    * @param \Volodymyr\OutOfStock\Api\Data\NotificationInterface $notification
    * @return \Volodymyr\OutOfStock\Api\Data\NotificationInterface
    */
   public function save(NotificationInterface $notification);
   /**
    * @param \Volodymyr\OutOfStock\Api\Data\NotificationInterface $notification
    * @return void
    */
   public function delete(NotificationInterface $notification);
   /**
    * @param int $subscribeId
    * @return void
    * @throws \Magento\Framework\Exception\NoSuchEntityException
    */
   public function deleteById($subscribeId);
}
```

Контролер для масового видалення записів таблиці

```
<?php

namespace Volodymyr\OutOfStock\Controller\Adminhtml\MassActions;

use Magento\Backend\App\Action\Context;
use Volodymyr\OutOfStock\Model\ResourceModel\Notification\Grid\CollectionFactory;
use Magento\Ui\Component\MassAction\Filter;
use Magento\Framework\Controller\ResultFactory;
use Volodymyr\OutOfStock\Api\NotificationRepositoryInterface;
use Magento\Backend\App\Action;

/**
 * Class MassDelete
 *
 * @package Volodymyr\OutOfStock\Controller\Adminhtml\MassActions
 */
class MassDelete extends Action
{
   const ADMIN_RESOURCE = 'Volodymyr_OutOfStock::stock_alerts';
   /**
    * @var NotificationRepositoryInterface
    */
   protected $notificationRepository;
   /**
    * @var Filter
    */
   protected $filter;
   /**
    * @var NotificationFactory
    */
   protected $notificationFactory;

   /**
    * @param Context $context
    * @param Filter $filter
    * @param NotificationFactory $notificationFactory
    */
   public function __construct(
      Context $context,
      Filter $filter,
      CollectionFactory $notificationFactory,
      NotificationRepositoryInterface $notificationRepository
   ) {
      $this->notificationRepository = $notificationRepository;
      $this->filter = $filter;
      $this->notificationFactory = $notificationFactory;
      parent::__construct($context);
```

```
    }

    /**
     * @inheritDoc
     */
    public function execute()
    {
        $collection = $this->filter->getCollection($this->notificationFactory->create());
        $collectionSize = $collection->getSize();
        foreach ($collection as $item) {
            $this->notificationRepository->deleteById($item->getSubscribeId());
        }

        $this->messageManager->addSuccess(__('A total of %1 record(s) have been deleted.', $collectionSize));

        /** @var \Magento\Backend\Model\View\Result\Redirect $resultRedirect */
        $resultRedirect = $this->resultFactory->create(ResultFactory::TYPE_REDIRECT);
        return $resultRedirect->setPath('*/outofstocknotification/notificationgrid');
    }
}
```

Спостерігач наявності продуктів

```
<?php

namespace Volodymyr\OutOfStock\Observer;

use Magento\Framework\Event\ObserverInterface;
use Magento\Framework\Event\Observer;
use Volodymyr\OutOfStock\Helper\Email;
use Volodymyr\OutOfStock\Model\NotificationFactory;
use Volodymyr\OutOfStock\Model\ResourceModel\Notification\Grid\CollectionFactory;
use Magento\Framework\App\Request\Http;
use Volodymyr\OutOfStock\Api\NotificationRepositoryInterface;

/**
 * Class StockObserver
 *
 * @package Volodymyr\OutOfStock\Observer
 */
class StockObserver implements ObserverInterface
{
    /**
     * @var \Volodymyr\OutOfStock\Helper\Email
     */
    protected $sendEmail;
    /**
     * @var \Volodymyr\OutOfStock\Model\NotificationFactory
     */
    protected $notificationFactory;
```

```
    /**
     * @var \Volodymyr\OutOfStock\Model\ResourceModel\Notification\Grid\CollectionFactory
     */
    protected $collectionFactory;
    /**
     * @var \Magento\Framework\App\Request\Http
     */
    protected $request;
    /**
     * @var \Volodymyr\OutOfStock\Api\NotificationRepositoryInterface
     */
    protected $notificationRepository;

    /**
     * StockObserver constructor.
     *
     * @param \Volodymyr\OutOfStock\Helper\Email                                    $sendEmail
     * @param \Volodymyr\OutOfStock\Model\NotificationFactory                       $notificationFactory
     * @param \Volodymyr\OutOfStock\Model\ResourceModel\Notification\Grid\CollectionFactory $collection
Factory
     * @param \Magento\Framework\App\Request\Http                                   $request
     * @param \Volodymyr\OutOfStock\Api\NotificationRepositoryInterface             $notificationRepository
     */
    public function __construct(
        Email $sendEmail,
        NotificationFactory $notificationFactory,
        CollectionFactory $collectionFactory,
        Http $request,
        NotificationRepositoryInterface $notificationRepository
    ) {
        $this->sendEmail = $sendEmail;
        $this->notificationFactory = $notificationFactory;
        $this->collectionFactory = $collectionFactory;
        $this->request = $request;
        $this->notificationRepository = $notificationRepository;
    }

    /**
     * @param \Magento\Framework\Event\Observer $observer
     *
     * @return $this|void
     * @throws \Magento\Framework\Exception\NoSuchEntityException
     */
    public function execute(Observer $observer)
    {
        $notificationCollection = $this->collectionFactory->create();
        $notificationCollection->getSelect();

        foreach ($notificationCollection->getItems() as $notification) {
            if($this->request->getParam('id') == $notification->getData('product_id'))
```

```
        {

            $this->sendEmail->sendMail($notification->getData('customer_email'), $notification-
>getData('product_id'));

            //$this->notificationRepository->deleteById($notification->getId());
        }
      }

      return $this;
    }
}
```

Модуль “Customer”

Контролер завантаження фото

```
<?php
namespace Volodymyr\Customer\Controller\Adminhtml\Photo;

use Magento\Backend\App\Action;
use Magento\Backend\App\Action\Context;
use Magento\Catalog\Model\ImageUploader;
use Magento\Framework\Controller\ResultFactory;

/**
 * Class Upload
 * @package Volodymyr\Customer\Controller\Adminhtml\Photo
 */
class Upload extends Action
{
    /**
     * #@+
     * Constants
     */
    const ADMIN_RESOURCE = 'Magento_Customer::manage';

    /**
     * #@-
     */

    /**
     * @var ImageUploader
     */
    private $imageUploader;

    /**
     * Upload constructor.
     *
     * @param Context $context
```

```
     * @param ImageUploader $imageUploader
     */
    public function __construct(
        Context $context,
        ImageUploader $imageUploader
    ) {
        parent::__construct($context);
        $this->imageUploader = $imageUploader;
    }

    /**
     * Upload file controller action.
     *
     * @return \Magento\Framework\Controller\ResultInterface
     */
    public function execute()
    {
        $photoId = $this->_request->getParam('param_name', 'customer[customer_photo]');

        try {
            $result = $this->imageUploader->saveFileToTmpDir($photoId);
            $result['cookie'] = [
                'name' => $this->_getSession()->getName(),
                'value' => $this->_getSession()->getSessionId(),
                'lifetime' => $this->_getSession()->getCookieLifetime(),
                'path' => $this->_getSession()->getCookiePath(),
                'domain' => $this->_getSession()->getCookieDomain(),
            ];
        } catch (\Exception $e) {
            $result = ['error' => $e->getMessage(), 'errorcode' => $e->getCode()];
        }

        return $this->resultFactory->create(ResultFactory::TYPE_JSON)->setData($result);
    }
}
```

Клас для створення консольної команди

```
<?php
namespace Volodymyr\Customer\Console\Customer;

use Magento\Customer\Api\CustomerRepositoryInterface;
use Symfony\Component\Console\Command\Command;
use Symfony\Component\Console\Input\InputArgument;
use Symfony\Component\Console\Input\InputInterface;
use Symfony\Component\Console\Output\OutputInterface;

/**
 * Class AdditionalInfo
 * @package Volodymyr\Customer\Console\Customer
 */
```

```
class AdditionalInfo extends Command
{
    /**
     * #@+
     * Constants
     */
    const CUSTOMER_ID = 'customer_id';

    const PHOTO_ATTRIBUTE_NAME = 'customer_photo';
    const EDUCATION_ATTRIBUTE_NAME = 'customer_education';
    const SKILLS_ATTRIBUTE_NAME = 'customer_skills';

    /**
     * #@-
     */

    /**
     * @var CustomerRepositoryInterface
     */
    protected $customerRepository;

    /**
     * AdditionalInfo constructor.
     *
     * @param CustomerRepositoryInterface $customerRepository
     * @param string|null $name
     */
    public function __construct(
        CustomerRepositoryInterface $customerRepository,
        string $name = null
    ) {
        $this->customerRepository = $customerRepository;
        parent::__construct($name);
    }

    /**
     * @inheritDoc
     */
    protected function configure()
    {
        $arguments = [
            new InputArgument(
                self::CUSTOMER_ID,
                InputArgument::REQUIRED,
                'Customer ID'
            )
        ];

        $this->setName('customer:info:extra')
            ->setDescription('Get customer additional info by customer id')
```

```
            ->setDefinition($arguments);

        parent::configure();
    }

    /**
     * Executes the current command.
     *
     * @param InputInterface $input
     * @param OutputInterface $output
     * @return int
     */
    protected function execute(InputInterface $input, OutputInterface $output): int
    {
        $customerId = $input->getArgument(self::CUSTOMER_ID);
        if (!empty($customerId) && is_numeric($customerId)) {
            try {
                $customer = $this->customerRepository->getById(intval($customerId));

                $customerName = $customer->getFirstname() . ' ' . $customer->getLastname();
                $customerPhoto = $this->getCustomAttributeString(
                    $customer->getCustomAttribute(self::PHOTO_ATTRIBUTE_NAME)
                );
                $customerEducation = $this->getCustomAttributeString(
                    $customer->getCustomAttribute(self::EDUCATION_ATTRIBUTE_NAME)
                );
                $customerSkills = $this->getCustomAttributeString(
                    $customer->getCustomAttribute(self::SKILLS_ATTRIBUTE_NAME)
                );

                $output->writeln(PHP_EOL . "ADDITIONAL INFO FOR $customerName (id $customerId):");
                $output->writeln(" Photo: $customerPhoto");
                $output->writeln(" Education: $customerEducation");
                $output->writeln(" Skills: $customerSkills" . PHP_EOL);
                return 0;
            } catch (\Throwable $exception) {
                $output->writeln(
                    PHP_EOL . '<error>' . $exception->getMessage() . '</error>' . PHP_EOL
                );
            }
        } else {
            $output->writeln(
                PHP_EOL . "<error>First argument must be numeric</error>" . PHP_EOL
            );
        }

        return 1;
    }

    /**
```

```
     * Get attribute value.
     *
     * @param \Magento\Framework\Api\AttributeInterface $attribute
     * @return string
     */
    private function getCustomAttributeString($attribute): string
    {
        return $attribute ? $attribute->getValue() : '<error>empty</error>';
    }
}
```

Клас встановлення атрибутів користувача

```
<?php
namespace Volodymyr\Customer\Setup;

use Magento\Customer\Model\Customer;
use Magento\Customer\Setup\CustomerSetupFactory;
use Magento\Eav\Model\Entity\Attribute\SetFactory as AttributeSetFactory;
use Magento\Framework\Setup\InstallDataInterface;
use Magento\Framework\Setup\ModuleContextInterface;
use Magento\Framework\Setup\ModuleDataSetupInterface;

/**
 * Class InstallData
 * @package Volodymyr\Customer\Setup
 */
class InstallData implements InstallDataInterface
{
    /**
     * @var CustomerSetupFactory
     */
    protected $customerSetupFactory;

    /**
     * @var AttributeSetFactory
     */
    private $attributeSetFactory;

    /**
     * InstallData constructor.
     *
     * @param CustomerSetupFactory $customerSetupFactory
     * @param AttributeSetFactory $attributeSetFactory
     */
    public function __construct(
        CustomerSetupFactory $customerSetupFactory,
        AttributeSetFactory $attributeSetFactory
    ) {
        $this->customerSetupFactory = $customerSetupFactory;
        $this->attributeSetFactory = $attributeSetFactory;
```

```
}

/**
 * Install custom attributes for customer.
 *
 * @param ModuleDataSetupInterface $setup
 * @param ModuleContextInterface $context
 */
public function install(ModuleDataSetupInterface $setup, ModuleContextInterface $context)
{
    $setup->startSetup();

    $customerSetup = $this->customerSetupFactory->create(['setup' => $setup]);

    $customerEntity = $customerSetup->getEavConfig()->getEntityType(Customer::ENTITY);
    $attributeSetId = $customerEntity->getDefaultAttributeSetId();
    $attributeSet = $this->attributeSetFactory->create();
    $attributeGroupId = $attributeSet->getDefaultGroupId($attributeSetId);

    $attributeForms = [
        'adminhtml_customer',
        'customer_account_create',
        'customer_account_edit'
    ];

    $customerSetup->addAttribute(
        Customer::ENTITY,
        'customer_photo',
        [
            'type' => 'varchar',
            'label' => 'Photo',
            'input' => 'image',
            'required' => false,
            'visible' => true,
            'user_defined' => true,
            'position' => 1000,
            'system' => 0,
            'backend' => 'Volodymyr\Customer\Model\Customer\Attribute\Backend\Photo',
            'data' => 'Volodymyr\Customer\Model\Customer\Attribute\Data\Photo'
        ]
    )->addAttribute(
        Customer::ENTITY,
        'customer_education',
        [
            'type' => 'text',
            'label' => 'Education',
            'input' => 'textarea',
            'required' => false,
            'visible' => true,
            'user_defined' => true,
```

```
            'position' => 1010,
            'system' => 0
        ]
    )->addAttribute(
        Customer::ENTITY,
        'customer_skills',
        [
            'type' => 'text',
            'label' => 'Skills',
            'input' => 'multiselect',
            'required' => false,
            'visible' => true,
            'user_defined' => true,
            'position' => 1020,
            'system' => 0,
            'backend' => 'Magento\Eav\Model\Entity\Attribute\Backend\ArrayBackend',
            'source' => 'Volodymyr\Customer\Model\Customer\Attribute\Source\Skills'
        ]
    );

    $customerSetup->getEavConfig()
        ->getAttribute(Customer::ENTITY, 'customer_photo')
        ->addData([
            'attribute_set_id' => $attributeSetId,
            'attribute_group_id' => $attributeGroupId,
            'used_in_forms' => $attributeForms
        ])
        ->save();
    $customerSetup->getEavConfig()
        ->getAttribute(Customer::ENTITY, 'customer_education')
        ->addData([
            'attribute_set_id' => $attributeSetId,
            'attribute_group_id' => $attributeGroupId,
            'used_in_forms' => $attributeForms
        ])
        ->save();
    $customerSetup->getEavConfig()
        ->getAttribute(Customer::ENTITY, 'customer_skills')
        ->addData([
            'attribute_set_id' => $attributeSetId,
            'attribute_group_id' => $attributeGroupId,
            'used_in_forms' => $attributeForms
        ])
        ->save();

    $setup->endSetup();
  }
}
```

Додаток Д

Фрагменти коду користувацької частини

Шаблон листа для “Out of Stock Notification”

```
<!--@subject {{trans "Did you forget something?"}} @-->
{{template config_path="design/email/header_template"}}
<table style="float:left; margin-right: 10px" class="fue-abc-items">
  <tr>
    <td style="width:100%;">
      <div class="greeting">Hi dear customer,</div>
      <div>
        {{trans "I’m from %store_name. " store_name=$store.getFrontendName()}}
        {{trans "Your product is already in stock."}}
        {{trans "Thus, we would love to be sure that below items will catch your attention again."}}
      </div>
    </td>
  </tr>
  <tr>
    <td style="width:20%;">
      <img src="{{var Photo}}">
    </td>
    <td style="width:20%;">
      {{var Name}}
    </td>
    <td style="width:20%;">
      {{var SKU}}
    </td>
    <td style="width:20%;">
      {{var Price}}
    </td>
  </tr>
  <tr>
    <td style="text-align: center;">
      <div style="margin-top: 30px;">
        {{trans "Use coupon code "}}
        <span style="font-weight: bold">{{var coupon.code}}</span>
        {{trans "at checkout!"}}
      </div>
      <div>{{var checkOpened}}</div>
    </td>
  </tr>
  <tr>
    <td style="text-align: center;">
      {{trans "If you want to complete your purchase just click the link below."}}
    </td>
  </tr>
  <tr>
```

```
    <td style="text-align: center;">
      <a style="color: #fff;border: 1px solid transparent; padding: 16px 32px; outline: none; text-
decoration: none; cursor: pointer; text-align: center;border-radius: 4px;"
        href="{{var resumeLink}}" target="_blank">
         <button style="color: white; margin-top: 10px;margin-bottom: 30px;border: 0; -webkit-box-
shadow: none; -moz-box-shadow: none; box-shadow: none; background: #000; padding: 10px 15px; -webkit-
border-radius:5px; -moz-border-radius:5px; border-radius:5px;">
           {{trans "Checkout Now"}}
         </button>
      </a>
    </td>
  </tr>
  <tr>
    <td>
      <div>
        {{trans "Or if anything got you confused in shopping process?"}}
        {{trans "Feel free to contact again to discuss problems you may have."}}
        {{trans "Your satisfaction is the best product we desire to deliver."}}
      </div>
    </td>
  </tr>
  <tr>
    <td>
      <div>
        {{trans "Looking forward to seeing you around!"}}
      </div>
    </td>
  </tr>
</table>
{{template config_path="design/email/footer_template"}}
```

UI компонент для відображення нових атрибутів

```
<?xml version="1.0"?>
<form xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="urn:magento:module:Magento_Ui:etc/ui_configuration.xsd">
  <fieldset name="customer">
    <field name="customer_photo" formElement="imageUploader">
      <argument name="data" xsi:type="array">
        <item name="config" xsi:type="array">
          <item name="source" xsi:type="string">customer</item>
        </item>
      </argument>
      <settings>
        <elementTmpl>ui/form/element/uploader/image</elementTmpl>
        <dataType>string</dataType>
        <label translate="true">Photo</label>
        <visible>true</visible>
        <required>false</required>
      </settings>
      <formElements>
```

```
            <imageUploader>
              <settings>
                <required>false</required>
                <uploaderConfig>
                  <param xsi:type="url" name="url" path="volodymyr_customer/photo/upload"/>
                </uploaderConfig>
                <previewTmpl>Magento_Catalog/image-preview</previewTmpl>
                <allowedExtensions>jpg jpeg gif png</allowedExtensions>
                <maxFileSize>4194304</maxFileSize>
              </settings>
            </imageUploader>
          </formElements>
        </field>
        <field name="customer_education" formElement="textarea">
          <settings>
            <dataType>text</dataType>
            <label translate="true">Education</label>
            <visible>true</visible>
            <required>false</required>
          </settings>
        </field>
        <field name="customer_skills" formElement="multiselect">
          <argument name="data" xsi:type="array">
            <item name="config" xsi:type="array">
              <item name="source" xsi:type="string">customer</item>
            </item>
          </argument>
          <settings>
            <dataType>text</dataType>
            <label translate="true">Skills</label>
            <visible>true</visible>
            <required>false</required>
          </settings>
          <formElements>
            <multiselect>
              <settings>
                <options class="Volodymyr\Customer\Model\Customer\Attribute\Source\Skills"/>
              </settings>
            </multiselect>
          </formElements>
        </field>
      </fieldset>
    </form>
```

Додаток Е

Диск з матеріалами магістерської