МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІМ. Ю.М. ПОТЕБНІ ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему **Особливості побудови інтернет-магазину з продажу мисливської зброї**

8.1212-пзс-2

Виконав: студент 2 курсу, групи 8.1212-пзс-2
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Інженерія програмного забезпечення
(код і назва освітньої програми)
А.О. Жаворонков
(ініціали та прізвище)
Керівник доцент, к.т.н., О.М.Міхайлуца
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ «Дісітел»
П.О.Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІМ. Ю.М. ПОТЕБНІ
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

Кафедра Електроніки, інформаційних систем та програмного забезпечення

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)

Освітня програма ___Інженерія програмного забезпечення
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри_____Т.В. Критська_____

"_01_" __вересня_____2023_року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Жаворонкову Андрію Олександровичу
(прізвище, ім'я, по батькові)

1.Тема роботи __Особливості побудови інтернет-магазину з продажу мисливської зброї___

керівник роботи ___Міхайлуца Олена Миколаївна, доцент, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від 02.06. 2023 р. №597-с

2. Строк подання студентом кваліфікаційної роботи 1 грудня 2023 р.

3. Вихідні дані магістерської роботи

- комплект нормативних документів ;
- технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

- огляд та збір літератури стосовно теми кваліфікаційної роботи;
- огляд та аналіз існуючих рішень та аналогів;
- дослідження технологій, методів та засоби проектування веб-сайтів інтернет магазинів;
- створення програмного продукту та його опис;
- перелік вимог для роботи програми;
- дослідження поставленої проблеми та розробка висновків та пропозицій.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

слайдів презентації

6. Консультанти розділів магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата Завдання прийняв

7. Дата видачі завдання 01.09.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1	Аналіз предметної області	02.09-10.09. 23	виконано
2	Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником	11.09-12.09. 23	виконано
3	Ознайомлення із загальними положеннями інтернет-магазинів	13.09-14.09. 23	виконано
4	Аналіз існуючих методів рішення	15.09-20.09. 23	виконано
5	Аналіз методів розробки інтернет-магазинів	21.09-26.09. 23	виконано
6	Узгодження подальших дій з науковим керівником	27.09-28.09. 23	виконано
7	Реалізація сайту інтернет-магазину з продажу мисливської зброї	29.09-16.10. 23	виконано
8	Реалізація користувацького інтерфейсу	17.10-30.10. 23	виконано
9	Представлення отриманих результатів науковому керівнику та узгодження плану подальшого дослідження	01.11-09.11. 23	виконано
10	Усунення додаткових проблем та «багів»	10.11-18.11. 23	виконано
11	Перевірка працездатності створеного програмного застосунку	19.11-23.11. 23	виконано
12	Оформлення звіту	23.11-27.11. 23	виконано

Студент _____ А.О.Жаворонков
(підпис) (ініціали та прізвище)

Керівник роботи _____ О.М. Міхайлуца
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ І.А. Скрипник
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

Сторінок: 80

Рисунків: 28

Таблиць: 3

Джерел: 20

Жаворонков А.О. Особливості побудови інтернет-магазину з продажу мисливської зброї : кваліфікаційна робота магістра спеціальності 121 «Інженерія програмного забезпечення» / наук. керівник О.М. Міхайлуца. Запоріжжя : ЗНУ, 2023. 80 с.

Метою і завданням кваліфікаційної роботи є аналіз існуючих технологій для створення веб-додатків, визначення та подальше обґрунтування основних методів та засобів розробки веб-сайту інтернет-магазину для продажу мисливської вогнепальної зброї та її комплектуючих.

У процесі дослідження було розглянуто можливості сучасних мов програмування та фреймворків, таких як мова програмування Rust для написання бекенду, PHP для фронтенду, фреймворків Rocket та Elemental UI для розробки інтерфейсу користувача.

Спроєктовано та розроблено веб-сайт інтернет-магазину для продажу мисливської вогнепальної зброї та її комплектуючих Зроблено акцент на збереженні певного консерватизму та дотриманні принципів надійності, економічності та безпеки.

Ключові слова: веб-сайт, інтернет-магазин, Rust, PHP, Rocket, Elemental UI, мисливська зброя.

SUMMARY

Pages: 80

Figures: 28

Tables: 3

Sources: 20

Shavoronkov A.O. Features of building an online store selling hunting weapons: qualification work of the master of specialty 121 "Software Engineering" / Science head O.M. Mikhailutsa. Zaporizhzhia : ZNU, 2023. 80 p.

The purpose and objective of the qualifying work is to analyze existing technologies for creating web applications, identify and further justify the main methods and means of developing an online store website for the sale of hunting firearms and their components.

The capabilities of modern programming languages and frameworks, such as the Rust programming language for writing the backend, PHP for the frontend, the Rocket and Elemental UI frameworks for developing the user interface, were examined during the study.

An online store website for the sale of hunting firearms and its components was designed and developed. Emphasis is placed on maintaining a certain conservatism and observing the principles of reliability, efficiency and safety.

Keywords: website, online store, Rust, PHP, Rocket, Elemental UI, hunting weapons.

ЗМІСТ

ВСТУП

Актуальність теми

В нашій країні рівень інформатизації та прогресу в галузі інформаційних технологій постійно підвищується. Нині при веденні бізнесу активно застосовуються сучасні інформаційні технології на основі глобальної комп'ютерної мережі Інтернет. Хоча сама мережа Інтернет має досить довгу історію, її комерційне використання розпочалося лише в 1988 році.

Система World Wide Web (WWW), що визначає сучасний дружній вигляд мережі Інтернет, була розроблена 1992 року. Таким чином, історія застосування мережі Інтернет для ведення бізнесу налічує майже 30 років. Якщо в таких країнах, як США, Японія та країни Європи, пік розвитку мережі Інтернет і впровадження Інтернет технологій у бізнес уже позаду, то в Україні він, імовірно, ще не досягнутий і його можна очікувати найближчими роками. Про це свідчить світова та вітчизняна статистика зростання користувачів мережі Інтернет. Нині є багато друкованих видань та Інтернет джерел, присвячених різним аспектам ведення Інтернет бізнесу. Це книжки таких зарубіжних авторів, як Г. Грехам, Д. Козьє, В. Махайан, Дж. Вінд, Г. Хардакер, М. Гейг, В. Хенсон, Д. Еймор.

На сьогодні використання інформаційних технологій у сфері торгівлі дає змогу значно скоротити витрати на утримання торговельних площ шляхом відмови від них. Для більшості компаній продаж товарів і послуг через Інтернет є основним способом знайти нових постачальників і клієнтів. Також наявність Інтернет магазину є конкурентною перевагою.

Мета і завдання дослідження

Метою даної кваліфікаційної роботи магістра є визначення та подальше обґрунтування основних методів та засобів розробки веб-сайту інтернет магазину для продажу мисливської вогнепальної зброї та її комплектуючих.

В рамках визначеної мети роботи передбачена реалізація наступних завдань:

1) ознайомитися з поняттям веб-сайту;

2) дослідити принцип роботи довільного інтернет-магазину;

3) порівняти існуючи платформи та системи для розробки веб-сайтів;

4) спроєктувати та розробити веб-сайт інтернет магазину для продажу мисливської вогнепальної зброї та її комплектуючих.

Об’єкт дослідження

Об’єктом дослідження є методи та засоби розробки веб-сайту інтернет магазину.

Предмет дослідження

Предметом дослідження є проєктування та розробка веб-сайту інтернет магазину для продажу мисливської вогнепальної зброї та її комплектуючих.

Методи дослідження

Аналіз літературних джерел, синтез дослідження та аналіз досліджуваного матеріалу.

Наукова новизна

Виходячи з аналізу існуючих рішень та особливостей їх застосування для розробки інтернет-магазину для продажу мисливської вогнепальної зброї та її комплектуючих, виявлені основні принципи, щодо контенту, зручного інтерфейсу і комфортності роботи користувача з сайтом.

Практичне значення одержаних результатів

При порівняльному аналізі організації контенту виявлені кращі аналоги веб-сайтів з продажу мисливської вогнепальної зброї, що дозволило виконати й їх детальний аналіз та зрозуміти які алгоритми та фреймворки є найбільш ефективними для розробки інтернет-магазину мисливської вогнепальної зброї та її комплектуючих.

Глосарій

Пошукова оптимізація сайта SEO (Search Engine Optimization) — комплекс робіт із зовнішніми факторами в пошукових системах.

Івент (event), спеціальна подія — подія, що проводиться організацією та спрямована на вирішення PR-завдань, які є інформаційним приводом та потребують фінансових витрат й організаційної підготовки.

MYSQL — вільна реляційна система управління базами даних.

SER (сер, абревіатура для Secure (секьюр — безпечний), Efficient (ефішієнт — ефективний), Reliable (релайбл — надійний)) — вигаданий принцим, для забезпечення безпеки, продуктивності та надійності продукту.

Архітектура — спосіб побудови продукту.

Front-End (фронт-енд) — на противагу до Back-End — щось, що ближче (Front) до користувача. В нашому випадку це сервер на PHP, з яким користувач безпосередньо взаємодіє.

Back-End (бек-енд) — сервер, прихований від користувача, де відбувається вся логіка застосунку.

PHP — мова програмування для створення серверів для веб застосунків. Всі запити обробляє на сервері, відсилаючи користувачу вже готову веб-сторінку. Використовується для створення динамічних веб сторінок. PHP8 — новий стандарт мови програмування, який продовжує перетворювати її з динозавра в сучасну технологію.

Semantic — графічна бібліотека, що пришвидшує створення веб-інтерфейсів). Використовується багатьма великими компаніями, є досить новою, але популярною.

CSS — мова для створення стилів для веб-сторінок. Дана мова використовуєься Semantic.

HTML — мова розмітки, основна в інтернеті. Використовується разом зі CSS.

Rust — мова для системного програмування. Швидка, надійна, нова (лише 9 років), вважається однією з найскладніших.

Rocket — фреймворк для расту, який дає можливість створювати веб-сервери.

API — інтерфейс, функціонал, який користувач може використати.

Framework — сукупність бібліотек за спільною тематикою, набір інструментів для розробки в цій тематиці.

Diesel — фреймворк для расту для роботи з базами даних.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРОБЛЕМИ РОЗРОБКИ ВЕБ-САЙТІВ ІНТЕРНЕТ МАГАЗИНІВ

1.1 Сучасні технології розробки інтернет магазинів

Застосування електронних комп'ютерних технологій у бізнесі не обмежується межами, в яких це допускається існуючими виробничими та управлінськими процесами. Навпаки, з кожним роком наростає зворотна залежність — електронні інформаційні технології справляють значний, якщо не визначальний, вплив на формування і розвиток економічних відносин. Цей процес є основою для виникнення деяких принципово нових явищ в економіці, прикладом чого є електронна комерція.

Чіткого й однозначного визначення електронної комерції досі не існує, що пов'язано, по-перше, з її порівняно нещодавнім виникненням, а по-друге, з тим, що, як і електронні інформаційні технології, що її породжують, електронна комерція активно розвивається, збагачуючись дедалі новими рисами. Проте в законодавстві низки країн уже існують різні її визначення. Згідно з визначенням, даним автором роботи [1]: «Під електронною комерцією розуміють комерційну діяльність у сфері реклами та розповсюдження товарів і послуг за допомогою використання мережі Інтернет» [1]. Визначення, прийняте у США, дещо ширше: згідно з ним, електронною комерцією називається здійснення фінансових транзакцій електронними засобами [1].

Однак, незважаючи на розпливчастість існуючих визначень, електронна комерція бурхливо розвивається. Першими з'явилися роздрібні електронні магазини типу B2C (business-to-customers), націлені на кінцевого споживача [2]. Потім швидко стали розвиватися сайти B2B (business-to-business), призначені для корпоративних покупців. Нині обсяги замовлень на купівлю через Інтернет уже значні та зростають стрімкими темпами. Таким чином, спостерігається тенденція

розвитку електронної комерції, зокрема й інтернет-магазинів, як її елементів. Електронна торгівля у віртуальному магазині за своєю структурою схожа на традиційну торгівлю. Існують переваги віртуального магазину перед реальним. Можемо зменьшити чисельність персоналу за рахунок скорочення обсягу взаємодії з клієнтами, орендувати дисковий простір і розміщувати «електронні вітрини» дешевше і простіше ніж орендувати торговельні приміщення і розміщувати товари на полицях, нетреба касове обслуговування. Вирішуючи питання про підключення Інтернет магазину до існуючої платіжної системи, керівники заздалегідь оцініють кількість користувачів системи, що будуть цікавитися асортиментом такого магазину.

Говорячи про віртуальний магазин, ми говоримо про Інтернет магазин, або визнаємо його, як віртуальне підприємство. Згідно з визначенням, віртуальний магазин — це територіально роз'єднане співтовариство співробітників магазину (продавців, касирів) і покупців, які в свою чергу можуть спілкуватися і обмінюватися інформацією за допомогою засобів електронного звізку, не тільки при частковій відсутності особистого прямого контакту, але і за повної її відсутності.

Порівнюючи електронну торгівлю та традиціну, можна сказати, що їх структура є однаковою. На сьогодні можливе використання значної кількості інструментів і технологій, які дають змогу реалізувати Інтернет магазин. Серед таких допустимих засобів реалізації є сучасні мови програмування, такі як JavaScript, PHP, Java, Python, HTML, а також різні комбінації цих мов. Клас систем управління контентом, які називають CMS (Content Management Systems) використовують для зручної реалізації Інтернет магазину. CMS надає базовий функціонал, використовуючи який можно створити сайт

Порівнюючи з різними програмними продуктами, системи управління контентом також виділяють безкоштовні та платні, а також з відкритим та закритим вихідним кодом. До переваг платних систем відносять наступне:

наявність технічної підтримки з боку розробників, регулярні оновлення, можливість незначних доопрацювань у рамках ліцензійної угоди. Говорячи про системи, що розповсюджуються безоплатно, виділяють наступні переваги, а саме: відсутність необхідності придбання, можливість самостійного аудиту безпеки (недокументовані можливості).

Переваги CMS мають сісце для усіх зацікавлених в проекті осіб (розробники, користувачі). Модульна структура дає можливість гнучко налаштовувати функціонал, залишаючи при цьому тільки необхідний. Окрім того, використання модулів в проєктуванні забезпечує високу якість розробки, зменшуючи при цьому вірогідність помилки в програмному коді, яку немає можливості виявити та усунути. Навіть не маючи достатньо великих знань мов програмування, користувачі CMS мають зручний інтерфейс для управління контентом.

Розробники програмних додатків мають можливість використовувати досить велику кількість готових для застосування системи управління контентом. Найбільш популярними серед розробників є наступні: «OpenCart», «osCommerce», «WordPress», «1С-Бітрікс».

Wordpress — система управління, яка реалізована на мові програмування PHP, з базою даних MySQL. Wordpress випущено під ліцензією GPL v2. Завдяки наявності різних плагінів можливе створення проектів будь-якого рівня.

OpenCart — система управління контентом. Зазначена CMS має такі переваги: безоплатна система, патерн проектування MVC, підтримка об’єктно-орієнтованого програмування, багатий функціонал магазину. Як і будь-яка система, CMS OpenCart має свої недоліки: базова функціональність, проблеми з оновленнями, мінімальна підтримка і складнощі, що виникають зі створенням власних сторінок (наприклад, контактної форми). Є найбільш зручною для створення інтернет магазину, оскільки має відповідний функціонал.

Серед комерційних систем управління контентом виділяють osCommerce. До недолків можна віднести неможливість перевірити цю систему управління контентом на наявність недокументованих можливостей.

1С-Бітрікс — призначена для створення корпоративних сайтів, інформаційних і довідкових порталів. Для зберігання сайтів використовується саме реляційна система управління базами даних. Суттєвим недоліком є закритий вихідний код продукту. Також неможливо перевірити продукт на наявність недокументованих можливостей.

1.2 Огляд видів веб-сайтів

Серед поширених визначень виділяють: «Веб-сайт — це сторінка з інформацією, що розміщена в глобальній мережі» [20]. Розрізняють такі види веб-сайтів:

Landing page (односторінковий сайт)

«Посадкова сторінка або Лендінг (від англійського landing page)» [20] це спроєктована сторінка сайту за спеціальною формою, мета якої спонукати відвідувачів до вчинення певних дій, а саме: придбання через сторінку сайта товар, можливість оформити підписку, замовити послуги.

Сайт візитка

З метою просування власного біснесу на ринку кожна компанія намагається мати власне веб-представництво в Інтернеті. Тобто вони мають створену персональну сторінку, або замовляють для себе великий корпоративний ресурс або звертаються до інформаційнорозважального порталу. Але дослідження показали, що коли компанія тільки починає своє просування у бізнес-просторі, то вони замовляють для себе саме сайт-візітівку.

Корпоративні сайт

На сторінках корпоративного сайту розміщується повна інформація про організацію, ті послуги, що вона надає, продукцію, яку вона пропонує. Якщо говорити про оновну відмінність корпоративних сайтів від сайтів-візиток, то виділяють розширені функціональні можливості, а також можливість інтеграції з внутрішніми системами компанії.

Промо-сайти

Плануючи випуск нової продукції або нового товару компанія організує певні дії, щоб отримати лояльність і визнання від споживачів, а також розширити коло постійних покупців або замовників. Саме для просування нових пропозицій і створюють такі промо-сайти.

Інтернет-магазин

В сучасних умовах можна спостерігати таку ситуацію, що значна кількість існуючих компаній досить давно почали використовувати інтернет-магазини для того, щоб перенести продаж товарів в онлайн-простірт та щоб отримати прибуток через мережу. Говорячи про відмінності інтернет-магазину від інших видів сайтів акцентують увагу на можливості не тільки переглядати каталог товарів на сайті, а й здійснювати покупку прямо на сторінці сайту.

Іміджевий сайт

Іміджевий сайт — це сайт, призначений, перш за все, для того, щоб створити позитивне враження про власника цього сайту за допомогою візуального представлення. Серед замовників виділяють і доволі великі компанії і представників як середнього так і малого бізнесу, а також приватних осіб.

Персональний сайт

Подають інформацію про приватних осіб. Аналізуючи такі проєкти, можна виділити наступні характеристики як: малий об'єм, наявність в ньому відомостей біографічного або особистого характеру, окрім того, в ньому можна побачити

інформацію про рід діяльності власника сайту або про послуги, які він надає ця компанія.

Портали

На порталі користувач може знайти для себе всю необхідну інформацію, а саме: пошукові та поштові сервіси, форуми, новини, авторські блоги, голосування, та різну іншу інформацію.

Web-додатки

Веб-додатки, згідно з визначенням, це різного плану програмні продукти, але доступ до них здійснюється саме через браузер. Переважно серед замовників веб-додатків бачимо комерційні організації, тому описуючи функціонал веборієнтованого додатку розробники використовують інструменти, які орієнтовані на бізнес.

Обраний проєкт відноситься до категорії інтернет магазинів.

1.3 Огляд архітектурних шаблонів інтернет магазину

Шаблон проектування — архітектурна конструкція, призначена для проектування деяких контекстів, що часто виникають [10]. Шаблони є найбільш схожими на готові бібліотеки. Відповідними шаблонами для інтернет-магазинів є MVC.

У контексті розроблення веб-додатка можна використовувати класифікацію шаблонів проектування Мартіна Фаулера [7], згідно з нею шаблони можна розділити за класами:

1) базові шаблони;
2) шаблони веб-представлення;
3) шаблони архітектурних джерел даних;
4) шаблон об'єктно-реляційної логіки;
5) шаблони об'єктно-реляційного структурування;

6) шаблони логіки сутності;

7) шаблони розподілу даних;

8) шаблони локальної конкуренції.

Будь-який із цих класів містить у собі деякий набір шаблонів, одним із яких і є MVC (Model - View - Control), або, як зазначив автор роботи, його модифікації:

1) «MVP (Model - View – Presenter)» [7];

2) «MVVM (Model -View - View - View - Model)» [7];

3) «HMVC (Hierarchical MVC)» [7];

4) «PAC (Presentation - Abstraction - Control)» [7].

MVC дає змогу реалізувати бізнес-логіку додатка без необхідності витрачати значні зусилля на програмування. Концепція MVC була описана 1978 року Тюргве Реенскаугом, проте остаточна концепція MVC опублікована 1988 року в журналі Technology Object. Подальший розвиток представлено шаблонами HMVC, MVP, MVVM. Концепція MPV похідна від концепції MVC. MVP — шаблон проєктування користувацького інтерфейсу, спеціально розроблений для легкого автоматичного тестування та поділу відповідальності в презентаційній логіці, шляхом відокремлення логіки від відображення. У MVP Presenter виконує роль посередника, таку саму як контролер у MVC.

Також Presenter відповідає за керування подіями користувацького інтерфейсу, обробка яких у MVC відводиться уявленням (View). Паттерн MVVM (Model-View-ViewModel) допускає відокремлення логіки додатка від візуальної частини (подання). Цей патерн задає загальну архітектуру застосунку і має тісніший зв'язок між моделлю і поданням. Цю концепцію реалізовано у WPF і Silverlight.

Для реалізації інтернет магазину було обрано шаблон MVC, оскільки він забезпечує просту реалізацію бізнес-логіки інтернет магазину і є типовим рішенням під час розроблення веб-додатків

1.4 Аналіз ресурсів аналогічної тематики

Проведений аналіз аналогічних проектів показав, що є типові складові у розробці будь-якого інтернет-магазину, а саме: архітектура представленого інтернет магазину, за яким принципом він працює, як буде оформлюватися навігація. Окрім цього, ще треба звернути увагу на можливість отримати інформацію про каталоги товарів та послуг, на те, як буде все розташоване та відображено, а також, на доступ до новин, що пов'язані з товарами компанії, за певною тематикою. Для зручності використання на сайті інтернет-магазину повинні бути певні онлайн-сервіси (наприклад онлайн оплата та замовлення доставки товарів та послуг), відповідні посилання на контактну інформацію та інформацію про час та методи роботи компанії. Також, на сторінках сайтів інтернет-магазинів відображаються статті та новини відповідно до характеристик товарів і послуг, які він надає. Розглянемо додатально деякі з інтернет-магазинів.

ebay.com

Через цей інтернет-магазин продають товари та послуги в інтернеті. В ньому продають товари люди та компаній з усього світу. Сайт складається з наступних розділів:

1) головна сторінка;

2) контактна інформація, загальна інформація про сайт, яким чином користувач може оформити завмовлення;

3) наявність підкатегорій для зручності пошуку товарів користувачами;

4) інформація про розробників, представлення карти сайту містится у футері сторінки.

Головна сторінка сайту ebay.com представлена на рисунку 1.

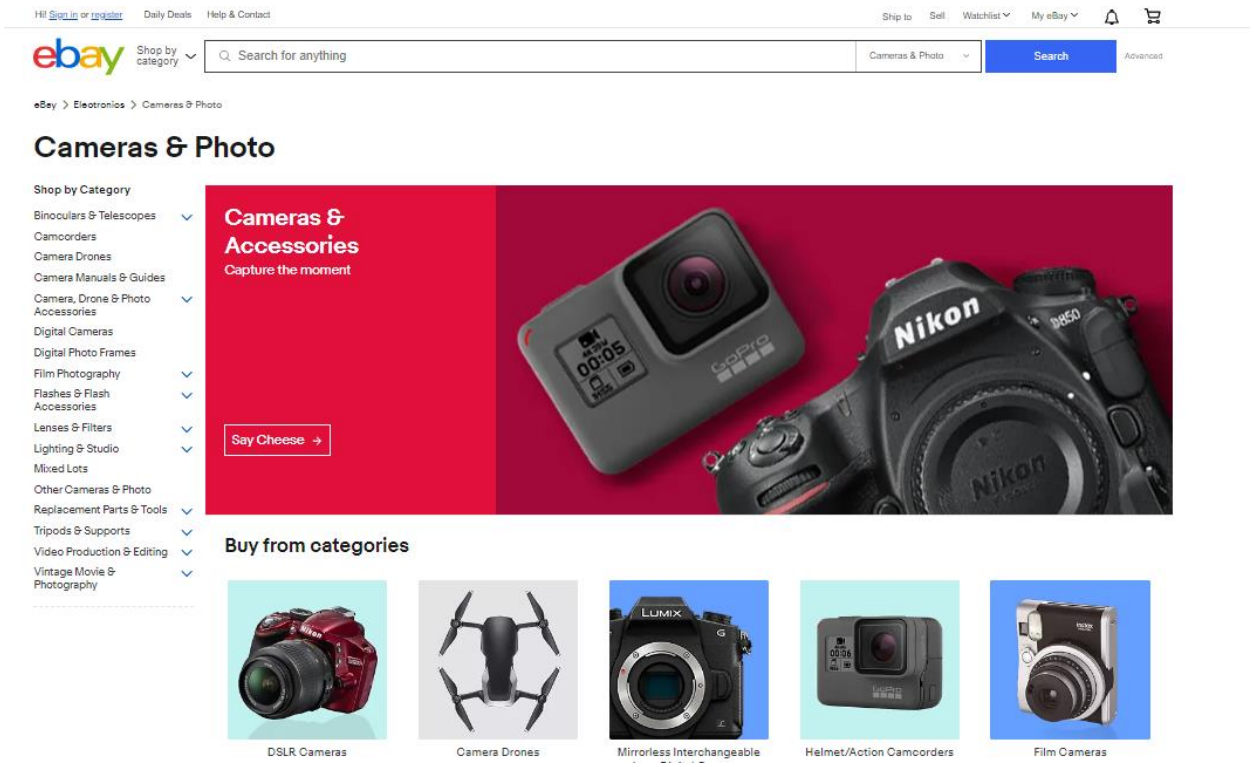

Рисунок 1 — Сторінка сайту ebay.com

Представлений інтернет-магазин було розроблено 4 вересня 1995 року, в рамках розробки власного веб-сайту відомий програміст П'єр Омідьяр створив онлайновий аукціон «AuctionWeb».

aliexpress.com

Схожим великим інтернет-магазином є сайт aliexpress.com, особливо на території Китая. За своїм головним призначенням він не відрізняється від більшості додатків аналогічного призначення. Структура інтернет-магазину складається з наступних елементів:

1) головна сторінка (див. Рис. 2) відображає основне меню, до якого входять популярні категорії і підкатегорії, а також нові, рекомендовані та популярні товари та послуги.

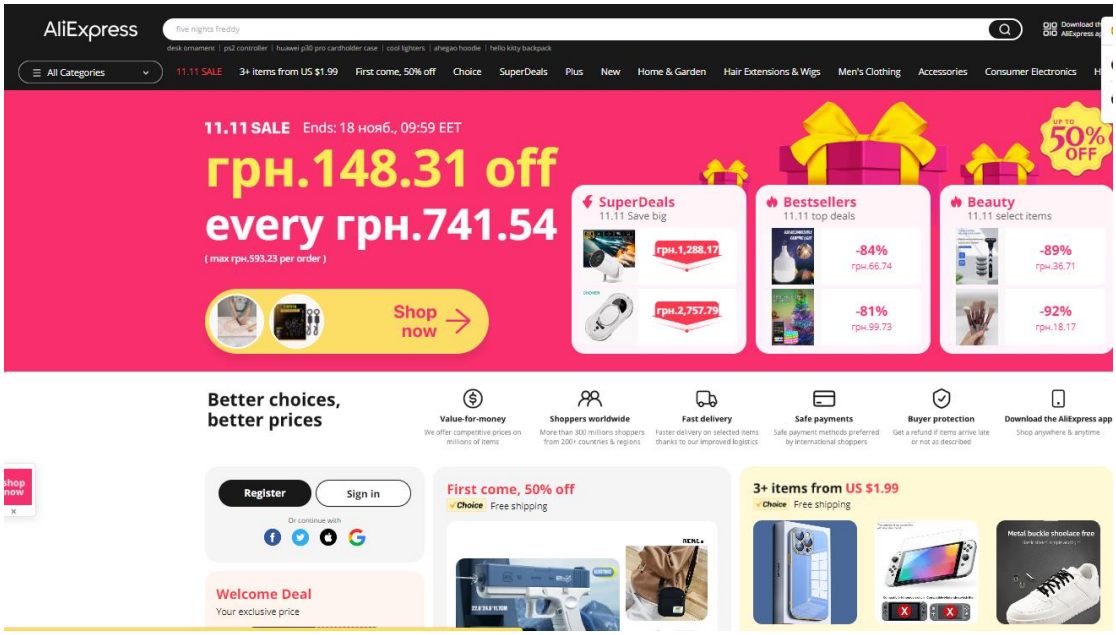

Рисунок 2 — Сторінка сайту aliexpress.com

2) Сайт містить футер, на сторінці якого містяться статті, контактна інформація, форма зворотнього звязку.

3) Наявність меню навігації категорій товарів дозволяє швидко знаходити та обирати товари.

4) Наявність форми для входу зареєстрованих клієнтів, у разі якщо користувач успішно пройшов авторизацію, він буде мати доступ до особистого кабінету та власного кошика з обраними товарами.

Розробниками даного інтернет магазину є Джек (1999 рік).

Як виділяє автор публікації [12]: «Успіх інтернет магазину залежить від ряду факторів, які, умовно, можна розділити на дві групи» [12]:

- якість с точки зору того, наскільки інтернет магазин відповідає технічному призначенню та вимогам користувача;
- якість відповідності бізнес потребам самої компанії.

Розглянемо структуру існуючих інтернет-магазинів, які пропонують аналогічну продукцію.

Інтернет-магазин ZBROIA пропонує широкий вибір пневматичної зброї, револьверів та пістолетів під набої Флобера, шумових пістолетів, різних куль та набоїв, прицілів для спортивної стрільби та мисливства, кріплень для оптики, ножів для мисливства і туризму, самозахисту, одягу для полювання, луків, арбалетів, макетів масо-габаритних (ММГ) і реплік різної зброї, сувенірної продукції.

Інтернет-магазин «ZBROIA» має інтуїтивно зрозумілий для користувача дизайн та структуру. Головна сторінка сайту інтернет-магазину ZBROIA наведена на рисунку 3.

Zbroia.com.ua

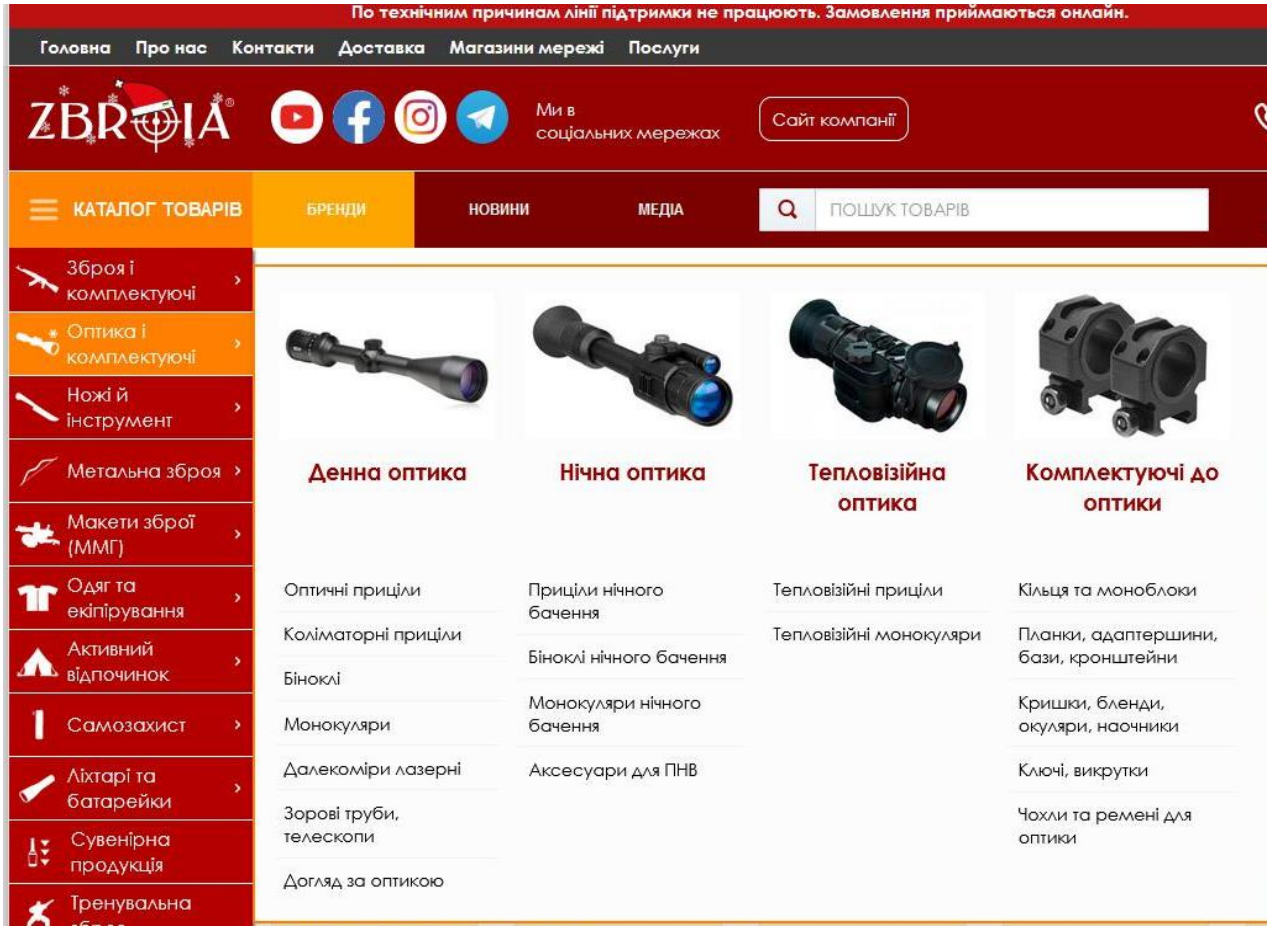

Рисунок 3 — Інтернет-магазин ZBROIA

До переваг даного сайту можна віднести те, що розбивка на категорії дозволяє користувачам легко орієнтуватися в каталогах товарів та швидко знаходити потрібний товар. В архітектурі сайту продумано кожен елемент, в кожному блоці міститься заклик до дії або тригер, що мотивує до покупки.

Збройовий магазин Сафарі-Україна

Український інтернет-магазин з продажу мисливської зброї і супутніх аксесуарів. Додаток має яскравий і красивий дизайн, що виконано у фірмовому стилі, він має зручну навігацію і функції пошуку по фільтрам Головна сторінка сайту збройового магазину Сафарі-Україна наведена на рисунку 4.

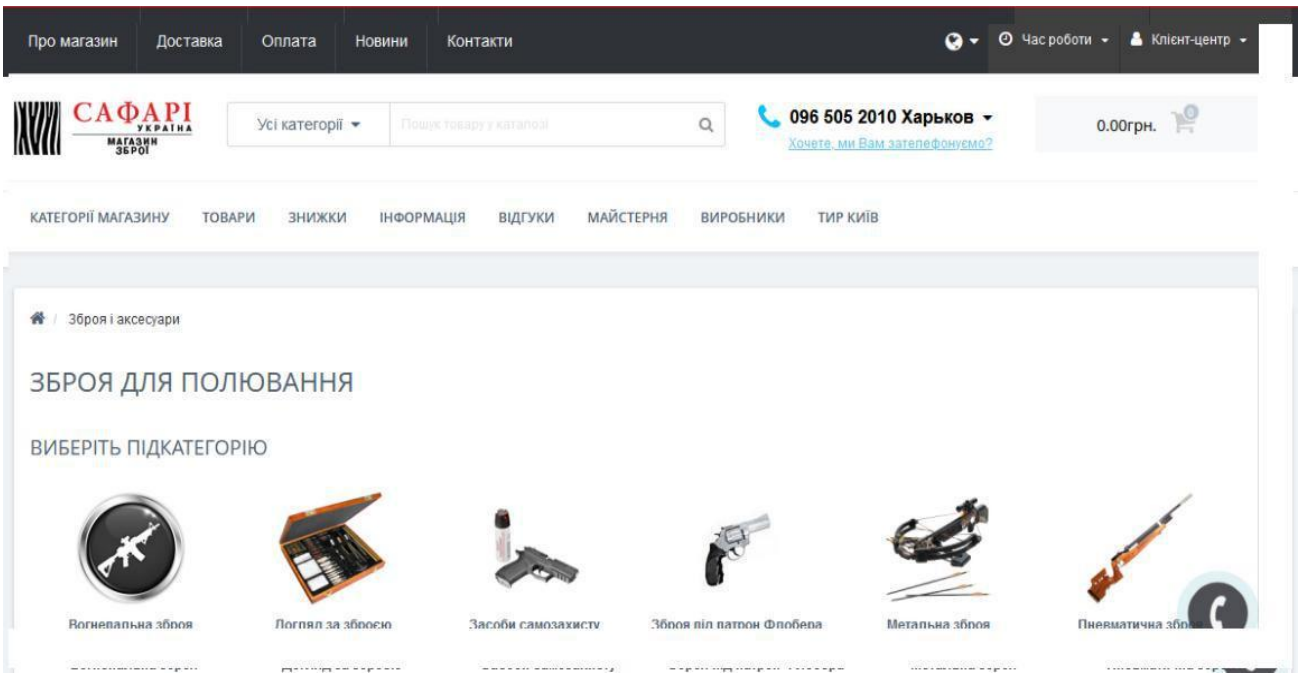

Рисунок 4 — Інтернет-магазин ZBROIA

Детально продуманий функціонал сайту дозволяє легко орієнтуватися відвідувачам та здійснювати покупки.

До особливостей можна віднести наявність пропозицій та рекомендацій з якими одразу зустрічається користувач при першому входженні на сайт. Сторінка

сайту з акціями та подарунками збройового магазину Сафарі-Україна наведена на рисунку 5.

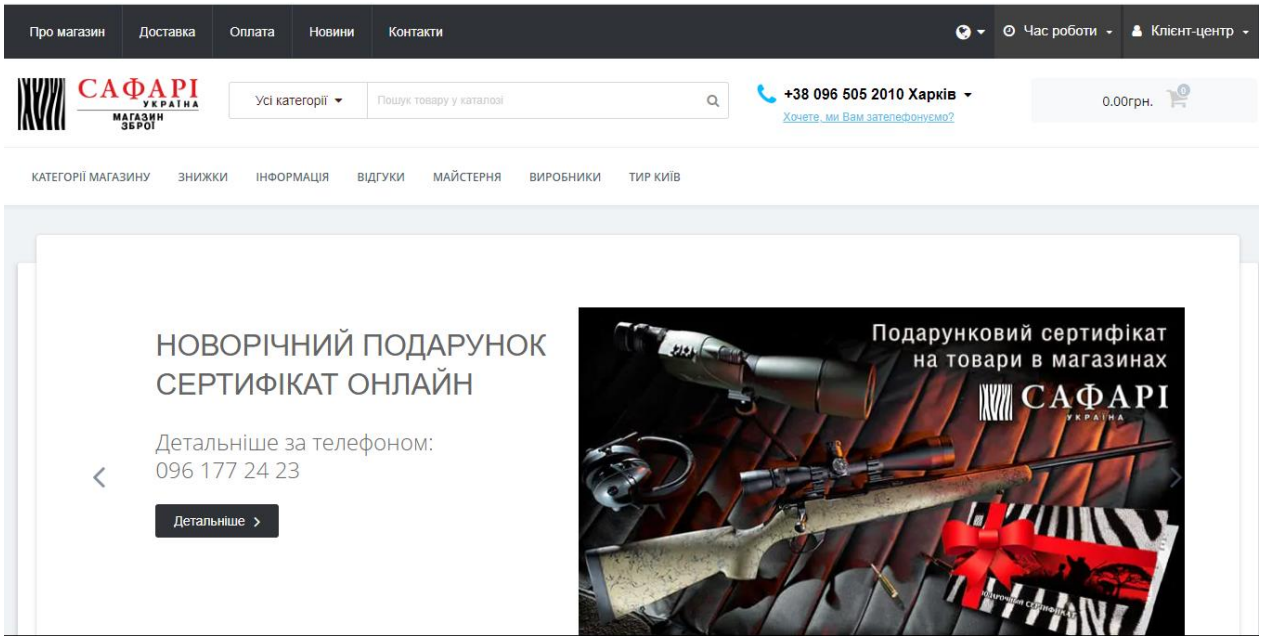

Рисунок 5 — Сторінка сайту з акціями та подарунками збройового магазину Сафарі-Україна

Stvol.ua

Інтернет магазин *Stvol* — це оптимальний баланс між потребами та можливостями стрільців в якому можна придбати зброю та товари для полювання, спорту і туризму. Головна сторінка сайту мережи збройових магазинів **Stvol** представлена на рисунку 6. Яскравий, повний анімаційних ефектів дизайн сайту вражає користувача одразу при переході на сторінку сайту.

До недоліків можна віднести неможливість візуалізації сервісних центрів мережі Stvol в обраному місті з прив’язкою до гугл-карт. Сторінка пошуку сервісного центру сайту наведена на рисунку 7.

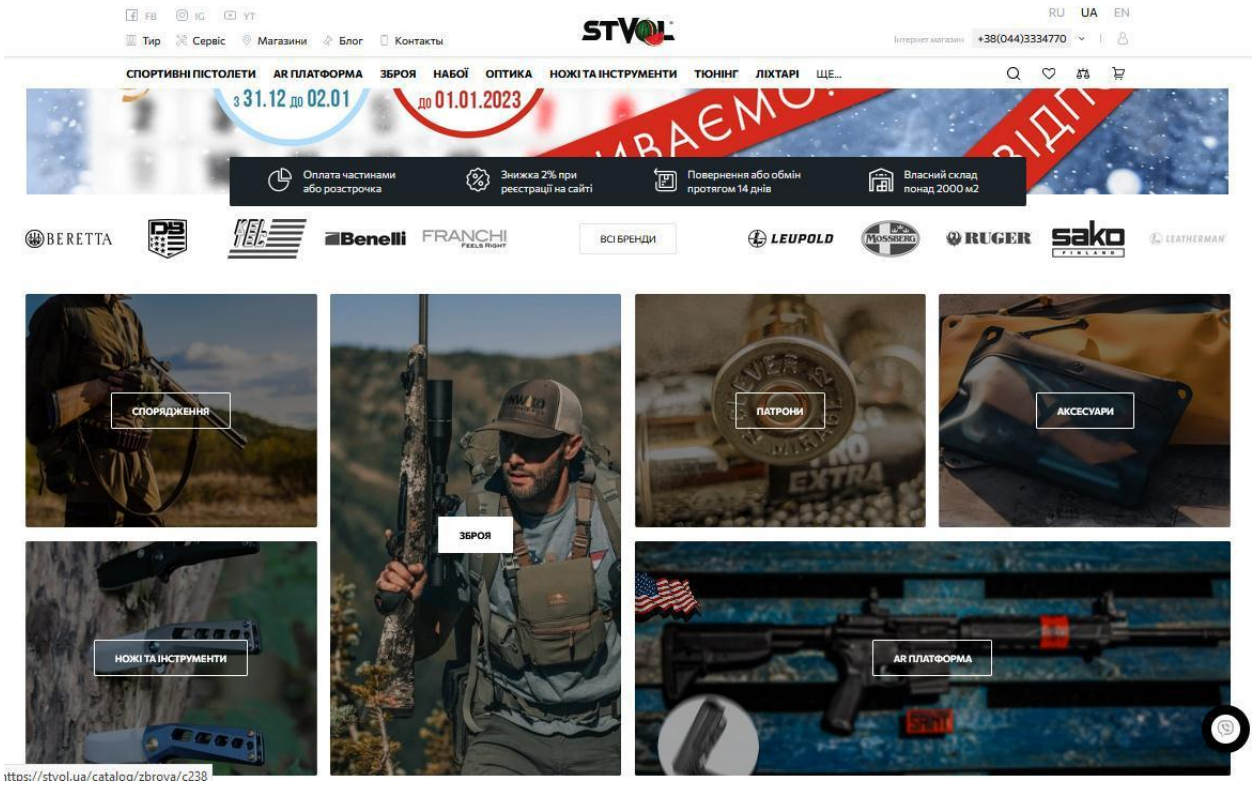

Рисунок 6 — Мережа збройових магазинів Stvol

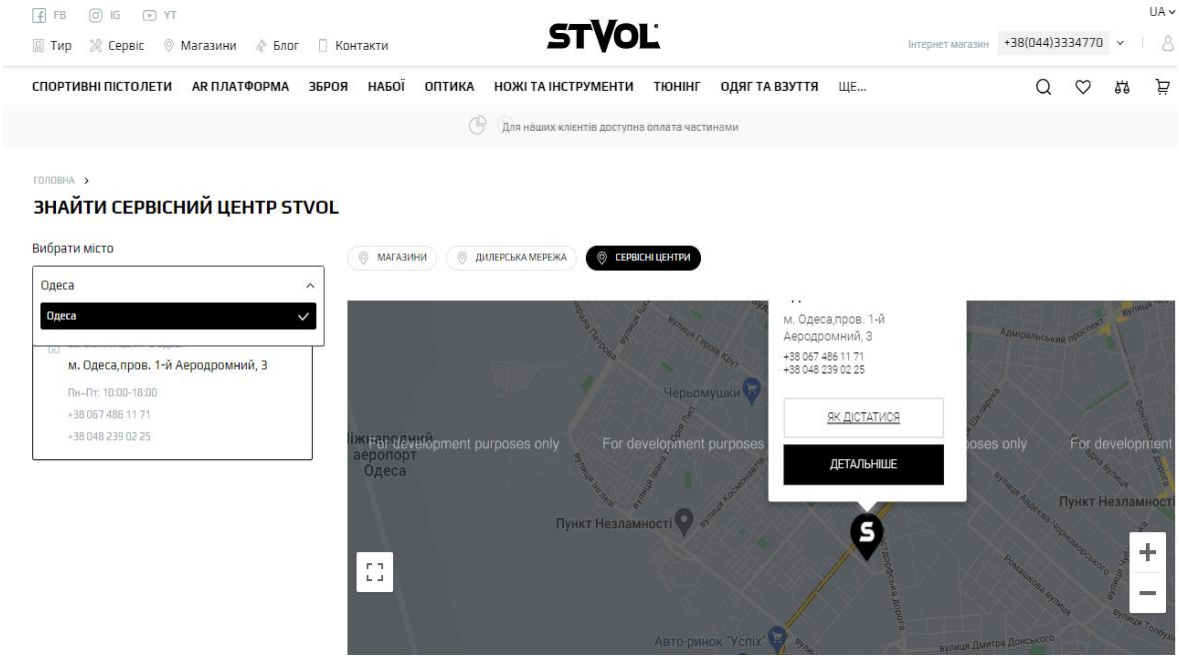

Рисунок 7 — Сторінка пошуку сервісного центру сайту мережі збройових магазинів Stvol

Huntmasters.com.ua

Мисливський магазин Hunt Masters, кращий інтернет магазин товарів для полювання. І дизайн і функціонал сайту максимально орієнтований на свого користувача, він, як і більшість з розглянутих сайтів мають зручну для SEO-просування структуру Головна сторінка сайту мисливського магазину Hunt Masters представлена на рисунку 8.

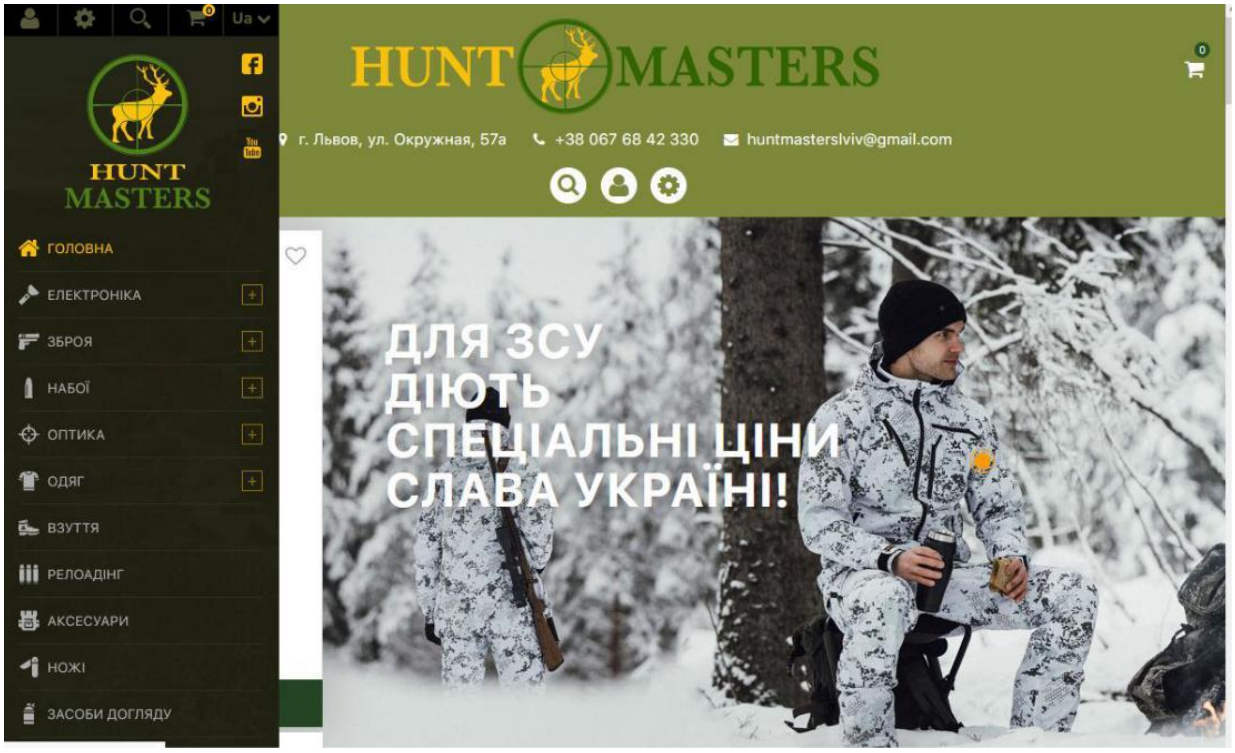

Рисунок 8 — Мисливський магазин Hunt Masters

Однією з особливостей даного сайту є можливість ведення блогу з іншими користувачами. Інформацію структуровано та згруповано за тематикою та датами. Інтерфейс сторінки блогу сайту мисливського магазину Hunt Masters наведено на рисунку 9.

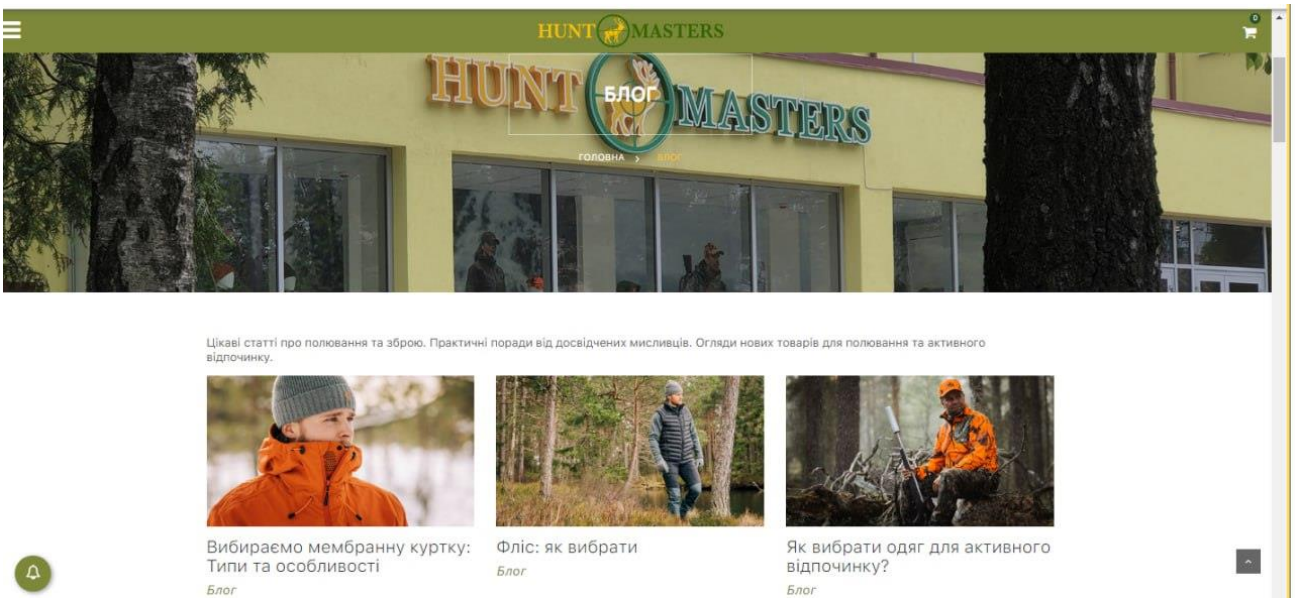

Рисунок 9 — Інтерфейс сторінки блогу

1.5 Висновки до розділу 1

В першому розділі представленої роботи визначено специфіку веб-сайту інтернет магазину. Наведено основні приклади веб-сайтів в інтернеті, їх особливості. Приведено приклади реальних сайтів, розглянуто їх архітектури на прикладі інтернет-магазинів: ZBROIA, Сафарі-Україна, Stvol, Hunt Masters.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ ПРОГРАМНИХ ЗАСОБІВ РЕАЛІЗАЦІЇ ЗАСТОСУНКІВ

2.1 Огляд сучасних технологій створення web-сайтів

Розробка сучасних web-додатків незалежно від рівня складності та функціонального призначення вимагає підвищеної надійності та безпеки від несанкціонованого доступу. Серед існуючих мов програмування багато з них використовується для створення web-додатків. Перевага кожної з них може виявлятися лише в контексті конкретного завдання. Деякі мови та фреймворки є нерозривним цілим, а інша частина може використовуватися і без фреймворку.

Під час розробки інтернет магазину можливе використання різних мов програмування, як інтерпретованих, так і компільованих: PHP, C, C++, Java, Ruby, Python. У PHP є вибір із великої кількості функцій, завдяки яким написання коду полегшено. Також PHP легко інтегрувати з різними мовами програмування. Відкритий вихідний код дає змогу вивчити PHP на предмет недокументованих можливостей. PHP 7 є актуальною версією.

Для забезпечення працездатності веб-додатка, що працює на PHP, потрібен веб-сервер. Як веб-сервер використовується Apache з підтримкою MySQL і PHP. СУБД, такі як SQLServer, MySQL, Oracle. MySQL, використовуються для керування базою даних.

В публікації [8] визначено переваги СУБД MySQL:

1) вільно поширюється;
2) має відкритий вихідний код, що дає змогу перевірити MySQL на наявність недокументованих можливостей;
3) добре масштабується.

Rust

Мова програмування Rust була розроблена 2010 року. Вона є спадкоємцем мов системного програмування, зокрема C++, поєднує в собі багато переваг своїх старших товаришів.

Rust — високорівнева мова, її підносять як інструмент, що компілюється, є системним з такими характеристиками [6]:

Компільованість. Програма, написана мовою, являє собою окремий файл, готовий для запуску на будь-якій машині з відповідною операційною системою. Тобто, встановлення компілятора і середовища розробки не потрібне, аби скомпільована версія була адаптована до комп'ютера.

Системність. Мова дозволяє писати програми для всіх складових системи. Тобто, для операційних систем, драйверів і службових утиліт. Rust підходить і для написання звичайних програм, що використовуються в калькуляторах, у системах управління базами даних тощо. Усі вони працюють дуже швидко і задіюють можливості заліза по максимуму.

Мультипарадигмальність. Зокрема, Rust об'єднує в собі кілька парадигм програмування, а саме: ООП, процедурне та функціональне програмування. Останнє взято з Haskell, а ОПП — із C++. Стиль написання коду та варіанти поєднання підходів в елементах програми розробник обирає на власний розсуд.

За допомогою цієї мови створюються потужні, швидкі системи, програми, драйвери, що використовують усі можливості заліза. У першій версії мови програмування Rust об'єднано швидкість і широкі функціональні можливості C++ з надійністю мови Haskell. Rust — це мова програмування, яка чудово підходить для розробки програм під платформи, в яких не використовуються операційні системи. Наприклад, Rust використовується в ядрах операційних систем, у мікроконтролерах, інших системах низького рівня. Rust — перша мова, в якій передбачена підтримка паралельного програмування із запобіганням гонки даних. В інших мовах низького рівня подібних опцій немає [7].

Платформи для веб-застосунків Rust

Для створення веб-програм, таких як веб-сервіси, веб-ресурси та інтерфейси прикладного програмування, потрібна платформа, за допомогою якої можна здійснювати розробку цих додатків. Платформи є поєднанням інструментів, допоміжних засобів і бібліотек, які дозволяють ефективно створювати, тестувати і запускати додатки. Використовуючи платформи, можна керуватися стандартним набором правил, створювати та розгортати веб-застосунки, а також автоматизувати витрати обчислювальних ресурсів, пов'язаних з типовими діями, що виконуються в процесі веб-розробки.

Rust підтримує кілька платформ для веб-розробки, а саме: Rocket, Actix, Warp, Yew, Nickel, Gotham, Iron.

У всіх у них є свої переваги та недоліки. Розглянемо плюси та мінуси двох платформ, що стали найпопулярнішими завдяки своїй цілісності та повноті

Rocket

Rocket — одна з найзріліших платформ, доступних на Rust. На цій платформі ви можете писати швидкі та безпечні веб-додатки, не побоюючись втратити швидкість, гнучкість або зручність використання.

Переваги:

Простота у використанні — засоби генерації коду Rust широко використовуються для роботи з чистим API.

Рядки запиту — Rocket полегшує роботу з параметрами та рядками запиту.

Потокова передача — розмір не проблема, адже Rocket виконує потокову передачу всіх вхідних та вихідних даних.

Шаблонізація — Rocket має вбудовану підтримку шаблонів. Можливість розширення — створюйте свої власні примітиви і будь-яка програма на Rocket зможе їх використовувати

Типобезпека — Rocket перевіряє відповідність типів URL-адрес маршрутів, тобто. він гарантує, що помилки невідповідності типів зведені до мінімуму.

Відсутність стереотипного коду — немає потреби у стереотипному коді, адже для чистого API достатньо використання засобів генерації коду Rust.

Тестувальна бібліотека — використовуючи вбудовану бібліотеку для тестування, Rocket легко запускає модульні тести в додатках.

Конфігураційні середовища — Ви можете налаштувати програму під власну мету в середовищі розробки, середовищі обкатки та експлуатаційному середовищі.

Cookies — безпроблемний перегляд, додавання та видалення файлів cookie із шифруванням або без нього.

Дзвінки, передбачені інтерфейсом API — з вбудованою підтримкою JSON. Отримуючи Deserialize або Serialize, можна відповідно отримувати або повертати JSON.

Усунення помилок у формі — спрощене усунення помилок у формі, під час якого неправильні запити у формі відфільтровуються, щоб уникнути пошкодження коду. Отримуючи FromForm для своєї структури, ви можете дати знати Rocket, який параметр використовувати. Потім платформа проаналізує та перевірить запит у формі, створить структуру та викличе вашу функцію.

Недоліки:

Nightly — єдиний недолік платформи у тому, що вона працює лише на Nighty-версії Rust.

Actix

Actix — це платформа із серверною візуалізацією. Це означає, що процес створення та обслуговування програми здійснюється з сервера, коли сторінка запитується користувачем. Архітектура ґрунтується на дуже потужній системі акторів Rust. Вона добре підходить для написання сервісів, що мають логіку вищого рівня складності.

Щоб було легше приступити до роботи, у Actix стереотипний код, який може допомогти швидко освоїтися і включитися в роботу, або буде занадто складним для створення простої програми. Це чудова платформа з дуже гарною документацією, доступною навіть для розробників-початківців.

Переваги:

Типобезпека — як і Rocket, Actix забезпечує типобезопасность і гарантує, що помилки невідповідності типів зведені до мінімуму.

Неймовірна швидкість завдяки потужній системі акторів платформа працює на дуже великих швидкостях.

Широка функціональність — WebSockets, HTTP/2, конвеєризація, журналування та інші вбудовані функції.

Можливість розширення — є можливість створювати власні бібліотеки і будь-яка програма на Actix зможе їх використовувати.

2.2 Кросплатформний фреймворк Flutter

Оскільки використовувані платформи не еквівалентні, то якісь окремі частини коду необхідно налаштовувати під певну ОС, наприклад, під iOS, проте більша частина коду може збігатися. Це дає змогу розробникам істотно заощадити час і ресурси на створення застосунків під усі підтримувані платформи. Як мова розробки використовується мова програмування Dart.

Під час побудови застосунку Flutter транслює код на Dart у нативний код застосунку за допомогою Dart AOT (компіляція застосунку перед його запуском), який можна запускати на Android чи iOS або іншій платформі. Однак під час розробки застосунку для його прискорення Flutter використовує JIT (компіляція застосунку в процесі його запуску).

Таким чином, Flutter — це кросплатформений інструмент розробки додатків, який я шукав протягом усієї своєї кар'єри. Він поєднує в собі простоту розробки з власною продуктивністю при збереженні візуальної узгодженості на різних платформах [4].

Flutter трохи відрізняється від своїх конкурентів у деяких ключових аспектах. До аспектів, які роблять розробку застосунків Flutter кращою, ніж інші фреймворки, можна віднести такі:

Однаковий користувацький інтерфейс і бізнес-логіка на всіх платформах

Практично будь-який кросплатформний фреймворк надає можливість обміну кодовою базою між цільовими платформами. Але не існує таких фреймворків додатків, які дають змогу спільно використовувати як код користувацького інтерфейсу, так і сам користувацький інтерфейс, крім Flutter. Такий вид процесу рендерингу спрощує створення додатка, який має нативний вигляд на будь-якій платформі. Однак, використання компонентів, залежних від платформи, для візуалізації викликає потребу в шарі зіставлення властивостей для віджета платформи.

На відміну від такого підходу, Flutter не потребує будь-яких компонентів користувацького інтерфейсу для конкретної платформи для візуалізації користувацького інтерфейсу. Єдине, що потрібно Flutter для відображення призначеного для користувача інтерфейсу застосунку, — це полотно, на якому можна малювати (див. Рис.10).

Спосіб рендерингу Flutter дає змогу позбутися будь-яких проблем з узгодженістю користувацького інтерфейсу на різних платформах. Таким чином, спільне використання призначеного для користувача інтерфейсу і бізнес-логіки, яке можливе з Flutter, економить час, не впливаючи на продуктивність кінцевого продукту.

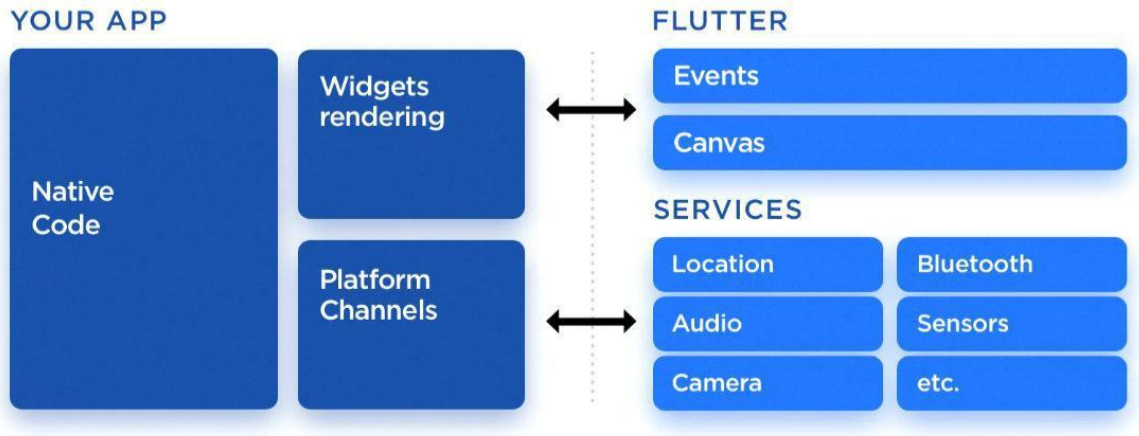

Рисунок 10 — Відображення користувацького інтерфейсу програми Flutter

Скорочення часу розроблення коду

Створення звичайного Android-додатку середнього розміру займає щонайменше 40 секунд, перш ніж його буде завантажено на тестовий пристрій. Функція "гарячого перезавантаження" Flutter, своєю чергою, дає змогу практично миттєво побачити внесені зміни, навіть не втрачаючи поточного стану застосунку. І це саме те, що прискорює розробку застосунків Flutter у кілька разів. На додачу до численних основних віджетів компонування Flutter надає великий набір віджетів Material і Cupertino, які ідеально імітують поведінку кожної мови дизайну (див.Рис.11).

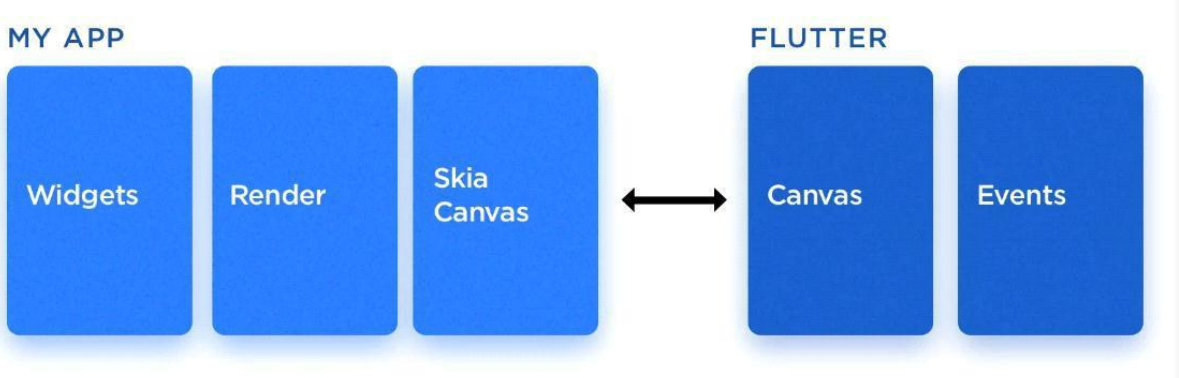

Рисунок 11 — Набір віджетів Flutter

Загалом пропускається кілька трудомістких кроків у розробці застосунків під час використання Flutter, що робить увесь процес швидшим, простішим і менш тривожним.

Збільшення термінів виведення на ринок

Фреймворк розробки Flutter працює швидше, ніж його альтернативи. Здебільшого застосунок Flutter потребуватиме щонайменше вдвічі менше людиногодин порівняно з тим самим застосунком, розробленим окремо для Android і iOS. Основна причина гранично проста: просто не потрібно писати код для конкретної платформи, щоб домогтися бажаних візуальних ефектів у вашому застосунку. Будь-який двовимірний користувальницький інтерфейс може бути реалізований у Flutter без взаємодії з аналогом нативного застосунку. Крім того, Flutter надає декларативний API для створення користувальницького інтерфейсу, який помітно підвищує продуктивність.

Аналогічно до продуктивності Native додатків

Продуктивність додатків має вирішальне значення для хорошого UX. Продуктивність застосунку Flutter здебільшого не відрізнятиметься від нативного застосунку та навіть буде кращою у складних сценаріях анімації користувацького інтерфейсу. Це тому, що на відміну від підходу більшості крос-платформних фреймворків, Flutter не покладається на будь-які проміжні уявлення або інтерпретації коду. Додаток Flutter вбудовано безпосередньо в машинний код, що усуває будь-які помилки продуктивності процесу інтерпретації.

Користувацький анімований інтерфейс будь-якої складності

Однією з найбільших переваг Flutter є можливість налаштовувати все, що ви бачите на екрані, незалежно від того, наскільки це складно. Хоча зазвичай можна зробити дуже настроюваний користувальницький інтерфейс і на власних платформах, обсяг необхідних зусиль різниться на порядок. Однак Flutter робить

процес гнучкішим і універсальнішим, не збільшуючи робочого навантаження. Загальні переходи елементів, маніпуляції з формою / кольором / тінню, обрізка, трансформації — Flutter дає змогу виконувати все це без особливих зусиль.

Власний движок рендерингу

Flutter дає змогу робити з додатками стільки речей, які недоступні на інших платформах. Очевидно, для цього потрібно, щоб фреймворк був досить потужним. Фактично, більшість пунктів, наведених вище, були б неможливими без високопродуктивного механізму кросплатформенного рендерингу. Flutter використовує Skia для рендерингу на полотно, що надається платформою. Завдяки рушію користувацький інтерфейс, вбудований у Flutter, можна запустити практично на будь-якій платформі. Іншими словами, більше не потрібно налаштовувати призначений для користувача інтерфейс, щоб перенести його на платформу, що значно спрощує процес розробки.

Проста реалізація логіки для конкретної платформи

Крім користувальницького інтерфейсу, багато реальних мобільних додатків покладаються на розширені функції рівня ОС, такі як отримання координат GPS, зв'язок через Bluetooth, збір даних з датчиків, обробка дозволів, робота з обліковими даними і т. д. Д. Багато з них доступні при розробці програми Flutter. через готовий до використання плагін, який підтримується Google.

Потенційна можливість вийти за рамки мобільних пристроїв

З Flutter можна вийти далеко за рамки розроблення застосунків Flutter для мобільних пристроїв. Також існує Flutter для Інтернету і Flutter Desktop Embeddings. Існує технічна попередня версія Flutter Web, яка дає змогу запускати чисті додатки Flutter у браузері без зміни вихідного коду. Таким чином, можна спостерігати перехід Flutter від кросплатформеної платформи мобільних додатків до повноцінного інструменту кросплатформеної розробки.

Таким чином, до переваг фреймворку можна віднести [4,5]:

- користувацький інтерфейс і логіка додатка не змінюються залежно від платформи;
- більш швидке розроблення коду;
- збільшення швидкості виведення на ринок;
- близька до продуктивності нативного застосунку;
- величезний потенціал налаштування користувацького інтерфейсу;
- окремий рушій рендерингу;
- відсутність залежності від компонентів користувацького інтерфейсу для конкретної платформи;
- підходить для будь-якої цільової платформи;
- мінімізує ризики та втрати для вашого бізнесу.

Отже, Flutter — це найшвидший спосіб створити кросплатформний мобільний застосунок, який добре працює.

2.3 Програмні засоби реалізації інтернет-магазину з продажу мисливської зброї

Для створення бекенду для інтернет-магазину обрано мову програмування Rust, оскільки вона відома своєю швидкістю, безпекою та паралельністю. Ці якості роблять Rust ідеальним вибором для створення бекенду для інтернет-магазину, оскільки вони забезпечують:

- Високу продуктивність: Rust може обробляти велику кількість запитів одночасно, що є важливим для інтернет-магазинів, які отримують велику кількість відвідувачів.
- Безпеку: Rust має вбудовані функції безпеки, які допомагають захистити інтернет-магазин від атак.
- Надійність: Rust є стійким до помилок і може продовжувати працювати навіть у разі виникнення проблем.

Перевагами цієї мови програмування над іншими є висока продуктивність та безпека. Код на Rust'і не буде поступатись в швидкодії коду на C++, але при цьому буде набагато більш безпечним.

Крім того, ефективність мови програмування виражається не тільки в швидкодії та вимогах по процесору і пам'яті, ефективність — це також менше споживання енергії при тому ж самому обсязі роботи, а отже економічність та екологічність.

Для створення серверу мовою програмування Rust обрано Rocket Framework — новий та потужний інструмент для створення сучасних веб-додатків. Rocket — це веб-фреймворк, який розроблений для Rust, сучасної мови програмування, яка відома своєю швидкістю, безпекою та паралельністю. Ця комбінація робить Rocket ідеальним вибором для створення бекенду для інтернет-магазину, оскільки він забезпечує:

- Високу продуктивність: Rocket може обробляти велику кількість запитів одночасно, що є важливим для інтернет-магазинів, які отримують велику кількість відвідувачів.
- Безпеку: Rocket має вбудовані функції безпеки, які допомагають захистити інтернет-магазин від атак.
- Надійність: Rocket є стійким до помилок і може продовжувати працювати навіть у разі виникнення проблем.

Хотіли створити інтернет-магазин, який був би швидким і реагував би на запити клієнтів. Rocket забезпечує високу продуктивність, що означає, що наші клієнти могли б швидко отримувати доступ до інформації про продукти та здійснювати покупки.

Не зважаючи на те, що Rocket —— достаньо нова технологія, втім вона була здатна задовольнити наші вимоги, пропонуючи необхідну гнучкість та величезний API. Для реалізації рівня даних здебільшого спирались на вже існуючий стек – а саме мову програмування Rust. Було обрано фреймворк Diesel, через те, що

він дає надійну та продуктивну обгортку для роботи з різними типами базами даних. Ось основні переваги використовування цього фреймворка:

- наявність обгорток під різні бази даних (отже можливість швидко змінити базу даних при потребі);
- строга типізація, що виключає можливість помилок;
- швидка розгортка бази даних за наявною схемою.

Для збереження даних обрано MySQL як просту, але також надійну і популярну базу даних. Що не менш важливо, MySQL дає можливість працювати з нею віддалено, при цьому забезпечуючи надійне шифрування та безпеку. Це дає можливість при потребі зберігати базу даних окремо від серверу, що знову ж таки підвищує безпеку.

РОЗДІЛ 3 РОЗРОБКА ВЕБ-САЙТУ ІНТЕРНЕТ МАГАЗИНУ З ПРОДАЖУ МИСЛИВСЬКОЇ ЗБРОЇ

3.1 Постановка завдання

Отже темою роботи є розробка Інтернет магазину з продажу мисливської зброї, головним завданням якого є реалізація можливості користувачів здійснювати покупки онлайн.

Додаток розроблено з використанням наступних технологій та фреймворків:

1. Rust: Мова програмування, яка використовується для написання бекенду інтернет магазину. Rust є мовою з високою продуктивністю, безпечною пам'яттю та масштабованою конкурентністю, що робить його ідеальним вибором для надійного та ефективного серверного програмування.
2. Rocket: Фреймворк на основі мови програмування Rust, який дозволяє легко створювати веб-додатки. Rocket надає потужні можливості маршрутизації, обробки HTTP-запитів та роботи з базами даних, що дозволить побудувати робочий бекенд інтернет магазину.
3. PHP: Мова програмування, яка використовується для фронтенду інтернет магазину. PHP надає розширені можливості для обробки форм, роботи з базами даних та взаємодії з бекендом.
4. Elemental UI: Фреймворк для стилізації веб-сайту. Elemental UI надає набір готових компонентів і стилів, які можна використовувати для розробки інтерфейсу користувача. Це дозволить швидко та зручно створити зовнішній вигляд інтернет магазину, забезпечивши приємний дизайн та користувацький досвід.

3.2 Загальні вимоги до веб-сайту

Типовою відмінністю будь-якого інтернет-магазину є наявність двох базових ролей (типів користувачів сайту), а саме звичайного покупця та адміністратора сайту, права яких розмежовуються. Візуальним представлення прав кожного з них є діаграма Use case (діаграма варіантів використання), яка доповнюється описом вказаних в ній прецедентів. Діаграма варіантів використання веб-сайту інтернет-магазину з продажу мисливської зброї та комплектуючих представлена на рисунку 12.

Рисунок 12 — Діаграма варіантів використання веб-сайту інтернет-магазину

Наведемо описи відповідних прецедентів.

Користувач

1. Прецедент: Перегляд товарів

Актор: Користувач інтернет-магазину.

Сценарій:

Опис: Користувач переглядає товари та здійснює вибір для подальшого перегляду та покупки.

Поведінка:

Користувач обирає категорію товарів або використовує фільтри для пошуку.

Система відображає перелік товарів з деталями.

Користувач переглядає деталі товарів та обирає необхідні для покупки.

Підсумкові умови:

Користувач має можливість обрати товари для покупки та переходу до кошика.

Інформація про товари відображається коректно.

2. Прецедент: Додавання товару у кошик

Актор: Користувач інтернет-магазину.

Сценарій:

Опис: Користувач додає товар до кошика для подальшого оформлення замовлення.

Поведінка:

Користувач вибирає товар та використовує опцію "Додати до кошика".

Система оновлює кошик користувача з вибраними товарами та показує загальну суму.

Підсумкові умови:

Товари успішно додані до кошика.

Сума у кошику відображається коректно.

3. Прецедент: Оформлення замовлення

Актор:

Користувач інтернет-магазину.

Сценарій:

Опис: Користувач оформляє замовлення на придбання обраних товарів.

Поведінка:

Користувач переходить до процесу оформлення замовлення, вводить необхідні дані для доставки та обирає спосіб оплати.

Система перевіряє інформацію та підтверджує замовлення.

Підсумкові умови:

Замовлення оформлено та відображається в історії покупок користувача.

Інформація про замовлення передана до системи обробки замовлень.

4. Прецедент: Перегляд історії замовлень

Актор:

Користувач інтернет-магазину.

Сценарій:

Опис: Користувач переглядає історію своїх попередніх замовлень.

Поведінка:

Користувач вибирає опцію "Історія замовлень" та переглядає деталі попередніх покупок.

Підсумкові умови:

Інформація про замовлення коректно відображається.

Користувач може переглядати та аналізувати свою історію покупок.

Для моделювання поведінки системи (додатка), що проєктується та аналізується, використовуються ще інші з UML діаграм, а саме діаграма діяльності (Activity diagram) (див. Рис.13), та діаграма послідовності (Sequence Diagram) (див. Рис.14). Вони дозволяють не тільки подати процес змін станів системи, але

й деталізувати особливості як алгоритмічної так і логічної реалізації операцій, які виконуються системою операцій.

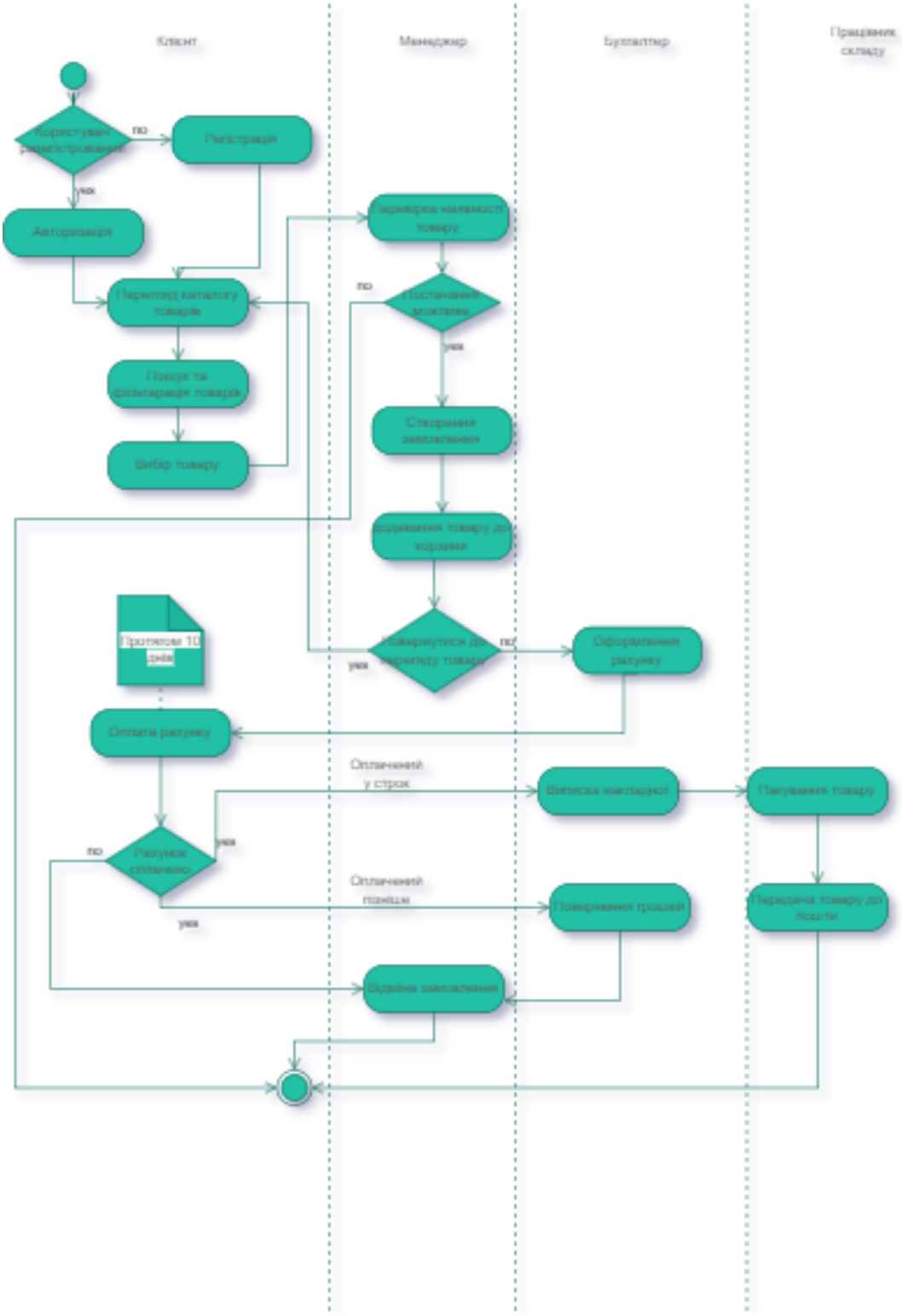

Рисунок 13 — Діаграма діяльності веб-сайту інтернет-магазину

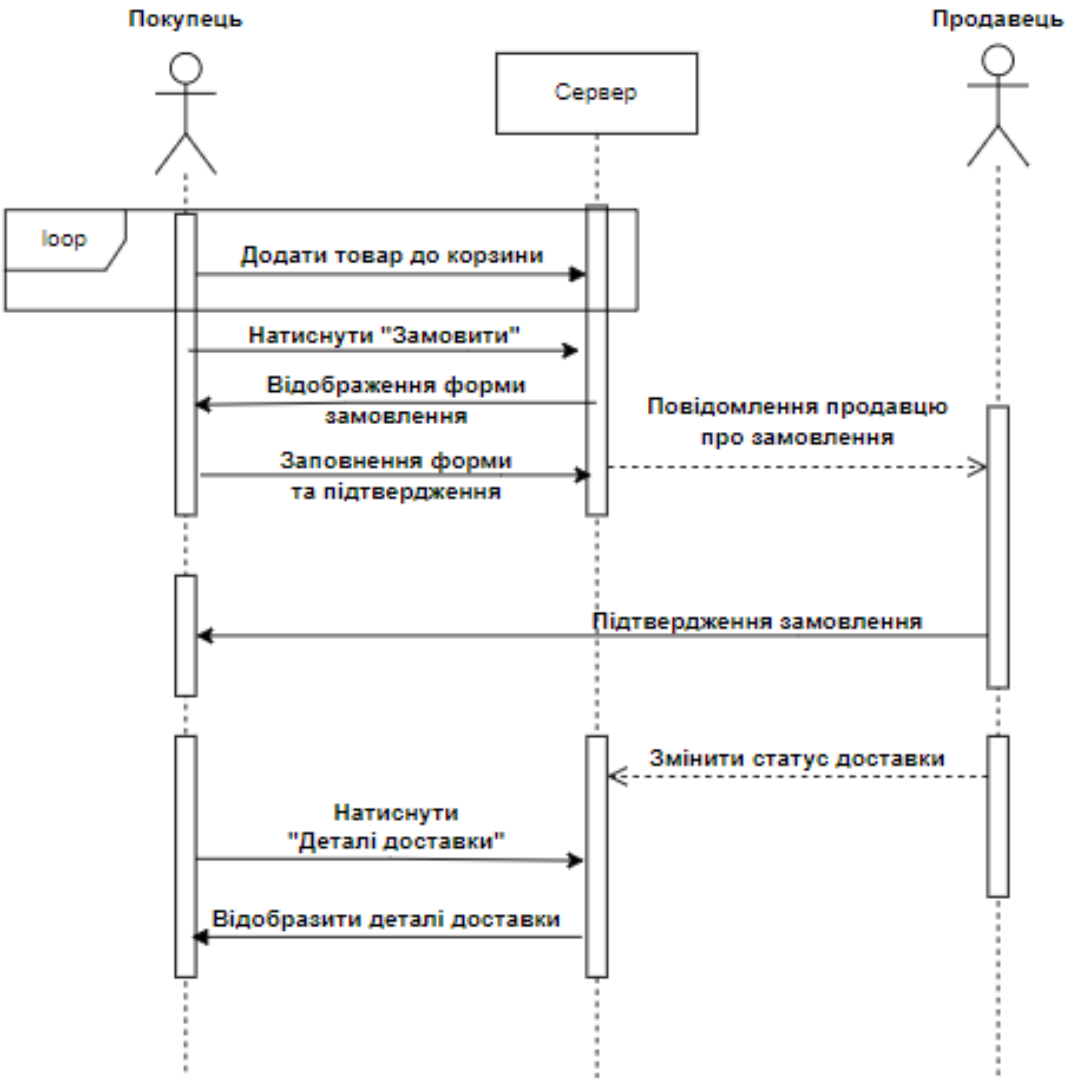

Рисунок 14 — Діаграма послідовності веб-сайту інтернет-магазину

3.3 Архітектура додатку

Згідно з визначення: «Архітектура сайту — систематизація інформації та навігації по ній з метою допомогти відвідувачам успішніше знаходити потрібні їм дані» [11]. Добре продумана грамотна архітектура сайту гарантує, що користувачі витратить менше часу на пошук потрібної інформації.

Багаторівнева архітектура

Ця архітектура реалізує ефективніший підхід до використання можливостей як серверів, і клієнтів. Типове представлення багаторівневої архітектури показано на рисунку 15. Вона підйде до великих долатків, в яких це дозволить покращити організацію та зручність обслуговування різних частин додатка.

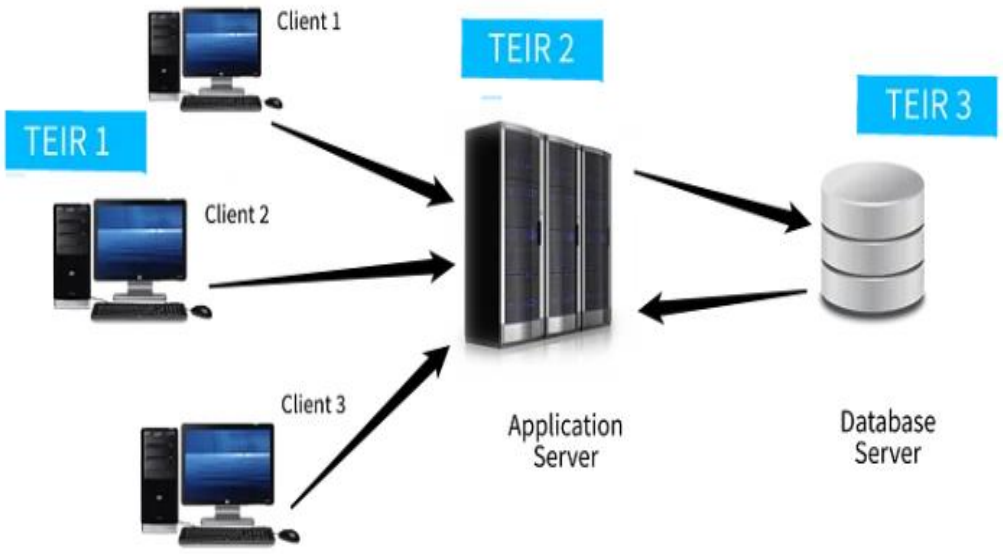

Рисунок 15 — Багаторівнева архітектура

Тришарова архітектура

Типове представлення тришарової архітектури показано на рисунку 16.

Структура комплексу управління інтернет магазином або торговельною частиною системи реалізовано у вигляді трирівневої архітектури клієнт/сервер (див. Рисунок 17).

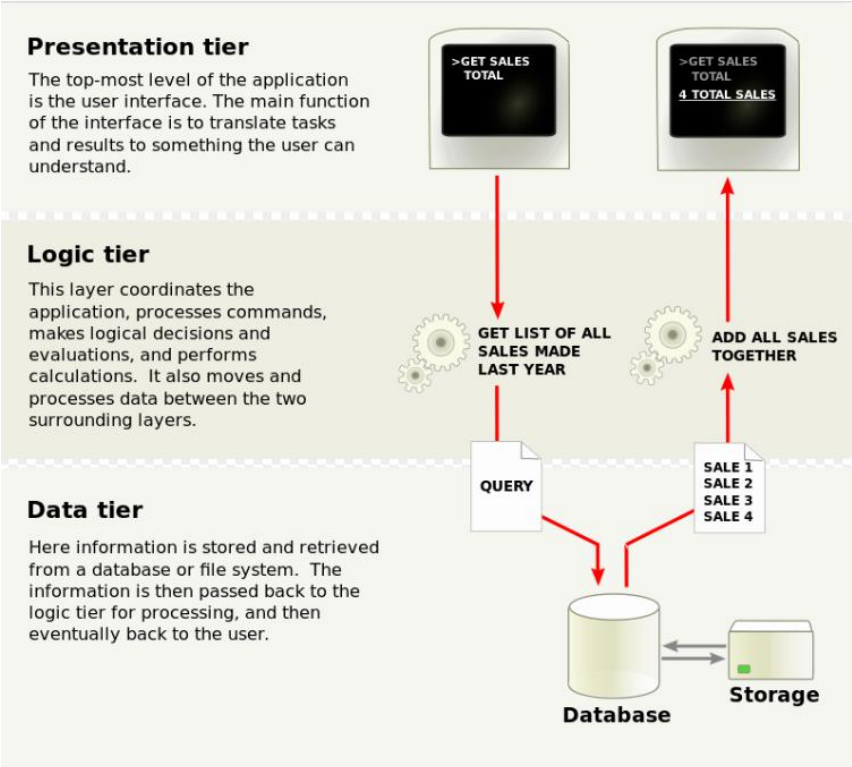

Рисунок 16 — Тришарова архітектура

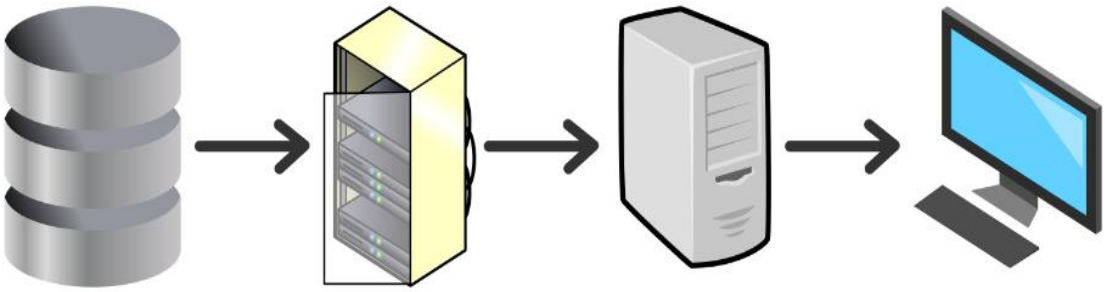

Рисунок 17 — Трирівнева архітектура

Представлений проєкт використовує трирівневу архітектуру. Трирівнева архітектура сайту - це модель, в якій всі компоненти сайту розділені на три рівні:

- Клієнтський рівень — це рівень, який взаємодіє з користувачем. Він відповідає за відображення веб-сторінок і обробку введення користувача. У вашому випадку клієнтський рівень реалізований на PHP.
- Бекенд-рівень -— це рівень, який відповідає за обробку запитів від клієнтського рівня. Він взаємодіє з базою даних і забезпечує логіку роботи сайту. У вашому випадку бекенд-рівень реалізований на Rust і Rocket.
- Даний рівень — це рівень, який зберігає дані сайту. Він відповідає за зберігання інформації, яка використовується клієнтським і бекенд-рівнями. У нашому випадку даний рівень реалізований на MySQL. Трирівнева архітектура має ряд переваг, серед яких:
- Розподіленість. Кожен рівень може бути розміщений на окремому сервері, що дозволяє масштабувати сайт і підвищувати його доступність.
- Безпека. Кожен рівень відповідає за певні функції, що полегшує захист сайту від атак.
- Зручність обслуговування. Кожен рівень може бути розроблений і підтримуваний окремою командою, що полегшує управління сайтом.

У нашому випадку використання трирівневої архітектури є хорошим вибором. MySQL — це популярна і надійна база даних, яка добре підходить для зберігання даних сайту. Rust і Rocket — це сучасні і потужні мовні технології, які дозволяють реалізувати бекенд-рівень з високою продуктивністю і безпекою. PHP — це популярна і проста в освоєнні мова, яка добре підходить для реалізації клієнтського рівня.

Ці три рівня дають абстракцію для наступного. Так, PHP рівень не знає нічого про тип бази даних, база даних — нічого, про представення даних, а користувач — про механізм обробки запитів.

На верхньому третьому рівні знаходиться віддалений сервер баз даних, він приймає інформацію від сервера додатків.

Перевагою трирівневої архітектури є:

- частини операцій перенесені на сервер додатків, що дозволило розвантажити сервер;
- видаляються зайві ходи, що дозволяє розвантажувати клієнтську програму, мінімізуючи розмір;
- єдина поведінка усіх клієнтів;
- автоматична зміна поведінки клієнтських програм.

На відміну від дворівневої архітектури, трирівнева модель має можливість ще сильніше встановити рівновагу навантаження на мережу.

Серед недоліків класичного багатошарового підходу є те, що обробка залежностей під час компіляції здійснюється зверху донизу. Тобто шар інтерфейсу користувача залежить від шару бізнес-логіки, який, у свою чергу, залежить від шару доступу до даних. Тоді шар бізнес-логіки, який зазвичай містить ключові функції програми, залежить від деталей реалізації доступу до даних (і найчастіше від наявності самої бази даних). Щоб протестувати бізнес-логіку в такій архітектурі треба вирішити ряд проблем та заповнити тестові бази даних.

Компонентне тестування в багатошаровій архітектурі ускладнене у зв'язку з тісною взаємодією компонентів. Тим не менш, при тестуванні будь-якої програми, яка працює з даними (або іншою програмою) потрібно цю залежність якось ізолювати. Наприклад, шар бізнес логіки може мати тестовий шар даних, а візуальний інтерфейс — тестовий шар бізнес логіки. При написанні тестів часто використовуються «заглушки» саме для того, щоб ізольовано перевірити якусь частину модуля.

Також було створено тестову базу даних, щоб мати можливість перевірити роботу програми від початку до кінця. Ця тестова база даних наповнювалася вже реальними значеннями і переростала у справжню.

Іншою особливістю визначення архітектури системи, що розробляється, є шаблон проектування, який використовується. Одним з найбільш поширених шаблонів у веб-розробці є MVC.

Model-view-controller

На рисунку 18 наведено графічне зображення цієї концепції.

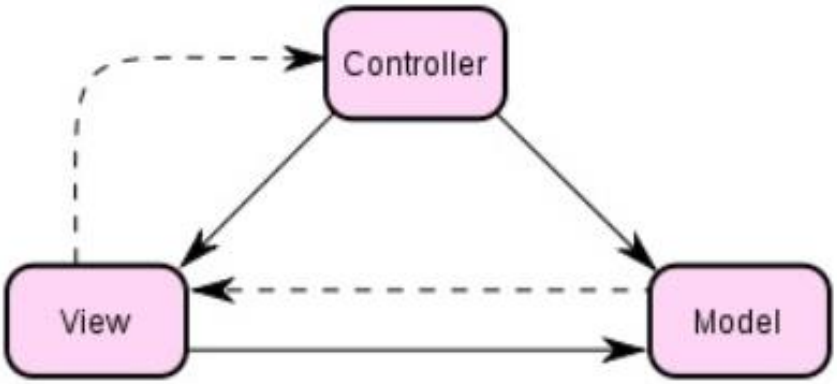

Рисунок 18 — Концепція Model-View-Controller

Дана модель визначає структуру бази даних і здійснює обробку всіх операцій, пов’язані з даними. Представлення дає можливість кристувачу взаємодіяти з додатком. Контролер знаходиться посередені між ними, і це дає користувачеві можливість бути посередником між ними. Отже, контролер обробляє запити користувачів та визначає, які саме дані повинні бути передані в представлення з моделі.

Проєктування бази даних є таким же важливим та комплексним завданням, як і проектування інших компонентів системи. Визначення структури проектованої бази є одним із першорядних завдань. Структура бази даних — це принцип чи порядок організації записів у базі даних та зав’язків між ними. У веб-платформі, що розробляється, використовується реляційна модель подання даних, що означає, що дані мають певні зв'язки між собою та організуються у вигляді таблиць, що складаються зі стовпців та рядків.

Таблиці зберігають інформацію про об'єкти, що представлені у базі даних. Відносини таблиць та їх поля подані графічно у вигляді схем діаграм класів FrontEnd та BackEnd частин додатку (див. Рис.19, 20).

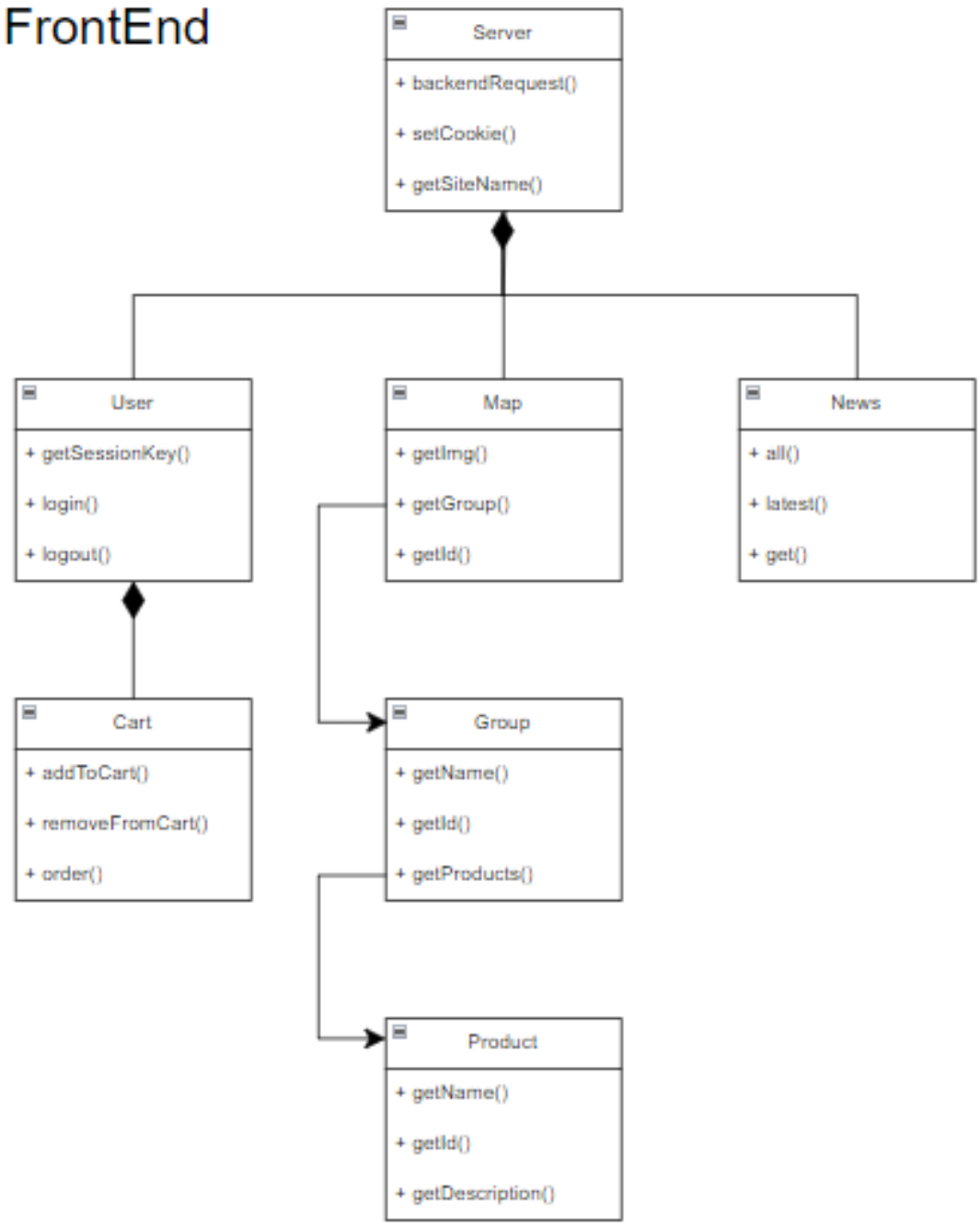

Рисунок 19 — Діаграма класів Frontend частини

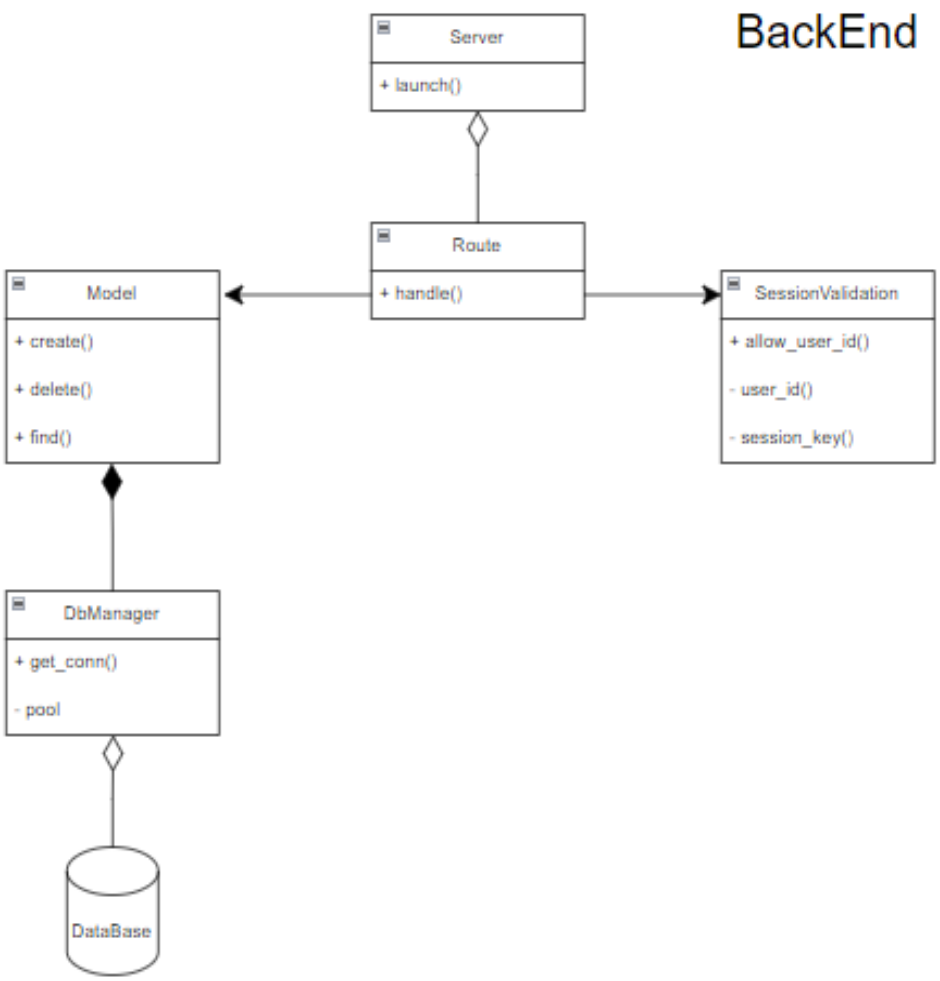

Рисунок 20 — Діаграма класів Backend частини

3.4 Інтерфейс застосунку

При створенні нашого веб-додатку керувалися принципом SER. Принцип SER (Secure, Efficient, Reliable) — це методологія розробки веб-сайтів, яка спрямована на створення сайтів, які є безпечними, ефективними та надійними.

Безпека є одним з найважливіших аспектів веб-сайту. Безпечний сайт захищає дані користувачів від несанкціонованого доступу, використання та розк-

риття. У нашому випадку використовували трирівневу архітектуру для підвищення безпеки сайту. Також використовували такі заходи безпеки, як SSL/TLS, HTTP Basic Authentication та власні сесії.

Ефективність означає, що сайт повинен швидко завантажуватись і працювати без збоїв. У нашому випадку використовували сучасні мовні технології, такі як Rust і Rocket, які дозволяють реалізувати бекенд-рівень з високою продуктивністю. Ми також використовували такі заходи для підвищення ефективності сайту, як кешування і оптимізацію коду.

Надійність означає, що сайт повинен бути доступний для користувачів 24/7. У нашому випадку використовували розподілену архітектуру, яка дозволяє масштабувати сайт і підвищувати його доступність.

Наведемо деякі конкретні приклади того, як використовувався принцип SER при створенні сайту інтернет-магазину мисливської зброї та комплектуючих:

- Для забезпечення безпеки ви використовували трирівневу архітектуру, SSL/TLS, HTTP Basic Authentication та власний механізм сесій.
- Для підвищення ефективності ви використовували сучасні мовні технології, кешування та оптимізацію коду.
- Для підвищення надійності ви використовували розподілену архітектуру, резервне копіювання та відновлення.

В даний час успіх веб-сайту багато в чому залежить від створеного інтерфейсу. Грамотно спроектований інтерфейс легкий у освоєнні, його основні функції винесені на передній план, а зайві заховані. Розташування елементів управління впливає на зручність роботи з сайтом, а його зовнішній вигляд на перше враження та задоволення від використання.

З розвитком мережі Інтернет та веб-технологій у користувачів складалися певні уявлення роботи з веб-сайтами. Подібними уявленнями можуть бути: розташування навігаційної панелі, кнопок реєстрації та входу в систему у шапці

сайту, контактної інформації, правил та іншого у футері сторінки та багато інших. Дані особливості безперечно необхідно враховувати при розробці такого веб-додатку, як сайт інтернет магазину.

Грунтуючись на на проведеному аналізі цільової аудиторії, можна виділити основні вимоги до інтерфейсу додатку:

- лаконічність та простота;

- використання нейтральних кольорів, що не викликають неприємних або тривожних почуттів;

- використання ненав'язливої, «м'якої» анімації графічних компонентів;

- адаптація інтерфейсів під різні розміри екранів мобільних додатків, оскільки саме вони поширюються на ринку.

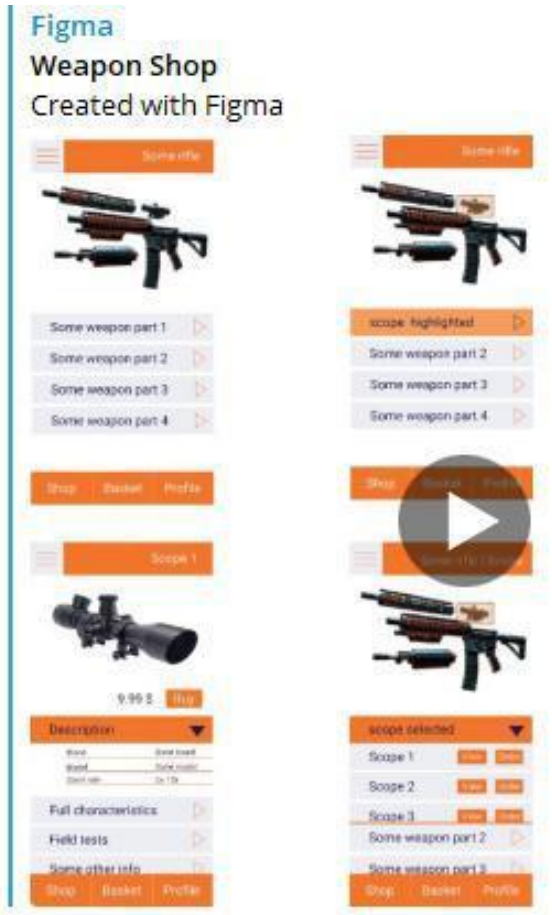

Рисунок 21 — Прототип застосунку

У ході розробки прототипу інтерфейсу веб-сайту було створено структури сторінок додатку. Прототип сайту інтернет-магазину з продажу мисливської зброї представлено за допомогою інструменту Figma. Макет однієї зі сторінок представлений на рисунку 21.

3.5 Програмна реалізація застосунку

Виділемо основні компоненти структури застунку та їх взаємозвязок.

Здійснюється запуск сервера, а саме: стається ініціалізація параметрів (зі змінних середовища), створюється з'єднання з базою даних, запускається rocket-сервер з переліком підтримуваних обробників, яким передається об'єкт з'єднання з базою даних. Реалізація запуску сервера бази даних наведена у лістингу 1.

Лістинг *1. Запуск сервера*

```
fn rocket() -> _ {
    dotenv().ok();
    SimpleLogger::new().init().unwrap();
    let db_manager = Arc::new(DbManager::new());
    let model = DieselModel::new(db_manager.clone());
    rocket::build()
        .mount(
            "/",
            routes![
                ping,
                maps,
                groups,
                group,
                group_products,
                product,
```

```
                login,
                signup,
                news_all,
                news_item,
                news_latest,
                order,
                user_orders,
                likes,
                add_order,
                add_like,
                remove_like,
                change_order_status,
                add_to_cart,
                remove_from_cart,
                cart,
                get_user,
            ],
        )
        .manage(model)
}
```

Програмна реалізація використовує мову програмування Rust та фреймворк Rocket для створення веб-додатка. Основні особливості для запуску серверу: атрибут `#[launch]` позначає функцію `rocket()`, яка є вхідною точкою для веб-додатка. Ця функція повертає екземпляр Rocket та є головною точкою входу для додатка. Реалізація ініціалізації конфігурації та логування наведено у лістингу 2.

Лістинг *2. Ініціалізація конфігурації та логування*

```
dotenv().ok();
SimpleLogger::new().init().unwrap();
```

Ці рядки ініціалізують конфігурацію з файлу .env та налаштовують простий логер.

Створюється менеджер бази даних (`DbManager`) та модель (`DieselModel`), яка використовує цей менеджер. Реалізація створення менеджера бази даних та моделі наведено у лістингу 3.

Лістинг *3. Створення менеджера бази даних та моделі*

```
let db_manager = Arc::new(DbManager::new());
let model = DieselModel::new(db_manager.clone());
```

Створюється інстанс Rocket, додаються маршрути та передаються ресурси (база даних) для використання (див. Лістинг 4).

Лістинг *4. Конфігурація Rocket та маршрутизація*

```
rocket::build()
    .mount(
        "/",
        routes![
            // Список маршрутів
        ],
    )
    .manage(model)
```

Створюється обробник для запиту, щоб дістати користувача за айді, перевіряється, чи клієнт має необхідні дозволи за допомогою ключа сесії; якщо

немає, повертається відповідь з кодом 401 Unauthorized, якщо є, то використовується модель для пошуку користувача в базі даних за user_id. Якщо знайдено одного користувача, повертаються його дані в форматі JSON. Якщо користувач не знайдений, повертається відповідь з кодом 404 Not Found.

Також визначається логіка обробки запиту для отримання інформації про користувача за його ідентифікатором. Повертається відповідь з кодом 500 Internal Server Error у разі помилки пошуку користувача в базі даних. Приклад обробника запиту наведено у лістингу 5.

Лістинг *5. Приклад обробника запиту*

```
    #[get("/users/<user_id>")]
    pub fn get_user(model: &State<ModelType>, session: 
SessionValidation, user_id: i32) -> Response {
        log::info!("Метод get_user було викликано з id {}", 
user_id);
        // Перевірка прав доступу за допомогою сесії
        if let Err(e) = session.allow_user_id(user_id) {
            return make_json(Status::Unauthorized, e);
        }

        // Пошук користувача в базі даних
        if let Ok(users) = model.find(&|user: &User| user.id 
== user_id) {
            if users.len() == 1 {
                let user = users[0].clone();
                return make_ok_json(user);  // Вертаємо JSON 
знайденого користувача з кодом 200 OK
            } else {
                return make_json(Status::NotFound, "
Користувач не знайдений");  // Користувач не знайдений
```

```
            }
        } else {
            return make_json(Status::InternalServerError, "Не
вдалося знайти користувача");  // Помилка пошуку користувача
        }
    }
```

Створюється структура для представлення інформації про сесію користувача, включаючи ключ сесії та ідентифікатор користувача, визначається перелік можливих помилок, що можуть виникнути при перевірці сесії, та реалізується трейт (інтерфейс) `FromRequest` для автоматичного витягування об'єкта `SessionValidation` з запиту Rocket. Реалізація валідації сессії наведена у лістингу 6.

Лістинг 6. *Валідація сесії*

```
pub struct SessionValidation {
    pub session_key: String,
    pub user_id: IDType,
}

impl SessionValidation {
    pub fn allow_user_id(&self, user_id: IDType) ->
ModelResult<()> {
        if self.user_id == user_id {
            Ok(())
        } else {
            log::error!(
                "User ID mismatch for session key {},
wanted user : {}, active user : {}",
                self.session_key,
```

```
                    user_id,
                    self.user_id
                );
                Err("User ID mismatch".to_string())
            }
        }
    }

    #[derive(Debug)]
    pub enum SessionValidationError {
        Missing,
        InnerFailure,
        NotFound,
    }

    #[rocket::async_trait]
    impl<'r> FromRequest<'r> for SessionValidation {
        type Error = SessionValidationError;

        async fn from_request(request: &'r Request<'_>) ->
request::Outcome<Self, Self::Error> {
            let auth_key =
request.headers().get_one("SessionKey");
            if auth_key.is_none() {
                return
Outcome::Failure((Status::Unauthorized,
SessionValidationError::Missing));
            }
            let auth_key = auth_key.unwrap();
            let model = request.rocket().state();
            if model.is_none() {
```

```
                return Outcome::Failure((
                    Status::InternalServerError,
                    SessionValidationError::InnerFailure,
                ));
            }
            let model: &ModelType = model.unwrap();
            if let Ok(session) = model.find(&|s: &Session|
s.session_key == auth_key) {
                if session.len() != 1 {
                    return
Outcome::Failure((Status::Unauthorized,
SessionValidationError::NotFound));
                }
                return Outcome::Success(SessionValidation {
                    session_key:
session[0].session_key.clone(),
                    user_id: session[0].user_id,
                });
            } else {
                return
Outcome::Failure((Status::Unauthorized,
SessionValidationError::NotFound));
            }
        }
    }
```

Структура `SessionValidation` визначає структуру для представлення інформації про сесію користувача, включаючи ключ сесії та ідентифікатор користувача.

Метод `allow_user_id` перевіряє, чи дозволено доступ до ресурсу для користувача з вказаним ідентифікатором. Повертає `Ok(())`, якщо дозвіл наданий, або `Err` з повідомленням про помилку.

`SessionValidationError` Визначає перелік можливих помилок, що можуть виникнути при перевірці сесії.

Метод `from_request` перевіряє наявність ключа сесії у заголовку запиту, витягує модель зі стану Rocket, і шукає сесію в базі даних. Визначає, чи сесія дійсна та повертає відповідний результат.

Створюється клас для роботи frontend-сервера, що має метод для виконання запиту до backend-сервера, включаючи встановлення адреси та опцій, додавання тіла запиту та ключа сесії, відправлення запиту та обробка відповіді, створення об'єкта `BackendResponse` та повернення результату. Реалізація доступу до backend-сервера наведена у лістингу 7.

Лістинг *7. Доступ до backend-сервера*

```
    public static function backendRequest($destination, 
$method, $body = null, $session_key = null)
    {
        $backend_address = self::$BACKEND_ADDRESS . 
$destination;
        $options = [
            'http' => [
                'method' => strtoupper($method),
            ]
        ];
        if ($session_key) {
            $options['http']['header'] = "SessionKey: 
$session_key\r\n";
```

```
        }

        $context = stream_context_create($options);
        try {
            $result = file_get_contents($backend_address,
false, $context);
        } catch (\Throwable $th) {
            $result = null;
        }

        return new BackendResponse($http_response_header,
$result);
    }
```

Визначається адреса `backend` шляхом об'єднання статичного `$BACKEND_ADDRESS` та `$destination`. Встановлюється опціїя для HTTP-запиту, включаючи метод. Додається заголовок та тіло запиту, якщо вони вказані. Якщо передано `$body`, встановлюється заголовок «`Content-Type: application/x-www-form-urlencoded`» та встановлюється тіло запиту. Якщо передано `$session_key`, встановлюється заголовок «SessionKey» з переданим ключем сесії.

Створюється контекст потоку на основі опцій та намагається виконати HTTP-запит за допомогою `file_get_contents`. В разі виникнення помилки захоплюється виключення та присвоюється `$result` значення `null`.

Створюється та повертається новий об'єкт BackendResponse, передаючи заголовки відповіді та результат запиту.

Створюється метод для обробки замовлення, отримується користувач та перевіряється авторизація, отримується ID користувача та виконується POST-запит до backend за вказаною адресою, передаючи дані замовлення та ключ сесії

користувача, перевіряється успішність запиту та обробляється результат, очищується кошик та оновлюється cookie, повертається HTTP-код статусу. Приклад логіки на php наведено у лістингу 8.

Лістинг *8. Приклад логіки на php*

```
public function order($order_data)
{
    $user = Server::getUser();
    if (!$user->authorized()) {
        return 401;
    }
    $user_id = $user->getId();
    $response =
Server::backendRequest("/users/$user_id/cart/order", 'POST',
$order_data, $user->getSessionKey());
    if (!$response->success()) {
        return $response->responseCode;
    }
    $this->cart = array();
    $this->items_number = 0;
    Server::setCookie('cart', json_encode($this->cart));
    return 200;
}
```

Частина html-php-лістінга сторінки для перегляду вподобань користувача; дістається об'єкт $likes від backend, якщо він порожній виводиться повідомлення з кнопкою повернутись до магазину, якщо не порожній — динамічно створюється список товарів, які юзер вподобав; в кінці додається кнопка для можливості швидко замовити всі вподобані товари. Приклад динамічного заповнення сторінки на php наведено у лістингу 9.

Лістинг 9. *Приклад динамічного заповнення сторінки на php*

```
    <div class="ui vertical stripe segment">
        <div class="ui segment middle container items
divided">
            <?php
            if (!$likes || count($likes) == 0) {
                echo <<<HTML
                    <div class="ui container middle segment">
                        <h1 class="ui middle header aligned
centered">Поки що тут нічого, самий час додати!</h1>
                        <a class="ui button black fluid"
href="/">В магазин</a>
                    </div>
                HTML;
            } else {
                foreach ($likes as $like) {
                    $product_id = $like['id'];
                    $product_name = $like['name'];
                    $product_image = $like['img'];
                    echo <<<HTML
                        <div class="ui item">
                            <div class="image">
                                <img src="$product_image">
                            </div>
                            <div class="content">
                                <a
class="header">$product_name</a>
                                <div class="description">
```

```
                                        <button class='ui icon
black button right floated'
onclick='add_to_cart($product_id)'>
                                            <i class="cart
icon"></i>
                                        </button>
                                        <button class="ui icon
black button right floated"
onclick='remove_like($product_id)'>
                                            <i class="heart
icon"></i>
                                        </button>
                                    </div>
                                </div>
                            </div>
                        HTML;
                    }
                    echo <<<HTML
                        <div class="ui divider"></div>
                        <div class="ui row">
                            <button class="ui black button
labelel icon" id="add_all_to_cart_button">
                                <i class="cart icon"></i>
                                Додати все до кошика
                            </button>
                        </div>
                    HTML;
                }
                ?>
            </div>
        </div>
```

3.6 Демонстрація веб інтерфейсу користувача

Згідно зі специфікацією до проекту, а саме у відповідності до функціональних вимог була створена клієнтська частина, в якій реалізовано весь визначений функціонал, а саме, у користувачів сайту є можливість зареєструватися (створити обліковий запис), авторизуватись (виконати вхід у систему) а також змінювати свої особисті дані. Головна сторінка сайту представлена на рисунку 22.

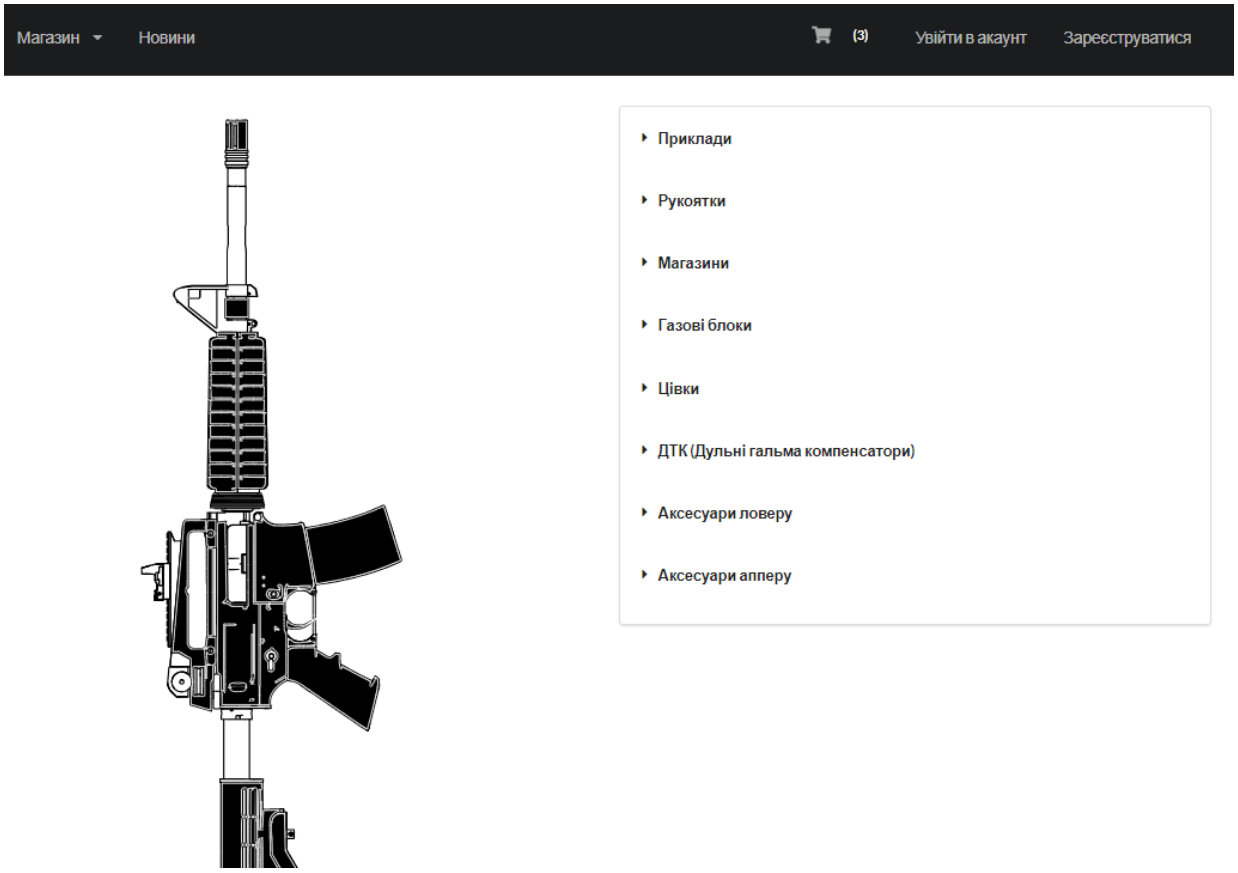

Рисунок 22 — Головна сторінка сайту

Будь-який користувач сайту має можливість переглядати доступні товари та деталі кожного товару, а також отримувати інформацію про ціни та наявність. Зареєстрований користувач має можливість додавати товари до свого кошика для

подальшої покупки та заповнити форму замовлення, обрати способи доставки та оплати, а також підтверджувати свої замовлення.

З головної сторінки сайту користувач має змогу обрати категорію товару та замовити товар, що є в наявності за обраною категорією. Сторінка сайту «Обрання товару за категоріями» інтернет магазину мисливської зброї представлено на рисунку 23.

Зареєстрований користувач може переглянути вміст свого кошика та при необхідності змінити його (додати, видалити). Сторінка сайту з кошиком представлено на рисунку 24.

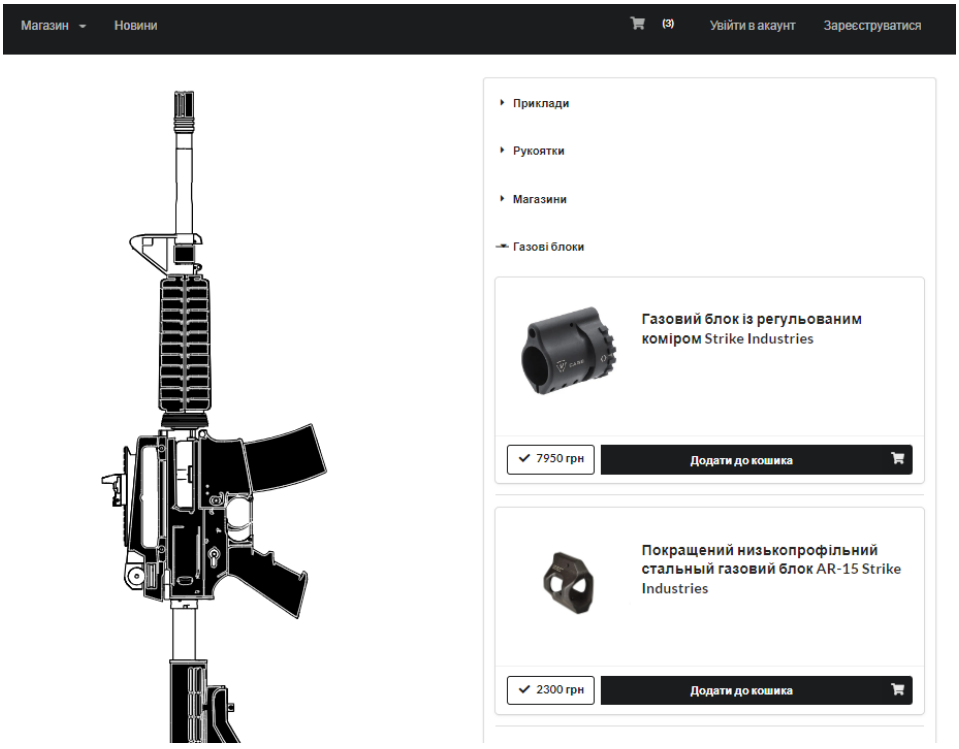

Рисунок 23 — Обрання товару за категоріями

Здійснивши пошук відповідного товару та додавши його в кошик, зареєстрований і незареєстрований користувач має можливість не тільки переглядати вміст кошика, а й змінювати його. Інтерфейс сторінки додатку з кошиком користувача представлено на исунку 23.

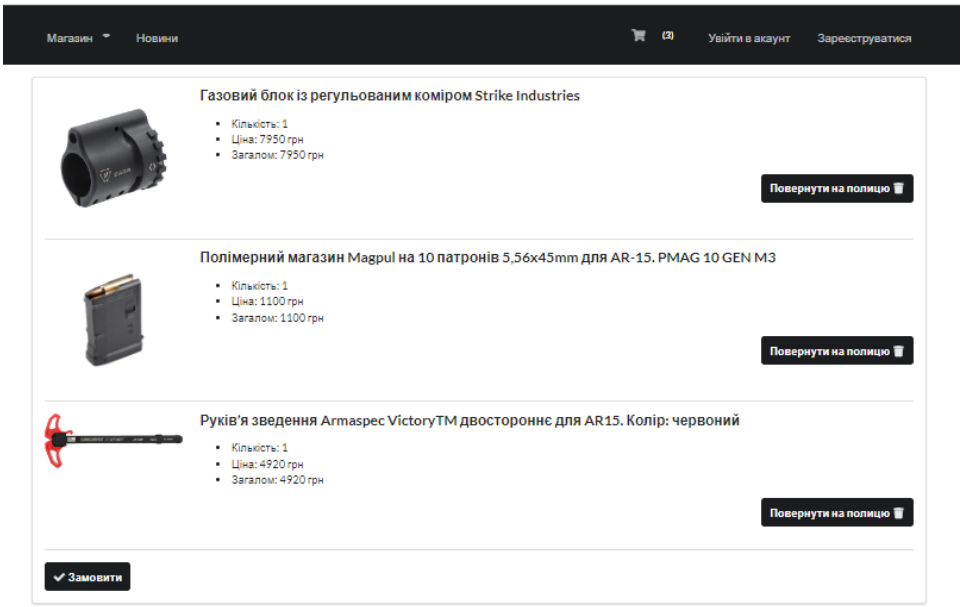

Рисунок 24 — Кошик товарів

Для зручності та залучення на сайт нових користувачів дступна сторінка новин, на якій постійно оновлюється інформація. Сторінка сайту з новинами представлена на сторінці 25.

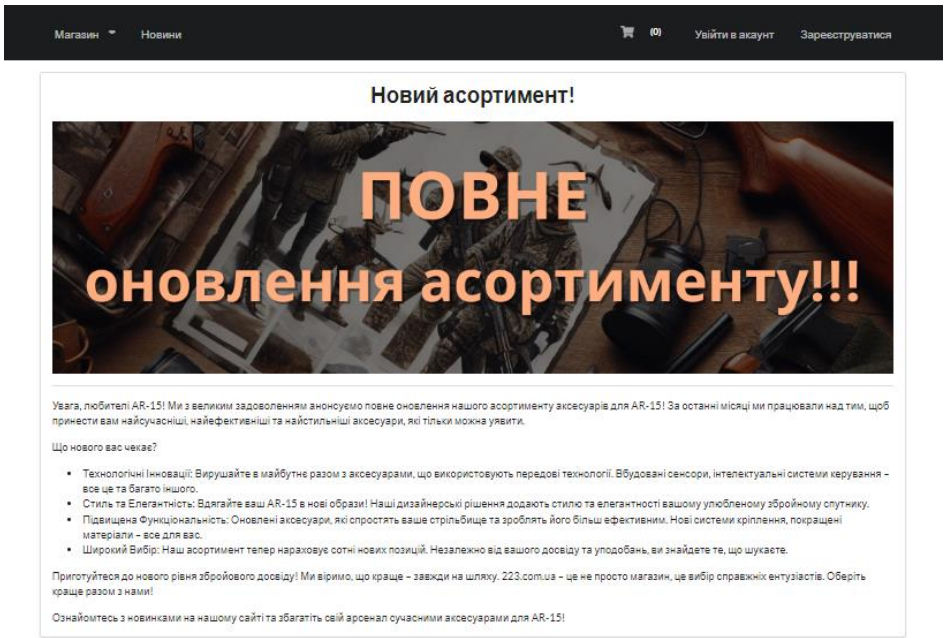

Рисунок 25 — Сторінка новин

3.6 Висновки до розділу 3

У даному проєкті використання трирівневої архітектури є хорошим вибором. MySQL — це популярна і надійна база даних, яка добре підходить для зберігання даних сайту.

Rust і Rocket — це сучасні і потужні мовні технології, які дозволяють реалізувати бекенд-рівень з високою продуктивністю і безпекою. PHP — це популярна і проста в освоєнні мова, яка добре підходить для реалізації клієнтського рівня.

Ці три рівня дають абстракцію для наступного. Так, PHP рівень не знає нічого про тип бази даних, база даних – нічого, про представлення даних, а користувач — про механізм обробки запитів.

РОЗДІЛ 4 ОСОБЛИВОСТІ ПОБУДОВИ ВЕБ-САЙТУ ІНТЕРНЕТ МАГАЗИНУ З ПРОДАЖУ МИСЛИВСЬКОЇ ЗБРОЇ

4.1 Ообливості проєктування застосунку

Графічне представлення потоку задач в процесі і з урахуванням його під процесів, включаючи специфічні роботи, інформаційні залежності та послідовність рішень і робіт наведено на рисунку 26.

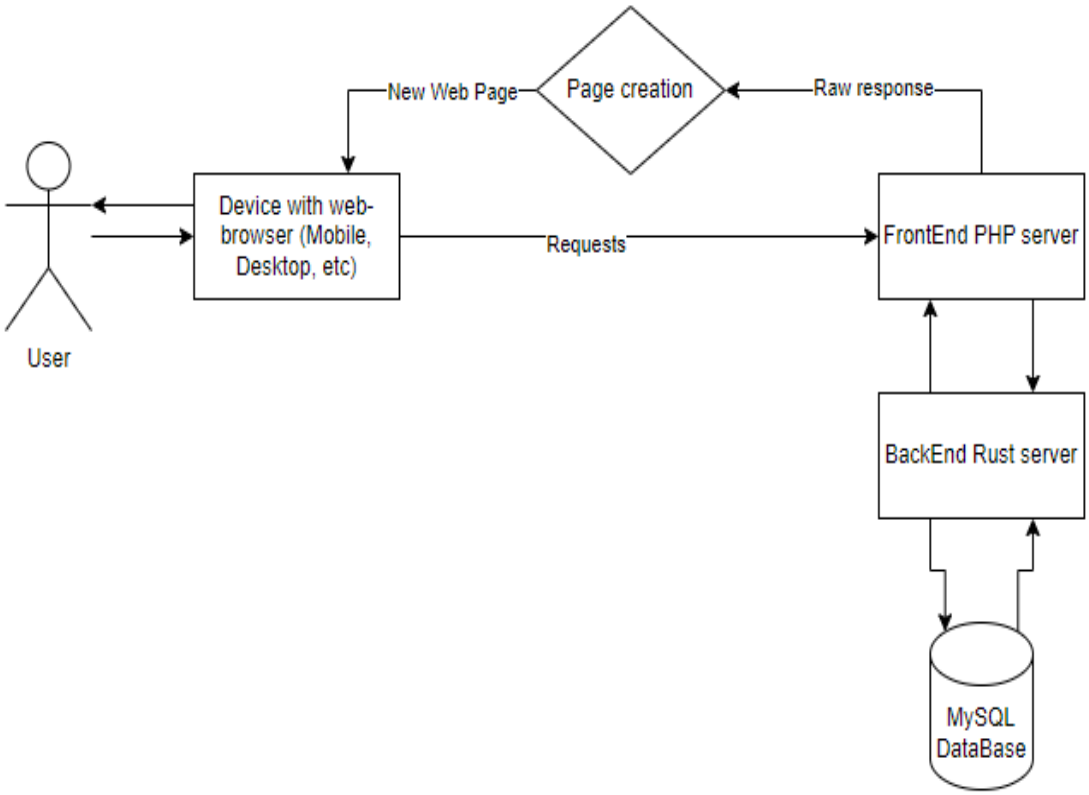

Рисунок 26 — Workflow веб-сайту інтернет-магазину для продажу мисливської вогнепальної зброї

Аналізуючи наведену послідовність рішень, слід виділити наступне:

- Користувач запитує товари в магазині та робить запити (наприклад, купує товар).

- Створення сторінки відбувається на FrontEnd сервері шляхом підстановки PHP. Це необхідно для наповнення шаблонних html-сторінок актуальними даними, отриманими з бекенду.
- Весь зв'язок (окрім Backend <-> DataBase) здійснюється через HTTP REST виклики, які базуються на стеку TCP/IP. Це означає, що сервери можуть бути запущені на різних пристроях і можуть бути легко замінені при необхідності.
- FrontEnd сервер використовується для взаємодії з користувачем. Він нічого не знає про бізнес-логіку, тому повинен взаємодіяти з BackEnd сервером для отримання та розміщення даних.
- BackEnd сервер відповідає за обробку всіх запитів від FrontEnd сервера. Він не взаємодіє безпосередньо з кінцевим користувачем і не призначений для того, щоб надавати відповіді, які можна було б прочитати людині.
- База даних використовується для зберігання продуктів, покупок, облікових записів користувачів тощо. Вона може взаємодіяти лише з BackEnd сервером, щоб зробити додаток захищеним від кібератак.

Web-сторінка буде розроблятись з огляду на адаптивність під екрани телефонів і десктопів, а завдяки модульній архітектурі — її буде в майбутньому легко замінити на натівний mobile app.

Функціональна блок-схема застосунку з описом функцій та взаємозв’язків системи наведена на рисунку 27.

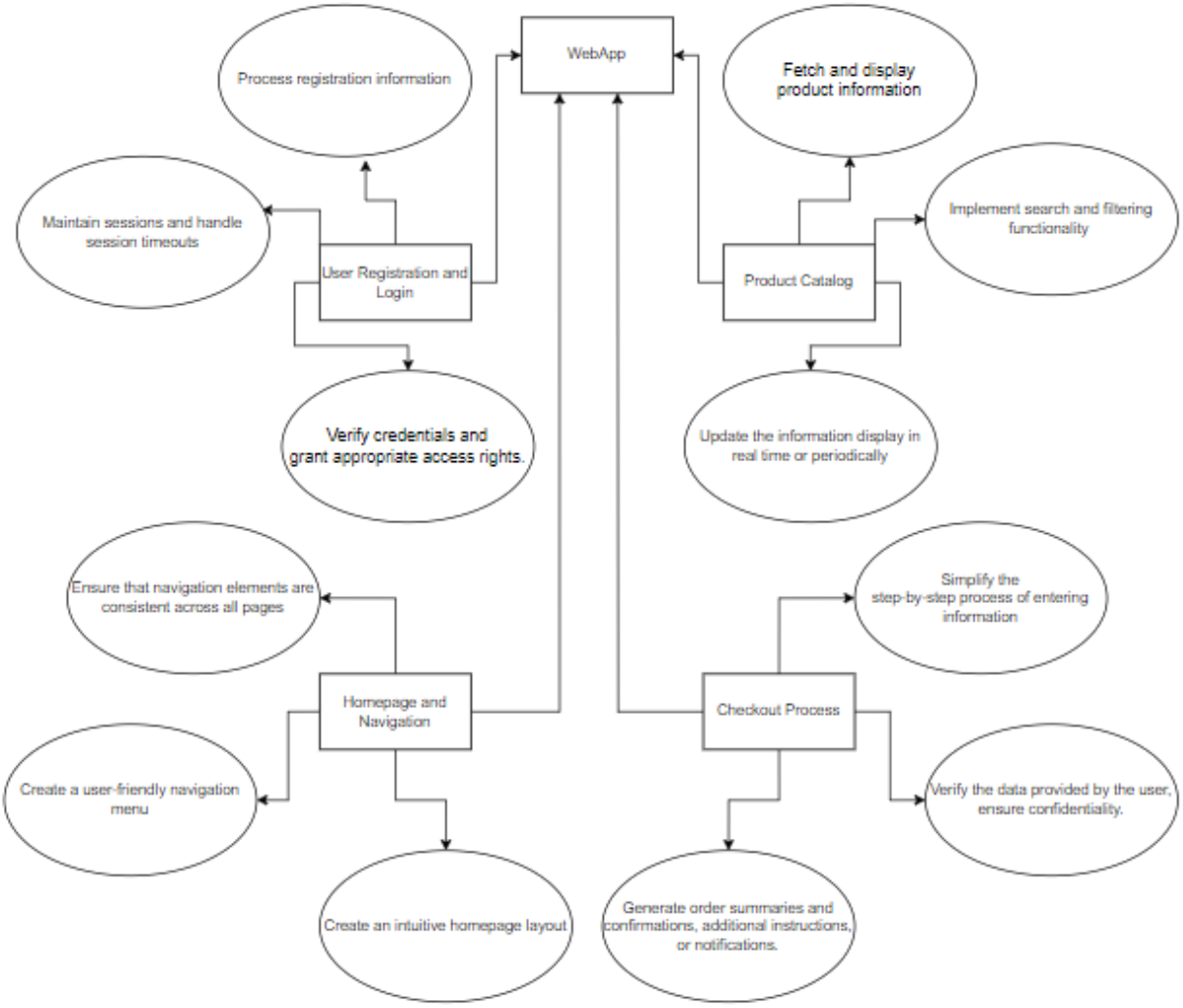

Рисунок 27 — FBD застосунку

Серед функцій, що містить вказана блок-схема можна виділити наступні:

- підтримуйте сеанси користувачів і обробляйте тайм-аути сеансів для підвищення безпеки;
- перевіряти та обробляти реєстраційну інформацію користувачів, забезпечуючи цілісність та безпеку даних;
- отримуйте та відображайте інформацію про товар з бази даних або системи інвентаризації, включно з назвою, описом, зображеннями та ціною;

- реалізуйте функцію пошуку та фільтрації, щоб користувачі могли знаходити конкретні товари за різними критеріями, такими як категорія, ціновий діапазон або атрибути;
- перевіряйте облікові дані користувача під час входу в систему та надавайте відповідні права доступу;
- реєстрація та вхід до системи;
- каталог продукції;
- забезпечити відображення точної інформації про наявність запасів, оновлюючи її в режимі реального часу або періодично, щоб відобразити зміни в запасах;
- забезпечте узгодженість елементів навігації на всіх сторінках, щоб користувачі могли повернутися на головну сторінку або отримати доступ до свого кошика та профілю з будь-якої точки інтернет-магазину;
- спростить користувачам поетапний процес введення інформації про доставку, платіжних реквізитів і вибору бажаних способів оплати та доставки;
- домашня сторінка та навігації;
- процес оформлення замовлення;
- створіть чітке та зручне навігаційне меню, яке дозволить користувачам легко отримати доступ до різних розділів інтернет-магазину, таких як категорії товарів, спеціальні пропозиції або підтримка клієнтів;
- створіть інтуїтивно зрозумілий і візуально привабливий макет домашньої сторінки, який демонструє основні продукти, акції або категорії;
- створюйте зведення та підтвердження замовлень, включаючи номери замовлень, орієнтовні дати доставки, а також будь-які додаткові інструкції або сповіщення;
- перевіряйте та безпечно обробляйте надані користувачем дані, забезпечуючи конфіденційність і запобігаючи несанкціонованому доступу або витоку даних.

4.2 Особливості графічного інтерфейсу додатку

Головну особливість («фішку») графічного інтерфейсу запозичено з географічних атласів — а саме з мапи. Подібно до географічних мап, зброю можна також розділити на області з чіткими кордонами, щоб користувач міг курсором обирати, яка саме категорія його цікавить. Це покращує досвід користувача, бо він не має перечитувати всі назви категорій, щоб знайти потрібне. Натомість знайти всі приклади на такому сайті так само легко, як тикнути пальцем в область на мапі. Графічний інтерфейс додатку представлено на рисунку 28.

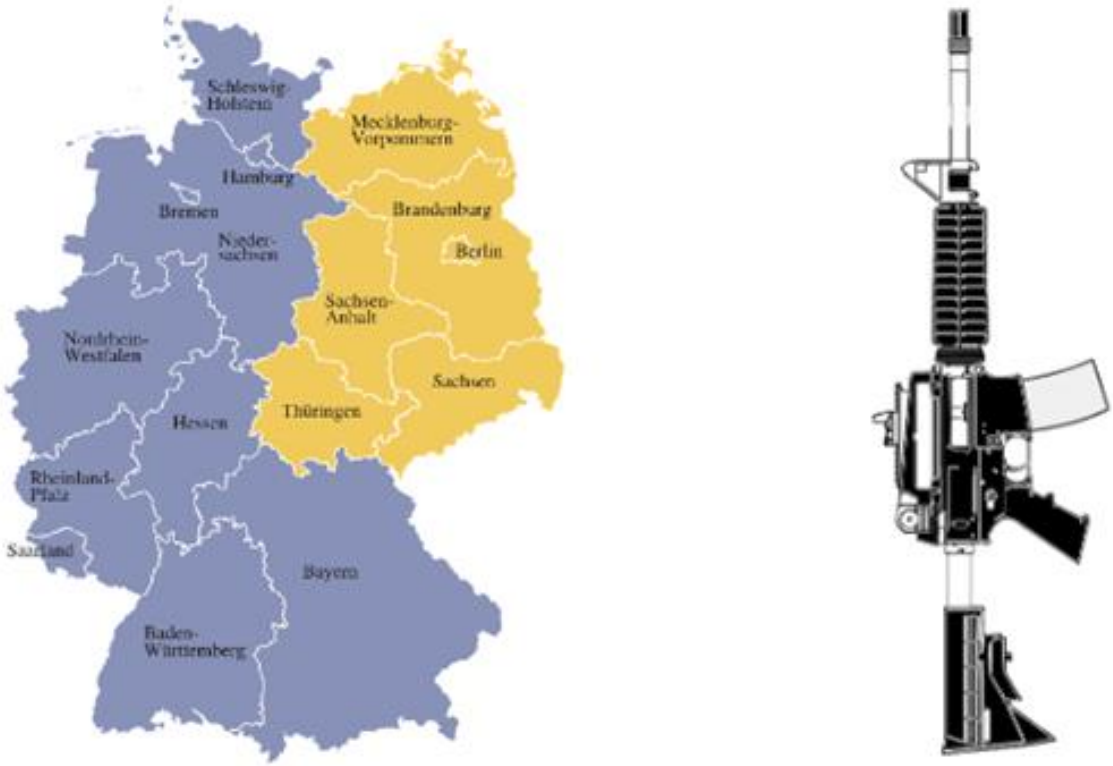

Рисунок 28 — Графічний інтерфейс додатку

Для створення мап зброї використали векторну графіку, що вбудовується безпосередньо в код веб-сторінки. Векторна графіка, як відомо, містить інструкції для побудови зображення (наприклад, проведу лінію такої довжини, такого-

то коліру, туди). Через це векторна графіка може вільно адаптуватись до різних пристроїв. Хтось може зайти на сайт з телефона, а хтось з телевізора – і всі будуть мати однаково чітке зображення.

4.3 Висновки до розділу 4

Наукова новизна

В роботі зроблено акцент на використанні нових технологій, але водночас на збереженні певного консерватизму та дотриманні принципів надійності, економічності та безпеки.

Нові технології

Технологічний стек, який використано для створення веб застосунку складається з мов та фреймворків, які:

- Є достатньо свіжими, але вже зарекомендованими в виробництві (Rust, виник 2010 року, перше застосування в 2015 ; Semantic UI - 2010, перше застосування в 2011).

- Є новими амбітними проектами, які зарекомендували себе через свою інноваційність та якість (Rocket, виник 2014 року, перше застосування в 2015).

- Є кращими версіями вже існуючий технологій (PHP8 - стандарт 2020 року).

Використання цього технологічного стеку дозволило створити конкурентний продукт по швидкодії та безпеці та більш економічне по використанню трафіку та обчислювальних потужностей (а отже і енергії). В подальшому, завдяки масштабуванню, це дасть істотну перевагу в сфері раціонального енергоспоживання та збереження довкілля (менше споживається енергії -> менше спалюється палива -> менший внесок у процес глобального потепління).

Консерватизм

Эндрю Таненбаум якось написав: «Колись інтернет був непоганим місцем, а потім хтось вирішив, що запускати на веб сторінках код це гарна ідея». Вважю, що сучасна тенденція додавати все більше функціоналу на веб застосунки, ціною обчислювальних можливостей та мобільного трафіку є хибним шляхом.

Наші користувачі не мають робити за нас роботу чи завантажувати зайве, всі дані до них приходять вже в готовому для користування виді. Таким чином ми піклуємося про клієнтів та покрашуємо їх досвід.

Щоб досягти цього відмовились від популярних фронт-енд фреймворків, таких як React, Vue, Angular, Svelte, та інших. Натомість для досягнення динамічності використано серверну мову програмування PHP, яка вже використовується для рендерингу сторінок, та додали до неї динамічність за допомогою JavaScript та AJAX.

SER

Основним стандартом, якого дотримувались при розробці є SER (Secure, Efficient, Reliable). Ним послуговувались як при виборі технологій, так і при реалізації функціоналу. Отже, дотримання певних стандартів додає розробці певний сенс і орієнтири. На початковому етапі визначили орієнтири, а потім, в процесі розробки, питали себе, чи відповідає цей крок нашим стандартам.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було поставлено та виконано наступні задачі:

1. Здійснено аналіз основної термінології об’єкта дослідження. У ході реалізації кваліфікаційної роботи було опрацьовано літературні джерела за темою розробки веб-сайтів.
2. Визначено специфіку веб-сайту інтернет магазину. Наведено основні приклади веб-сайтів в інтернеті, їх особливості. Приведено приклади реальних сайтів, розглянуто їх архітектури на прикладі інтернет-магазинів: ZBROIA, Сафарі-Україна, Stvol, Hunt Masters.
3. Визначено основні етапи створення інтернет магазину. Перед початком реалізації було проведено проектування архітектури системи та виділення варіантів використання.
4. Спроектовано застосунок та визначено його основний функціонал.
5. Розроблено веб-сайт інтернет магазину з продажу мисливської зброї.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Harris A. Boost E-commerce Sales and Make More Money: Three Hundred Tips to Increase Conversion Rates and Generate Leads : Kindle Edition, 2014. 114 с.

2. Системи електронної торгівлі. URL: https://pidru4niki.com/13710106/marketing/sistemi_formi_elektronnoyi_torgivli (дата звернення: 07.10. 20223).

3. Карпенко М. Ю., Манакова Н. О., Гавриленко І. О. Технології створення програмних продуктів та інформаційних систем : навч. посіб. Харків : ХНУМГ ім. О. М. Бекетова, 2017. 93 с.

4. Flutter & Dart — The Complete Guide [2023 Edition] URL: https://www.udemy.com/course/learn-flutter-dart-to-build-ios-android-apps (дата звернення: 07.12. 2022).

5. Flutter vs. React Native: Which is the right cross-platform framework for you? URL: https://stackoverflow.blog/ 2023/10/31/comparing-frameworks-for-cross-platform-apps-flutter-vs-react-native (дата звернення: 07.12. 2023).

6. Kirsh A. Rust vs C++ and Is It Good for Enterprise? URL : https://www.incredibuild.com/blog/rust-vs-c-and-is-it-good-for-enterprise (дата звернення: 07.12. 2023).

7. Klabnik S., Nichols C. The Rust Programming Language 1st Edition : No Starch Press; 1st Edition, 2018. 552 с.

8. MySQL Workbench : веб-сайт. URL: https://www.mysql.com/products/ workbench (дата звернення: 10.03.2023).

9. Obe R., Hsu L. PostgreSQL: Up and Running: A Practical Guide to the Advanced Open Source Database 3rd Edition : O'Reilly Media, 2017. 312p.

10. PostgreSQL : електронний ресурс. URL: http://www.postgresqltutorial.com/what-is-postgresql/ (дата звернення: 01.11.2022).

11. Richards M. Software Architecture Patterns : USA: O'Reilly Media, 2015. 350 p.

12. Мельник Р.А. Програмування веб-застосувань (фронт-енд та бек-енд): навч. посіб. Львів : Видавництво Львівська політехніка, 2018. 248 с.

13. Tweedie R. Learning FuelPHP for Effective PHP Developmentво. Livery : Packt Publishing Ltd, 2013. 104 p.

14. Drouyer S. FuelPHP Application Development Blueprints. Livery : Packt Publishing Ltd, 2015. 398 p.

15. Ніксон Р. Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS і HTML5, 2016. 510 p.

16. Zandstra M. PHP Objects, Patterns, and Practice : Apress: 5th ed. 2016. 603 p.

17. Єгорова І.М. Проектування та розробка Web-документів: навч. посібник. Харків: ХНУРЕ, 2018. 264 с.

18. Офіційний ресурс про HTML специфікацію W3C : електронний ресурс. URL: https://www.w3.org/TR/.

19. Офіційний ресурс про HTML специфікацію WHATWG : електронний ресурс. URL: https://html.spec.whatwg.org/multipage/ (дата звернення: 07.12. 2023).

20. Що таке сайт? : електронний ресурс. URL: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-sajt// (дата звернення: 07.12. 2023).

Декларація

академічної доброчесності

здобувача вищої освіти ЗНУ

Я Жаворонков Андрій Олександрович, студент 2 курсу,
форми здобуття освіти денна,
Інженерного навчально-наукового інституту ім. Ю.М. Потебні ЗНУ
Спеціальності121 Інженерія програмного забезпечення,
адреса електронної пошти se22m-06@stu.zsea.edu.ua,
підтверджую, що написана мною кваліфікаційна роб
ота на тему «Особливості побудови інтернет-магазину з продажу мисливської зброї» відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений/ознайомлена;

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

- згоден/згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою Інтернет-системи, а також на архівування роботи в базі даних цієї системи.

Дата_____ Підпис_____ А.О. Жаворонков
(студент)

Дата_____ Підпис_____ О.М. Міхайлуца
(науковий керівник)