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

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка інтернет-магазину "Bootshop" засобами HTML5, CSS3, JS та Laravel 8

Виконав: студент IV курсу, групи СН-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

(підпис) Каплун М.О. (прізвище та ініціали)

Керівник (підпис) Никитюк В.В. (прізвище та ініціали)

Нормоконтроль (підпис) Шимчук Г.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) Золотий Р.З. (прізвище та ініціали)

Тернопіль
2022

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

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

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Каплун Максим Олегович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інтернет-магазину "Bootshop" засобами HTML5, CSS3, JS та Laravel 8

Керівник роботи Никитюк Вячеслав Вячеславович, к.т.н., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 16 » березня 2022 року № 4/7-161

2. Термін подання студентом завершеної роботи 20.06 2022р.

3. Вихідні дані до роботи Літературні та інтернет джерела щодо розробки користувацької та серверної частини сайту.

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. Розділ 1. Аналіз предметної області та постановка завдання. Розділ 2. Проектування та реалізація інтернет-магазину. Розділ 3. Безпека життєдіяльності, основи охорони праці. Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титульний слайд. 2. Мета кваліфікаційної роботи, об'єкт та предмет дослідження. 3. Переваги та недоліки інтернет-магазинів. 4. Найпопулярніші рішення інтернет-комерції. 5. Варіанти використання. 6. Використані технології та програмне забезпечення. 7. Архітектура та структура інтернет-магазину. 8. Використані сутності БД. 9. Створення моделі даних та її міграція. 10. Серверна та користувацька частини сайту. 11. Можливості інтерактивної анімації. 12. Вигляд користувацької частини сайту. 13. Висновки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Гурик О.Я., доцент кафедри МТ		

7. Дата видачі завдання 24 січня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	24.01.2022	*Виконано*
2.	Підбір джерел про реалізацію веб сайту на Laravel та	04.01.2022-30.01.2022	*Виконано*
	створення складних анімацій на JavaScript.		
3.	Переклад та опрацювання джерел про реалізацію	31.01.2022-06.02.2022	*Виконано*
	моделей даних для Laravel.		
4.	Виконання дослідження щодо розробки інтернет-	07.02.2022-13.02.2022	*Виконано*
	магазинів.		
5.	Оформлення розділу «Аналіз предметної області та	14.02.2022-06.03.2022	*Виконано*
	постановка завдання»		
6.	Оформлення розділу «Проектування та реалізація	07.03.2022-03.04.2022	*Виконано*
	інтернет-магазину»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	04.04.2022-17.04.2022	*Виконано*
8.	Виконання завдання до підрозділу «Основи охорони	18.04.2022-01.05.2022	*Виконано*
	праці»		
9.	Оформлення кваліфікаційної роботи	02.05.2022-15.05.2022	*Виконано*
10.	Нормоконтроль	16.05.2022-22.05.2022	*Виконано*
11.	Перевірка на плагіат	01.06.2022	*Виконано*
12.	Попередній захист кваліфікаційної роботи	07.06.2022	*Виконано*
13.	Захист кваліфікаційної роботи	20.06.2022	

Студент _____ Каплун М.О.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Никитюк В.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка інтернет-магазину ”Bootshop” засобами HTML5, CSS3, JS та Laravel 8 // Кваліфікаційна робота освітнього рівня «Бакалавр» // Каплун Максим Олегович// Тернопільський національний технічний університет імені Івана Пулюя, факультет комп’ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СН-41 // Тернопіль, 2022 // С. – 64 , рис. – 18, табл. – 1, кресл. – 0, додат. – 4, бібліогр. – 31.

Ключові слова: PHP, Laravel, JavaScript, інтернет-магазин, спортивне взуття, анімація, веб-сторінка.

Кваліфікаційна робота присвячена розробці веб-сайту для розміщення товарів спортивного взуття з використанням фреймворку Laravel 8.

Мета роботи реалізувати веб-додаток інтернет магазину та з можливістю входу від адміністратора та клієнта. Також, інтегрувати в інтернет магазин анімації для взаємодії сайту з користувачем.

В першому розділі кваліфікаційної роботи розглянуто найкращі та найпопулярніші існуючі рішення онлайн магазинів в Україні та світи та проведено порівняльну характеристику. Також було розглянуто використані технології.

В другому розділі кваліфікаційної роботи розглянуто етапи розробки інтернет-магазину та реалізовано серверну та користувацьку частинку веб-додатку. Також було описано процедуру розміщення сайту на хостинг та запропоновано варіанти рішення помилок з якими можна зіткнутись при розробці.

ANNOTATION

Development of the online store "Bootshop" using HTML5, CSS3, JS and Laravel 8// Qualification work of the educational level "Bachelor" // Kaplun Maxym Olegovich // Ternopil Ivan Pului National Technical University, Faculty of Computer Information Systems and Software Engineering, Computer Science Department, group CS-41 // Ternopil, 2022 // P. – 64, pic. – 18, table – 1, draw. – 0, annexes – 4, ref – 31.

Keywords: PHP, Laravel, JavaScript, online store, sports shoes, animation, web page.

The qualification work is designed to develop a website for the placement of sports shoes using the Laravel 8 framework.

The aim of the work is to implement a web application of online stores and login options for the administrator and the client and to integrate animations into the online store for the interaction between the site and the user.

The first section of the qualification work considers the best and most popular solutions of online stores in Ukraine and in the world, their comparative characteristic was carried out. Moreover, the technologies used were also considered.

In the second section of the qualification work the stages of online store development are considered and the server and used part of the web application are implemented. The procedure for hosting the site was also described and the variants of correcting errors, which may be encountered during the development were suggested.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

Адмін – особа що здійснює обслуговування деякого програмного продукту.

Artisan - інтерфейс командної стрічки для Laravel.

ПЗ – Програмне забезпечення.

ООП – Об’єктно орієнтоване програмування.

ПІ – Програмний інтерфейс.

ПК – Персональний комп’ютер.

JS – JavaScript.

HTML – (англ. Hypertext Markup Language – мова розмітки для створення веб-сайтів і веб-застосунків).

CSS – (англ. Cascading Style Sheets – спеціальна мова, що використовується для опису зовнішнього вигляду сторінок).

SQL – (англ. Structed Query Language – декларативна мова програмування для реляційних БД).

DB – Data base (база даних).

PHP – (англ. Personal Home Page – однією з найпоширеніших мов, що використовуються у сфері веб-розробок).

HTTP – Hyper Text Transfer Protocol.

MVC – model–view–controller.

БД - база даних.

ЗМІСТ

ВСТУП

Актуальність теми. У теперішньому світі все більше поширюється використання веб-сайтів, зокрема інтернет-магазинів. Вони стають невід'ємною частиною життя людей. Все більше і приватних підприємців прикладають максимум зусиль, щоб задіяти свій бізнес в електронній комерції. Саме тому дана галузь в IT є однією з ключових та з кожним роком приваблює все більшу аудиторію та судячи по темпам росту цієї галузі її актуальність буде триматися на високому рівні ще довгий період часу.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Бакалавр» є:

– Проаналізувати найбільш успішні інснуючі рішення, та визначити ключові аспекти їхньої успішності.

– Спланувати розробку інтернет-магазину та визначити найраціональніші варіанти використання.

– Визначитись з середовищем розробки та обґрунтувати використовувані технології.

– Розробити архітектуру інтернет-магазину та створити модель даних.

– Реалізувати клієнтську та адміністративну частину веб-сайту, разом з інтегрованими анімаціями.

– Описати процес розміщення проекту Laravel на веб-хостингу та провести тестування розробленого програмного забезпечення.

Практичне значення одержаних результатів. Створений в результаті виконання кваліфікаційної роботи інтернет магазин на якому розміщені товари спортивного взуття. В даному проекті задіяно сучасний фреймворк Laravel, для зручної структуризації сторінок коду в проекті. Крім того на сайті розміщена інтегрована анімація, для можливості інтерактивного отримання знижки на продукцію клієнту.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Аналіз предметної області

Даний етап розробки програмного забезпечення є дуже важливим етапом розробки програмного забезпечення, оскільки включає в себе етапи, які можуть помогти вибрати та проаналізувати найкращі можливі рішення для реалізації того чи іншого завдання. Саме даний етап помагає виділити основні та дочірні процеси, головна задача яких полягає в тому щоб продукт був випущений вчасно та якісно.

1.1.1 Переваги та недоліки інтернет-магазинів

Зараз значна частина бізнесу та сфери, яка так чи інакше зв’язана з торгівлею, почали активно та посилено переходити на інтернет площадку, яка дає безліч переваг та можливостей для власників бізнесу та просто для людей, які хочуть попробувати себе в цій сфері. Розвиток інтернету та сфери IT, дав можливість простим людям попробувати надавати свої послуги на просторах всесвітньої павутини. Прикладом цього може слугувати нам відома на весь світ інтернет платформа соціальної мережі – Instagram. Спочатку дана платформа проектувалась для того, щоб люди використовували її як засіб для обміну фотографіями, але згодом сторінки на даному сайті люди почали використовувати, як інтернет-магазини, для реклами власного бізнесу, себе як особи яка може надати послуги та багато інших прикладів зв’язаних з інтернет комерцією. Звісно такий приклад не можна наводити, як приклад повноцінної інтернет торгівлі, оскільки дана платформа використовується виключно для опису товарів. Та є і перевага, яка недоступна в більшості інтернет магазинах, а саме реалізація зручної комунікації клієнта та продавця, що є дуже важливим пунктом для успішного бізнесу [1].

Отож, говорячи про переваги інтернет-магазинів над фізичними, перше що можна відмітити – це зручність у користуванні для клієнта, адже людина не повинна затрачати час на дорого до магазину, вибір товару та розмови з продавцем. Що для підприємців значно збільшує попит та клієнтську базу магазину. Другою з основних переваг інтернет-магазинів є те що для ним немає потреби орендувати приміщення, затрачати час на складання вітрин тощо. Також власник інтернет магазину не повинен жертвувати фінансами та часом на набір персоналу: адміністраторів, продавців, менеджерів, касирів, товарознавців тощо. З попередніх двох пунктів випливають такі плюси як зменшення витрат на зарплату для персоналу, виплата оренди або ж покупка приміщень та зменшення податків. Інтернет торгівля не буде займати багато часу, а також дозволяє працювати коли зручно, та не ставить людину в робочі рамки. Також однією з важливих переваг саме такого виду комерції є те що, маючи інтернет магазин та спеціалістів, які значним образом сприяють його популяризації на інтернет просторах, його власник може значним образом підвищити, клієнтську базу та мати набагато більші продажі, ніж в звичайного матеріального магазину.

Щодо недоліків, до яких може призвести торгівля на інтернет просторах, то тут більшість пунктів випливай з плюсів. А саме маючи інтернет-магазин, можна зіткнутися з такими проблемами як помилки програмного забезпечення, оскільки кожна програма через людський фактор може перестати працювати, саме це спричиняє додаткові витрати на системних адміністраторів, робота яких забезпечувати безперервну роботу веб-сторінки. При розробці програмного забезпечення такого типу слід також добре продумати алгоритм покупки, а саме подумати як буде здійснюватися оплата при покупці, для цього найкраще використовувати послуги, різних сервісів для здійснення оплати, які можуть гарантувати безпеку клієнтських даних, або ж обійти цю проблему шляхом можливістю здійснення покупок по передплаті. Адже інформація це найцінніше, чим може володіти людина в 21 столітті, а також втрата деяких клієнтських даних може призвести до кримінальної відповідальності. Також слід пам’ятати що, якою б довершеною не була площадка для продажів в інтернеті або ж власне

сам інтернет магазин, не слід забувати що обійтись без людського спілкування неможливо, адже ніщо не може замінити людину, саме з цього випливає необхідність у службі підтримки.

Незважаючи на всі недоліки, з якими може зіткнутися власник інтернет магазину дана сфера зараз з кожним днем зазнає все більшого і більшого розвитку, адже за цим стоїть майбутнє, якого не можна уникнути.

1.2 Огляд існуючих рішень

На сьогоднішній день існує безліч вартих уваги, а саме головне робочих рішень електронної комерції. Amazon- найбільш поширений та відомий. Компанія була заснована у 1994 році Джеффрі Безосом та вважається найбільшою по продажі товарів онлайн. Починав магазин з продажі книг та з часом асортимент розширився та у продаж ввійшли все можливі товари. А саме- музичні інструменти, іграшки, одяг, меблі, ювелірні вироби, продукти, косметика, побутова техніка та останні роки ще й рецептурні та звичайні ліки. Ну але й на цьому асортимент онлайн магазину не закінчується. Зайшовши на їх сайт можна знайти все, що вас цікавить. Про що компанія дійсно постаралась- це якщо клієнт не впевнений, що продукція хорошої якості, він завжди може прочитати відгуки та подивитись чесні оцінки. До того ж завжди можна зв’язатись з продавцем та задати йому питання які цікавлять стосовно вибраного товару. А саме- його характеристики, призначення та використання. Якщо товар прийшов пошкодженим чи не вчасно, достатньо подати скаргу та компанія зі всім розбереться.

Додатковим плюсом є те, що Amazon відрізняється від звичайного Інтернет магазину. Оскільки клієнт може не тільки купляти а ще й бути продавцем.

Незважаючи на величезні масштаби та популярність магазину, робота там далеко не найкраща і не проста. Зі слів співробітників умови роботи дуже жорстокі. Часто люди просто вигорають, тому що працюють значно більше чим можуть. Також, Amazon не завжди дотримується правил, через що виникають

конфлікти. Співробітників у компанії близько 1 мільйону 113 тисяч, можливо більше.

Доказом того, що данна компанія стає все більше успішнішою є її неймовірна капіталізація. Також, Amazon вважається найдорожчим брендом у світі. У 2018 році бренд оцінювався в 150 мільярдів доларів, а у 2019 році компанія змогла обійти Microsoft та отримати звання найдорожчої компанії світу. На той момент вона оцінювалась в 797 мільярдів доларів.

Rozetka- це найбільший та один з найвідоміших інтернет магазинів в Україні. Він надає можливість магазинам продавати свої товари онлайн та допомагає людям знайти все, що їм потрібно.

Медикаменти, одяг, техніка, аксесуари, продукти споживанння та інше. Це все можна знайти на сайті та зручно замовити. Доставка доступна новою поштою, кур'єром додому, у відділення або ж поштомат. Даний сервіс набирає все більшої і більшої популярності в нашій країні, доказом для цього може слугувати навіть той факт, шо будучи спочатку лише інтернет магазином, розетка починає широкомаштабне відкриття офлайн магазинів, щоб у людей була можливість прийти поспілкуватись з консультантом та підібрати для себе найбільш задовільний варіант.

Загалом на сайті доступно майже 4 мільйони товарів. Що не менш важливо, не потрібно ніякої передплати та є повернення, обмін. За потреби можна оформити продукцію в кредит. Для повної зручності було створено мобільний онлайн додаток.

Наступний не менш популярний приклад електронної комерції являється платформа OLX. Даний інтернет магазин розроблений спеціально для простих людей, які не планують розвивати бізнес в інтернеті, а просто користуються послугами даного сайту для того, що провести одноразову домовленість зв'язану з покупкою чи продажем того чи іншого товару чи послуги. За таку концепцію ведення бізнесу дана платформа і завоювала серця мільйонів українців. Достатньо зареєструватись за допомогою електронної пошти чи номеру телефону. Створюючи оголошення продавець повинен додати свої контактні

дані, опис товару та його фотографії. Найбільш зручний варіант- скачати мобільний додаток який доступний як і для android, так і для iOS. Є меню з різноманітними категоріями. Такими як: тварини, спорт, мода, відпочинок, техніка і багато іншого.

Будь хто, хто хоче позбавитись від старих непотрібних речей може створити оголошення та вказати «віддам безкоштовно» чи «обмін». Великим плюсом є те, що замовляючи товар використовуючи OLX доставка, покупець може відмовитись від товару і не платити за зворотню доставку. Усі ці витрати покриває сама платформа OLX.

На сайті можна не тільки щось придбати а й знайти роботу. Відкривши розділ «робота підібрана для тебе», особа повинна вказати регіон проживання та назву бажаної посади. Звичайно ж, надати контактну інформацію та скласти невеличке резюме. А саме: вік, освіта, професійний досвід. Для швидшого та більш точного результату існує додаткова інформація. Закінчивши якісь курси, сертифікат чи інформацію також можна додати.

На платформі OLX можна знайти навіть нового друга, а точніше домашнього улюбленця. Починаючи від єнотів та мавпенят, закінчуючи бездомними котиками та собачками які шукають люблячих господарів.

Для продавців було створено функцію де вони зможуть подивитись скільки людей переглянуло їх оголошення, скільки вподобало і хто який відгук залишив.

1.3 Планування розробки інтернет-магазину

Планування розробки інтернет-магазину слід почати з того, щоб чітко розуміти на який сегмент саме бізнесу буде розрахований дана програма. Це може бути як і магазин з можливістю продажу та покупки товару та і сторінка націлена на один вид продукції, а саме до прикладу взуття. Також варто пам’ятати, що зараз, кількість товарів та на ринку невпинно зростає, з кожним днем все більше і більше малих та середніх бізнесів просуваються в інтернеті.

Саме тому конкуренція в всіх сегментах як і в даному дуже велика, тому користувач-покупець, стає все більш вибагливішим як до сервісу так і до програмного забезпечення на якому ведеться комерція саме тому, дане програмне забезпечення повинно містити в собі відповідні вимоги для того, щоб задовільнити всі сучасні потреби. На щастя сучасні технології дозволяють реалізувати всі потреби сучасного користувача в повній мірі. Було вирішено взяти за основу середньо-статистичний інтернет магазин, розрахований на невелику кількість цільової аудиторії, оскільки вона дуже обмежується тематикою даного магазину, а саме інтернет магазин спортивного взуття.

В загальному, розробку інтернет магазину можна умовно поділити на такі етапи як: вибір ідеї, юридичне оформлення магазину, пошук постачальників, створення сайту, що в даному розділі буде поділено на під етапи, та пошук постачальників. Оскільки дана робота більше націлена на розробку програмного забезпечення, то відповідно, такі пункти, як пошук постачальників або ж інша бюрократична робота не буде виділятися в даному розділі. Більший акцент буде кинуто на розробку саме програмного забезпечення. Саме тому як оголошувалось нижче етап – розробка веб-сайту буде розділений на під-етапи, а саме: вибір програмного середовища та інструментів розробки, реалізація логіки програмного забезпечення, тобто робота з back-end, реалізація інтерфейсу користувача та просування сайту, тестування програмного забезпечення та пошук недоліків та багів та оптимізація сайту, проведення аналітики та SEO оптимізація сайту [2].

Такий етап як - вибір програмного середовища та інструментів розробки, включає в себе вибір програмного забезпечення для написання коду та обґрунтування середовищ розробки, що більш детально буде описано в розділі спеціально посвяченому даній темі.

Щодо реалізації інтерфейсу користувача, то даний етап є не менш важливим у розробці програмного забезпечення, оскільки напряму зв’язує користувача з можливістю покупки товару в інтернет магазині. Тут слід пам’ятати, що саме інтерфейс веб-сайту робить його конкурентно спроможним

на рівні інших пропозицій, завдяки таким на перший погляд простим речам як анімації, візуальний інтерфейс та маркетингові трюки зв’язані з гучними заголовками та банерами, можна добитися неабиякого результату як і в продажах так і в відвідуванні веб-сторінки. Ніколи не можна нехтувати даного складової програмного забезпечення, адже саме це динамічні елементи веб-сайту, в деяких випадках є одним з найосновнішим аспектом популярності окремих веб-сторінок.

Наступний пункт – це тестування програмного забезпечення. Даний процес є дуже важливим та в окремих випадках може займати більше часу ніж сама розробка. Оскільки цей процес полягає в тому що тестувальники, а деколи і самі розробники програмного забезпечення займаються пошуком інформації, щодо некоректності роботи або ж помилок у роботі ПЗ. Також їхня робота і включає перевірку на всі вимоги безпеки та початково запланованого функціоналу сайту. Даний процес здійснюється як зі сторони користувачів так і зі сторони замовників [3]. Хоча вище сказано, що даний процес, являється наступним після двох попередньо перечислених, але це не зовсім так. Оскільки він може проводитись, зразу після написання так званого нульового коду. Саме тому можна стверджувати що даний етан супроводжується весь цикл розробки програмного забезпечення.

Останній та заключний етап розробки інтернет магазину – це оптимізація сайту. Даний етап не слід порівнювати та путати з оптимізацією програмного забезпечення, що є більш притаманний для етапу зв’язаним з тестуванням програмного забезпечення, через те що оптимізована робота ПЗ це являється саме заслуга back-end розробки. А щодо SEO оптимізації сайту як веб сторінки, яка лежить на хостингу та шукається за допомогою одною з пошукових систем, найпопулярнішою наданий час з яких є пошукова система Google – це ряд дій спрямованих на реалізацію пошуку веб сторінки по ключових словах або ж фразах що вводить користувач у своєму пошуковому запиті. Даний етап є обов’язковим, адже без нього залучення аудиторії, користувачів або ж клієнтів веб-додатку практично унеможливлюється. Даний етап, хоч він і починається

вкінці можна сміло називати фундаментом у розробці сайту. Щоб реалізувати SEO-оптимізацію перш за все потрібно розуміти як працюють алгоритми пошукових систем [4]. До прикладу, ключові слова , які використовуються для оптимізації повинні бути в тренді, а саме подобатись користувачу та цільовій аудиторії які на перспективу може вживати дані слова, адже слова, які подобаються користувачам – це слова які подобаються алгоритму. Даний підхід називається псевдоптимізацією та з кожним днем стає все більш популярним інструментом в руках розробників та маркетологів. Також дана робота може понести за собою потенційні проблеми, оскільки практично в кожній системі пошуку є історія сайту, яка в тому числі несе за собою історію SEO-оптимізації, саме тому початково неправильно оптимізований сайти, навіть при подальшому правильному налаштуванні, можуть не подобатись пошукачам і даний проблему вже неможливо виправити. Варто пам’ятати, що якими б хорошими не були пошукові алгоритми систем, та оптимізація пошуку, як мінімум в найближчому майбутньому буде невід’ємною складовою в маркетинговій роботі [5].

Підсумовуючи даний розділ, було зроблено висновки, що при планування розробки програмного забезпечення включає в себе безліч технологій та вимагає знання різних галузей IT, зокрема веб-програмування, маркетинг, тестування та у випадку з розробкою саме інтернет магазину також і знання з електронної комерції та ведення бізнесу.

1.4 Пошук актантів та варіантів використання

Система інтернет-магазину працює за певними правилами:

- адміністратор;
- критерії товарів;
- клієнти;
- каталог товарів.

Розпочнемо з адміністратора. Пройшовши автентифікацію, дана людина може переглянути увесь список замовлень, який можна редагувати або зовсім

видалити. Також у можливості даного актора входить додавання, редагування та вилучання будь-якого товару з каталогу.

Звичайний користувач має доступ до усього каталогу товарів. Переглядаючи якийсь певний товар, є можливість переглядати детальну інформацію, фото та відео. Також на сторінці товару можна здійснити замовлення. Користувача буде переадресовано но спеціальну форму для заповнення особистої інформації.

Таблиця 1.1 – Реєстр варіантів використання

Актор	**Найменування**	**Формулювання**
Користувач	Введення даних для здійснення покупки	Валідація введених даних
	Можливість редагування корзини	Запис даних про замовлення в базу даних
	Пошук товарів	Можливість здійснити пошук за критеріями
	Перегляд товару	Можливість перегляду фотографій/відео товару
	Доступ до бази даних	Дозволяє надсилати замовлення адміністратору
Адміністратор	Доступ до товарів	Дозволяє створювати та редагувати товари
	Доступ до замовлень	Дозволяє переглядати, видаляти, та редагувати замовлення
	Доступ до категорій	Дозволяє переглядати, видаляти, редагувати і створювати категорії
	Доступ до бази даних	Дозволяє виконувати запити до бази даних

До функціональних об’єктів входять користувачі та адміністратори. Адміністратори володіють усіма привілеями, тобто вони можуть редагувати, добавляти, видаляти товари. Також у них є повний доступ до листа замовлень, мають право редагувати та видаляти. А ось користувачі не мають таких привілеїв, у них є доступ тільки до каталогу товарів. Вони можуть лише переглядати відомості та оформити замовлення. Таким чином наповнюючи базу даних замовлень зроблених з сайту.

1.4.1 Варіанти використання

Мета даного підрозділу – це візуально продемонструвати, як кожен з користувачів даного веб застосунку буде використовувати його. Як було вияснено вище, користуватись даним сервісом може два типа користувачів – адміністратор, та звичайний клієнт. Щодо двох цих ролей, то клієнт це безпосередній користувач веб сайту, тобто це головний споживач контенту та безпосередньо людина для якої розроблявся даний веб-застосунок, а в свою чергу адміністратор це обслуговуючий персонал інтернет магазину, який має у своєму арсеналі дещо привілейованіші та розширеніші функції, такі як: редагування видалення та додавання всієї інформації зв’язаної з товарами. Даля візуального опису функціоналу можливих варіантів використання буде використано сервіс draw.io. На рисунку 1.1 зображено UML діаграму для клієнта інтернет магазину.

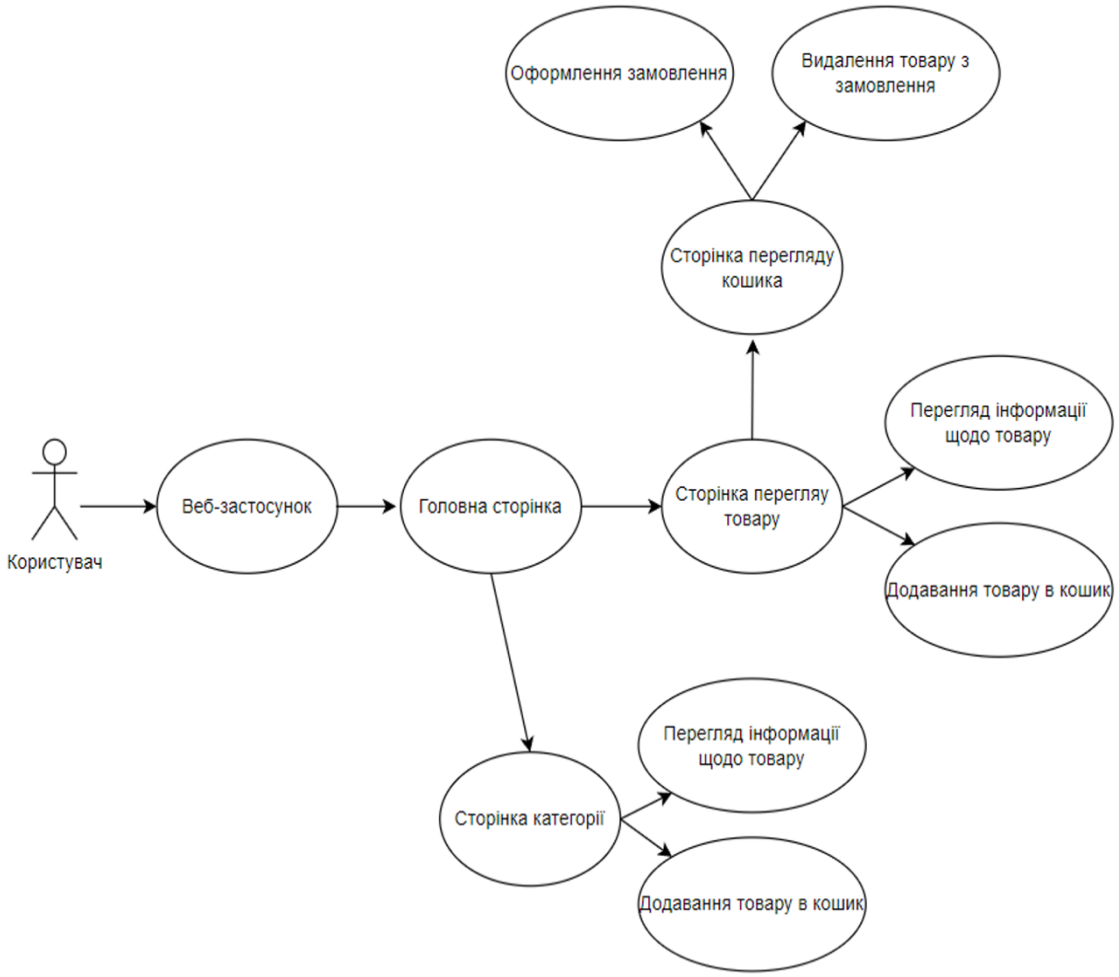


Рисунок 1.1 – UML діаграма для користувача

Для того щоб адміністратор мав набагато більш привілейованіший функціонал, потрібно реалізувати систему окремих сторінок, які будуть мати можливість редагувати всю необхідну інформацію. Дану систему прийнято називати сторінка адміністратора, доступ до неї буде мати лише обмежене коло людей. Реалізація адмін панелі буде розглянута в подальших розділах, але функціонал даної частини наведений на рисунку 1.2.

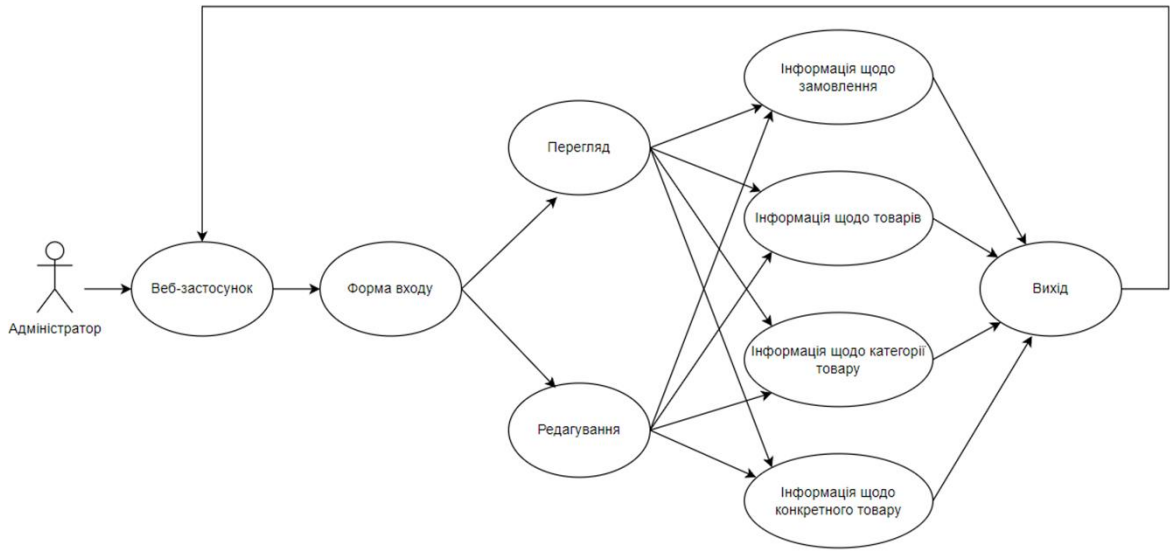


Рисунок 1.2 – UML діаграма для адміністратора

Як можна побачити на рисунку для адміністратора інтернет-магазину будуть доступні такі функції як редагування товарів, категорій та замовлень з їхніми наповненнями, також для адміністратора доступні всі сторінки користувача, а його адміністративна панель захищена логіном та паролем реалізація яких буде описана в наступних розділах.

1.4.2 Методи розв’язання задачі

Щоб виконувались всі задачі інтернет-магазину, було розроблено веб-сайт, на якому використовується база динних, з якої можна завантажуються увесь контент для каталогу. Усю важливу інформацію необхідно зберігати у базі даних, такі як: дані що до каталогу та його наповнення, замовлення клієнтів. Окрім зазначеного можна виділити також і сторінку з контактною інформацією. У другій же частині продемонстровано панель адміністратора, саме тому було вирішено використовувати два окремі підрозділи сайту – адмін частину(невидиму для користувача та дуже корисну для адміністратора) та користувацьку частину(Єдину частину доступну для користувача та непотрібну для адміна, оскільки всі необхідні функції та навіть більше він може виконати у

своїй частині). До додаткових функцій адміністратора можна віднести насичення сайту медіа контентом, а саме фото та відео товарів. Додавання нових товарів, їх редагування та виконання тих самих операцій із замовленнями та категоріями товарів.

1.4.3 Оптимальний метод розв’язання задачі

Сам веб-сайт містить чотири шаблонних сторінок: головна сторінка, перегляд товару, сторінка адміністратора та сторінка замовлення. Базова сторінка є фактично загальною конструкцією з footer та header для реалізації першої частини, у якій містяться головна сторінка, сторінка для замовлення товару, сторінка з формою для входу у панель адміністратора. Головна сторінка є ключовою, на якій відображається доступні товари. Сторінка з створенням замовлення є формою, яку потрібно заповнити особистою інформацією. Відправлену форму може переглядати та редагувати адміністратор. Також побічними сторінками шо до головної є сторінки відображення товарів по критеріям. Сторінка відображення товару є окремим елементом. Що до сторінки з входом у панель адміністратора немає ніякого значення для звичайних користувачів, адже там нема можливості реєстрації. Тільки адміністратор з паролем зможе увійти. У самій панелі адміністратора є можливість редагувати та додавати новий товар у каталог. Важливою функцією також є перегляд та редагування усіх замовлень від клієнтів.

1.4.4 Життєвий цикл ПЗ

Щодо життєвого циклу у розробці програмного забезпечення. То даний процес можна розділити на декілька етапів. Перший етап характеризується вивченням актуальності ПЗ та програмних рішень. Наступний етап полягає в тому що встановлюються терміни розробки того чи іншого кроку розробки ПЗ. Також узгоджуються всі витрати як матеріальні так і трудові. Третій етап – це

безпосередньо розробка програмного забезпечення, на цьому етапі вирішуються всі поставлені задачі та вимоги, також шукаються найкращі шляхи для вибору та обґрунтування технологій розробки. Та четвертий етап – це безпосередньо реалізація програмного забезпечення, що є найважливіший та найважчий з всіх вищеперечислених, адже потребує найбільше ресурсів. Після цього йде етап тестування, на якому проводиться корекція та оптимізація деяких рішень. Та завершальний етап – це технічне обслуговування програмного забезпечення, даний етап, в окремих випадках, не припиняє своє існування практично ніколи, прикладом такого випадку може слугувати обслуговування інтернет магазину, де товар та інформацію практично завжди приходиться коригувати, в залежності від сезону та продукції.

1.5 Вибір середовища розробки

Середовищем для розробки було редактор коду Atom, так як у ній є підтримка всіх необхідних засобів та комфортний інтерфейс для роботи із будь-яким програмним забезпеченням. Було обрано такі засоби: HTML 5, CSS 3, JS, PHP та MySQL.

PHP – мова програмування початково створення для генерації HTML сторінок. Даний інструмент є одним із найпопулярніших інструментів для веб-розробки. Переваги даного рішення є те що є безліч хостинг провайдерів, які підтримують цю мову програмування. Також у файл із розширенням php можна інтегрувати html код, без використання фреймворків та інших допоміжних бібліотек зі своїм кастомним розширенням, як це робиться в проектах на JavaScript [6].

Для того щоб реалізувати необхідні анімації та поведінку інтерфейсу для користувача, було використано JavaScript. Дана мова програмування є однією із найпопулярніших на сьогоднішній день та має безліч переваг, з точки зору веб-програмування. Найперше потрібно визначити швидкість її роботи, оскільки дана технологія не потребує першочергової компіляції, реалізовується за

допомогою JIT, який підтримується більшістю сучасних браузерів. Також швидкість реалізовується за допомогою того факту, що дана мова у випадку цього проекту, реалізовується на стороні клієнта тому не буде зайвий раз навантажувати сервер. Також дана технологія може вважатись відносно молодою, саме тому проекти будуть підтримуватись довго та не будуть втрачати свою актуальність. Але також треба пам’ятати про недоліки, які несе за собою JavaScript, а саме невисокий рівень захищеності, адже всі дії ведуться на стороні клієнта. Проте дана проблема не буде зачіпати конкретно даний проект, адже використовуватись мова програмування буде, лише для візуальної частини сайту [7].

Також щодо несуттєвих та не найважливіших інструментів розробки, в проекті буде використовуватись програмне забезпечення AdobePhotoshop – графічний редактор, розроблений компанією AdobeSystems. Дана програма буде використовуватись для візуальної розробки елементів сайту, прикладом для цього може слугувати логотип інтернет-магазину або ж фото товарів виконаних на білому фоні.

1.6 Обґрунтування використовуваних технологій

Використання HTML5 та CSS3 необхідне для розробки візуальної частини сайту. Даний інструмент є практично незамінним у веб-розробці, і у будь яких проектах зв’язаних з такими продуктами як сайти або ж веб програми, розробники так чи інакше звертаються до даної технології.

Основною ж технологією, яка буде використовуватись для написання серверної частини інтернет магазину – стане PHP. Про цю мову програмування було описано в попередньому розділі. Також для PHP написана велика кількість фреймворків та бібліотек. Одним з таких є фреймворк Laravel. Laravel – створений спеціально для великих проектів на PHP для того щоб реалізувати, саме шаблон model–view–controller. Де model виступає саме модель, яку потрібно використати для того щоб здійснити деякі маніпуляції з базою даних. View –

елемент, який використовується для того щоб шаблонізувати види сторінки [8]. Переваги, які несе в собі таке програмне рішення це реалізація так званого односторінкового веб-сайту, що в теорії мало б зменшити навантаження на баузер при завантажені сторінки, але все ж в реальності даний фреймворк буде використовуватись для того щоб забезпечити проекту так званий вид синтаксичного цукру, тобто завдяки цьому самий синтаксис стане більш зрозуміліший для користувача [9]. Але слід розуміти що не можливо досягнути ідеального вигляду структури, тому дане поняття є достатньо умовним, оскільки практично кожен проект має достатньо багато залежностей, які не можуть по тій чи інакшій причині бути структуризованими, саме тому такі залежності будуть винесені в привичному порядку та не будуть вважатися базовими для проекту.

Отже наступним пунктом після серверу веб-сайту – це зберігання та виведення інформації. Саме тому було вирішено використати технологію SQL та систему керування базами даних MySQL [10]. Щодо SQL то дана мова програмування один із найкращих варіантів для взаємодії із БД. Також важливо, що саме SQL забезпечує роботу з реляційними базами даних, що несе за собою такі переваги як: простота, що дуже важливо для того хто в майбутньому буде працювати над підтримкою програмного забезпечення, також серед переваж можна виділити проста реалізація відношень, яка у рамках даного проекту буде реалізовуватись за допомогою дублювання ключів сутностей [11].

Для того щоб запустити SQL сервер буде використано програму MAPM або ж більш функціонального брата MAPM pro. Дане програмне забезпечення є незамінним для операційної системи Mac OS. Детальніше про MAMP буде описано в розділі про реалізацію інтернет магазину.

Що до здійснення запитів до бази даних, то тут можна відмітити декілька варіантів підключення до БД таких як PDO, простий PHP та підключення яке вбудоване в Laravel [12]. В подальшому розвитку даного проекту такий інструмент як PDO підключення до бази даних не буде потрібно, оскільки в Laravel, як зазначалось вище має своє зручне підключення до бази даних, яке

реалізовується за допомогою файлів конфігурації та моделей даних, що є дуже зручно та потребує набагато менше часу для подальшої роботи з даними.

1.7 Висновок до першого розділу

В першому розділі кваліфікаційної роботи було пророблено роботу над аналізом предметної галузі, а саме було розглянуто найпопулярніші інтернет-магазини в світі та Україні. Було зроблено висновки щодо тенденції популяризації інтернет комерції та визначено її переваги та недоліки. Також в даному розділі було пророблено етап складання технічного завдання та вимог, а саме розроблено діаграму використання ПЗ та описано його життєвий цикл. До вище сказаного, було вибрано середовище розробки та програмне забезпечення, яке буде використовуватись для виконання кваліфікаційної роботи. Також було описано, вибрано та обґрунтовано вибір використовуваних технологій для інтернет-магазину.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ ІНТЕРНЕТ-МАГАЗИНУ

2.1 Моделювання архітектури та структури інтернет-магазину

Як було зазначено в меті роботи, після етапу моделювання – йде етап розробки програмного забезпечення. В даному розділі розглядаються такі аспекти виконання, як моделювання архітектури самого програмного забезпечення, використання модулів та розробка програмного забезпечення та розробка моделей даних.

2.1.1 Моделювання архітектури інтернет-магазину

Архітектура програмного забезпечення є надзвичайно важливим та основним етапом у розробці програмного забезпечення. Даний процес являється фундаментальною та її невід'ємною частиною. Існує чимала кількість визначень, які описують таке поняття як архітектура програмного забезпечення. Було зроблено висновок, що архітектура – це розділення системи на структурні елементи. В подальшому ці структурні рішення не будуть змінювати свою структуру та ідеологічну функціональність впродовж всього життя програмного забезпечення, що підтверджується тим що дані елементи є взаємнопов'язаними та залежать один від одного [13].

Архітектура програмного забезпечення складається з трьох ключових аспектів, а саме:

- структура;
- поведінка;
- стиль.

Структура – власне характеристика яка говорить, про складові частини архітектури, про функції, які ця складова виконує та її властивості. Така складова як поведінка, виконує дуже важливу роль, а саме показ взаємозв'язків між структурними елементами архітектури. Прикладом для цього можуть бути

клієнтські запити на сервер або ж відповіді сервера клієнту. Та складова стиль – яка відповідає за основні правила керівництва архітектурою та її складовими. До правил архітектури можна віднести такі аспекти як: написання загальних тенденцій керівництва, що встановлюють основні обмеження та рекомендовані рішення в тій чи іншій ситуації, також даний аспект є дуже важливим тоді коли настає час прийняття рішень, які напряму будуть відповідати за розвиток або розширення програного забезпечення, саме тоді розробник буде спиратись саме на аспект стиль архітектури [14].

Щодо даного проекту, а саме розробка інтернет-магазину спортивного взуття, то було вибрано до використання одну із найпопулярніших та найпоширеніших у світі типів архітектури, а саме – клієнт-серверна архітектура. Даний тип архітектури полягає у тому що кожна дія або ж функція поділяються на два типи: дії які виконує сервер та дії які виконуються на клієнтській стороні.

Переваги які несе за собою дана архітектура – це перш за все широке поширення та простота у використанні. Також дана архітектура може дати дуже зручно концепцію, яка називається «Слабкий клієнт». Основна перевага даної концепціє – це те що на стороні клієнта при реалізації даної архітектури, немає потреби у сильному користувачеві, оскільки всі складні операції з використання бази даних та всі обчислення будуть вестися на стороні сервера. Більш детально про основні концепції буде описано згодом у даному розділі.

Для того щоб клієнт-серверна архітектура працювала та виконувала всі необхідні задачі та функції, потрібна наявність трьох типів компонентів. Перший тип – сервери, прикладом цього компоненту може слугувати сервер на якому розміщуються всі компоненти веб-сайту. Другий тип – це мережа та протокол, за допомогою яких передаються віх необхідні данні між клієнтом та сервером. У даному проекті буде використовується найрозповсюдженіший протокол, а саме – HTTP протокол. Даний мережевий протокол використовується в переважній більшості веб-сторінок, і відповідає за передачу гіпертекстових документів. Передача інформації здійснюється за схемою запитів та відповідей, що цілком

задовільняє стиль саме вибраної архітектури. Та останній тип необхідний для вибраної архітектури – це клієнт, який буде взаємодіяти з сервером [15].

Клієнт-серверна архітектура програмного забезпечення поділяється на два типи:

– дволанкова клієн-серверна архітектура;
– тьохланкова клієн-серверна архітектура.

Суть двохланкової архітектури полягає у тому, що при її реалізації існують дві ланки, які передають інформації один до одного – сервер та користувач. Особливість даної реалізації являється те, що при ній особливо чітко можуть бути виділені два типа клієнтів: сильний та слабкий. Слабкий клієнт використовується тільки для представлення тих чи інших сторінок та звертається до сервера тільки для того щоб отримати інформацію. У свою чергу сильний клієнт відрізняється від попереднього тим що може не тільки оперувати представлення, а ще й керувати даними на сервері, тобто додавати зміни.

Тьохланкова архітектура програмного забезпечення передбачає використання додаткового елементу, а саме додаткового серверу, яких містить на якому розміщується база даних. Цей другий додатковий сервер контактує тільки з сервером на якому міститься сам веб-сайт. У свою чергу клієнт також може проводити деякі маніпуляції з другим сервером, та все ж робить він це через посередника, а саме через основний сервер. Трьох ланкова структура програмного забезпечення практично унеможливлює використання слабкого клієнта, оскільки більшість сучасних сайтів, які працюють на даній клієнт-серверній архітектурі так чи інакше зв’язую клієнта з базою даних, оскільки даний аспект впроваджується в систему саме для того, щоб клієнту було зручніше користуватись веб-сайтом та для того щоб добавити в сайт деякі нові функції.

Щодо даного програмного забезпечення, то до використання було вибрано саме тьохланкова архітектура, оскільки для реалізації інтернет магазину база даних являється невід’ємним аспектом. На рисунку 2.1 зображена архітектура інтернет магазину спортивного взуття. Також для даного сервісу буде створено

два види користувачів. Обоє ці два види будуть відноситися до так званих «Товстих клієнтів», тобто кожен з цих різновидів користувачів буде мати доступ до бази даних, але у свою чергу не в кожного клієнта будуть однакові права. І саме за поділом цих прав вони будуть називатись адміном та користувачем. Більш детально про права адміна та користувача буде описано у наступному розділі.

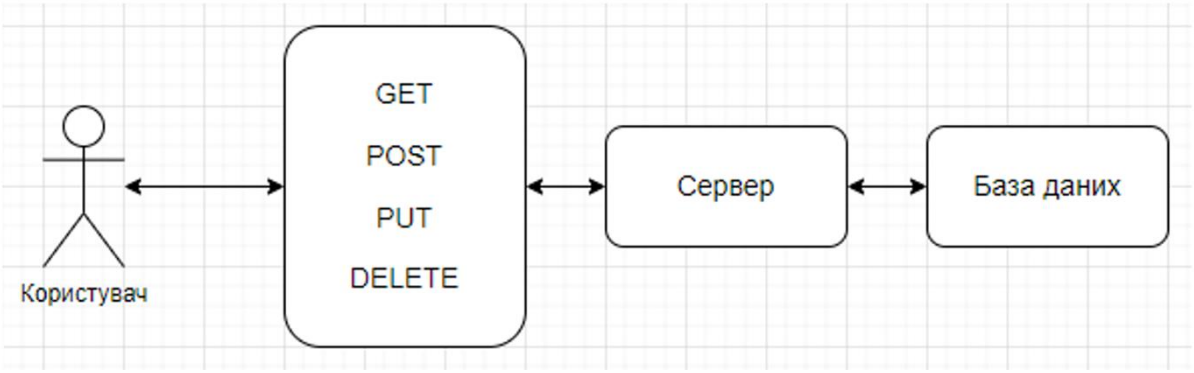


Рисунок 2.1 – Архітектура програмного забезпечення

Щодо бази даних, яка в даній діаграмі представляє з себе кінцевий елемент, то база даних слугує для веб-застосунка базою для збереження даних необхідних для роботи додатка, а саме дані про замовлення, товари категорії товарів та про адміністраторів сайту. Також дана частина клієнт серверної архітектури відповідає за всі маніпуляції, які можуть бути здійсненні над базою даними магазину.

2.1.2 Проектування структури інтернет-магазину

ЩПісля того як було вибрано архітектуру для інтернет-магазину, наступний етапом стане етап реалізації структури програмного забезпечення. Структура веб-застосунку, в даному випадку інтернет магазину, буде спрощеною, оскільки як було сказано в попередньому розділі цільові клієнти сайту – власники малого бізнесу, саме тому функціонал та структура, які супроводжуються в більш глобальних проектах не мають сенсу в даному проекті.

Саме тому основні аспекти структури даного веб-застосунку буде містити в собі головну сторінку на якій буде розміщено товари, дочірніми сторінками від головної будуть сторінки з відсортованими товарами по категоріям, які генеруються статично в залежності від кількості категорій товарів в базі даних. Другим типом сторінки буде сторінка з можливістю перегляду товару детальніше. Та останнім пунктом для користувача буде сторінка де можна оформити замовлення з товарами які були додані в корзину за допомогою попередніх сторінок. На рисунку 2.2 зображено діаграму компонентів користувацького інтерфейсу.

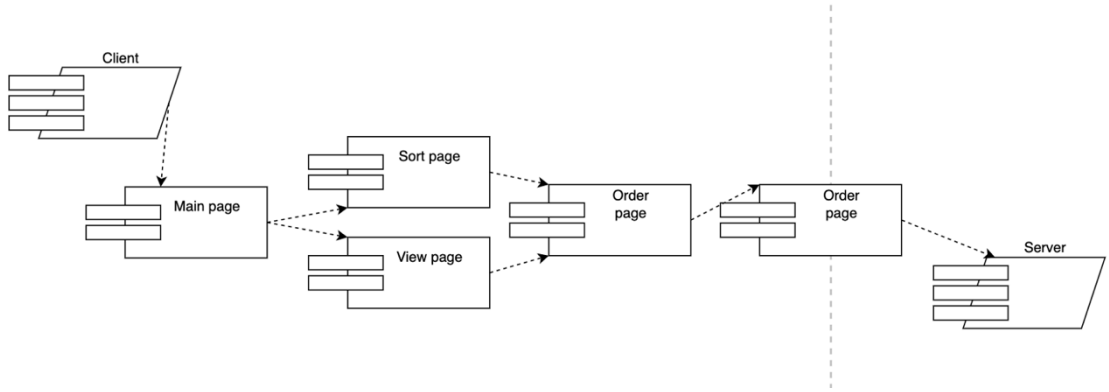


Рисунок 2.2 – Діаграма компонентів (користувач)

Тепер, щодо основного функціоналу, який стосується користувача адміна. Щодо його функціоналу, то в даному випадку всі його функції будуть більше стосуватись не до споживання контенту, а до його зміни, але все ж, всі можливості звичайного користувача у адміністратора сайту будуть. Щодо додаткових можливостей користувача, то основну увагу в даному випадку було взято на те щоб адміністратор міг вносити зміни в базу даних, маніпулювати даними. Прикладом цього може прослужити можливість адміністратора додавати нові продукти до інтернет магазину, а також редагувати категорії, редагувати інформацію додавати нову та видаляти неактуальну.

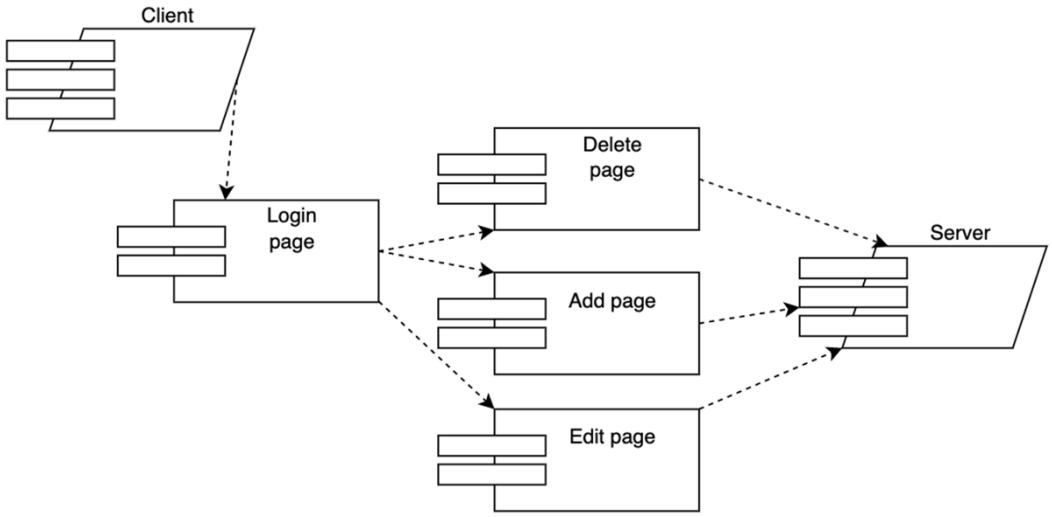


Рисунок 2.3 – Діаграма компонентів адміністратора

Такий користувач як адміністратор повинен буди захищений, так само як і доступ до сторінок редагування сайту. Для цього було використано авторизацію користувача та відповідно сторінки де адмін повинен ввести логін і пароль перш ніж приступити до роботи. Також для зручності було реалізовано доступ до сторінки авторизації з головної сторінки інтернет магазину. Більш детально про реалізацію даного програмного рішення буде описано згодом.

2.2 Розроблення моделей даних

Після всієї роботи зв’язаною моделюванням архітектури та структури інтернет-магазину, настав перший етап практичної частини розробки програного забезпечення, а саме розробка моделей даних. Першим кроком для даного етапу стане визначення інформаційних сутностей, які не обхідні для реалізації інтернет магазину. Оскільки як говорилось раніше буде використовуватись трьохланкова клієнт-серверна архітектура, яка містить в собі ще один тип серверу, який відповідає за базу даних, тому наступний крок – це пошук програмного забезпечення, яке б дозволило на локальній машині запустити SQL сервер.

2.2.1 Перелік інформаційних сутностей

Отже оскільки тема кваліфікаційної роботи є розробка інтернет-магазину спортивного взуття, то початкова база даних буде містити в собі такі основні сутності, що відповідають за наповнення товарів інтернет магазину, їхні категорії, таблиці що відповідають за замовлення та таблиці наповнення товарів. Дві останні будуть зв'язані за id. Також буде використано фреймворк Laravel, а для його нормального на повноцінного функціонування містить в собі в собі чотири базові міграції, які створюють чотири сутності в базу даних. На рисунку 2.4 зображений перелік всіх сутностей для даного проекту.

	Table	Action						Rows
☐	admin_data	☆	Browse	Structure	Search	Insert	Empty	Drop	1
☐	cats	☆	Browse	Structure	Search	Insert	Empty	Drop	3
☐	failed_jobs	☆	Browse	Structure	Search	Insert	Empty	Drop	0
☐	migrations	☆	Browse	Structure	Search	Insert	Empty	Drop	9
☐	odrders_data	☆	Browse	Structure	Search	Insert	Empty	Drop	0
☐	orders	☆	Browse	Structure	Search	Insert	Empty	Drop	0
☐	password_resets	☆	Browse	Structure	Search	Insert	Empty	Drop	0
☐	personal_access_tokens	☆	Browse	Structure	Search	Insert	Empty	Drop	0
☐	products	☆	Browse	Structure	Search	Insert	Empty	Drop	2
☐	users	☆	Browse	Structure	Search	Insert	Empty	Drop	0
	10 tables	Sum							15

Рисунок 2.4 – Таблиці бази даних

Як можна побачити окрім таблиць з які будуть служити, саме для роботи інтеренет-магазину, Laravel після здійснення міграції бази даних створив додаткові таблички такі як: personal_access_tokens, migratios, failed_jobs, users. Данні сутності служать для більш зручної робот з базою даних та для дотримання більшої структуризації проекту. До прикладу можна навести табличку migrations

2.3 Проектування структурних елементів інтернет-магазину

В даному проекті використовувався процедурний підхід розробки програмного забезпечення. Процедурне програмування як методика розробки програмного забезпечення розділяє програму на дані і процедури, які обробляють ці дані.

Переваги, якими володіє процедурне програмування в порівнянні з неструктурним програмуванням. Процедурное програмування задає загальну структуру програми:

– дані;
– процедури.

Використання процедур допомагає розробляти програму, призначену для вирішення конкретного завдання. Замість написання одного великого блоку обробки даних в ході розробки програма весь час розбивається на процедури і більш дрібні підпрограми. Крім того, процедури можна використовувати повторно. Можна навіть створювати бібліотеки багаторазово використовуваних процедур. В даному проекті буде виеористовуватись шаблон model-view-controller, де основний елемент буде використовуватись в так званих контроллерах програми. Види використовуються для того щоб реалізувати інтерфейс користувача, а зв’язувати їх будуть роути програми. Щодо моделей то в laravel вони використовуються для зручного використання бази даних, а також дозволяє безпомиклоко використовувати модель даних [16s].

2.3.1 Основні модулі для веб-застосунку та їх функціонал

В даному підрозділі будуть описуватися всі ключові структурні елементи, які використовуються в веб-застосунку. Перший і одним з найважливіших модулів в проекті – це модуль підключення проекту до бази даних. Це робиться за допомогою laravel де треба підлаштувати данні підключення до БД. Для цього в файлі .env потрібно підправити дані так щоб вони відповідали локальному

хостингу. В лістингу 2.1 зображено конфігурація підключення до локального хостингу.

Лістинг 2.1 – Конфігурація файлу .env

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=8889
DB_DATABASE=diplom_bootshop
DB_USERNAME=root
DB_PASSWORD=root
```

Далі потрібно змінити інформацію в іншому конфігураційному файлі, даний файл знаходиться в папці config і називається database.php і відповідає за те щоб веб-сайт працював саме з mysql.

```
'database' => env('DB_DATABASE', 'diplom_bootshop'),
'username' => env('DB_USERNAME', 'root'),
'password' => env('DB_PASSWORD', 'root'),
```

Після того як було пророблено роботу над підключенням до бази даних, можна переходити до створення моделей даних. Як було сказано в попередньому розділі будуть використовувати такі данні як дані про клієнта, замовлення, категорії та продукцію. До прикладу можна навести модель категорій. Дана модель буде включати в себе такі дані як самі категорії товарів магазину, id та їхня назва на англійській мові для того щоб дані категорії було зручно використовувати в контролерах при здійсненні маніпуляцій над ними.

Після того як було створено модель даних та підключення до бази даних в терміналі потрібно звернутися до PHP та artisan та прописати команду подану нижче:

```
php artisan migrate
```

Після впровадження даної команди Laravel створив файл міграції таблички та заповним додатковими даними таблицю migrations та створив саму табличку

cats. Далі після цього, фреймворком Laravel було автоматично додано колонку в файл, який відповідає за міграцію даної таблички, а саме колонку з типом даних timestamps(). Дана колонка відповідає за те щоб записувати дані про час додавання та редагування змісту кожної сутності. Дане поле є обов'язковим та при видалені timestamps() сутність не можливо буде використовувати для редагування або ж видалення даних.

Після того як було створено модель даних та файл міграції, можна приступати до наповнення категорій товару та саме головне до створення вигляду хедера товару для того щоб дані категорії можна було б відобразити для користувача. Для цього буде використовуватись шаблонізатор – blade та файли з розширенням php. Даний шаблонізатор був вибраний не спроста, оскільки саме він допомагає інтегрувати вбудовані функції Laravel в сторінки інтерфейсу користувача. Це потрібно перш за все для динамічної генерації даних на сторінку, оскільки дана методика дає можливість значним чином скоротити кількість коду та зробити його більш читабельним та зрозумілішим для інших розробників. В лістингу 2.2 зображена динамічна генерація категорій товару для магазину у файлі header.blade.php.

Лістинг 2.2 – Частина сторінки header.blade.php

```
<!doctype html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Shop</title>
    <link rel="stylesheet" href="{{ asset('/css/app.css')}}">
    <link rel="stylesheet" href="/css/sim-slider-styles.css">
    <link rel="stylesheet" href="/css/fb.css">
</head>
<body>
<nav>
    <ul>
        <div class="seper"></div>
        @foreach($cats as $item)
        <li><a href="{{route('cats', ['slug'=>$item-
>id])}}">{{ $item->ua_name }}</a></li><div
class="seper"></div><li>
        @endforeach
```

```
            <li><a href="/game">Знижка</a></li><div
class="seper"></div>
            <li>
```

Як можна побачити в лістингу було використано Laravel функцію @foreach, яка дозволяє реалізувати цикл перебирання кожного елементу масиву що був переданий на сторінку. За допомогою даного циклу в HTML тег було додано назву категорії та в посилання до тегу було додано посилання яке в майбутньому буде вести на сторінку де продукти будуть відноситись до конкретної категорії товару.

Після того як шаблон сторінки був створений, лишилось передати данні з бази даних на дану сторінку, а саме змінну $cats. Даний етап реалізується за допомогою контролера створеного спеціально для того щоб реалізовувати backend, що буде відповідати за всі маніпуляції зв’язані з категоріями товарів. Даний контролер буде відповідати та передавати дані про категорії на всі сторінки які цього потребують. В лістингу 2.3 наведена реалізація функцій деякий функцій контролера, які в даному випадку відповідають за виведення категорій товару на головну сторінку та передачу конкретної категорії товару на сторінку де буде виведене взуття, яке належить до даної категорії.

Лістинг 2.3 – Контролер категорій товарів

```
class CatsController extends Controller
{
  public function allDataCats()
  {
     $cats = Cats::all();
     $products = Products::all();
     return view('index', compact('cats', 'products'));
  }
  public function oneProduct($slug)
  {
     $cats = Cats::all();
     $product = Products::where('id', $slug)->firstOrFail();
     return view('oneProduct', compact('cats', 'product'));
  }
}
```

Як можна побачити з даного лістингу тут використовується так звана заглушка $slug, яка і містить в собі вибрану категорію.

Тепер після того як в програмі було створена модель даних, яка бере дані з бази даних та яку використовує контролер, сам контролер, які ці дані обробляє та сторінка яка ці дані показує користувачу, залишився один крок, а саме зв’язати контролер та сторінку в якій міститься header.blade.php. Іншими словами потрібно передати дані з бекенду в фронтент. Дане рішення буде реалізавоне за допомогою роута. Всі роути, що всідповідають за передачу даних або ж за перехід з одної сторінки в другу, знаходяться в документі web.php, який знаходить в папці ресурсів проекту. До прикладу можна навести реалізацію роута, шо відповідать за головну сторінку.

```
    Route::get('/','App\Http\Controllers\CatsController@allDataCa
ts')->name('index');
```

Структура роута дуже проста, адже містить в собі шлях до файлу посилання на функцію в контролері та власне ім’я роута. У свою чергу функція та посилання обгорнуті в метод, в даному випадку використовується метод – get. Дана структура несе за собою також дуже непоганий функціонал оскільки, ім’я роута в деяких випадках дозволяє в бекенді зручно реалізовувати редіректи та назва посилання дозволяє переходити на сторінки з фронтенту.

2.3.2 Робота з cookie

В даному підрозділі буде розроблена основна робота з cookie для проекту. Cookie – це файли, які зберігаються на комп’ютері користувача та слугують для того щоб записувати дії користувача на тому чи іншому веб-сайті. В контексті даного проекту цей функціонал буде використовуватись для того щоб записувати дані добавлених в корзину товарів для користувача. Для цього буде використовуватись спеціально створений контролер та з багатьма функціями, які будуть використовуватись для видалення та додавання продуктів. Також зокрема

продукції інтернет-магазину, cookie будуть використовуватись для того, щоб обрахувати загальну знижку до замовлення [17]. В лістингу 2.4 наведена реалізація однієї з функцій, яка відповідає за додавання знижки в замовлення користувача.

Лістинг 2.4 – Робота з cookie

```
public function AddSale(Request $request, Response $response)
{
    $sale = $request->input('sale');

    unset($_COOKIE['sale']);
    setcookie('sale', json_encode($sale), time()+3600);
    return redirect('/');
}
```

Як можена побачити у функцію AddSale() входять дві змінні, власне $request містить в собі данні, які користувач отримує з фронтенту, а саме дані про знижку до замовлення php функція uset() відповідає за те щоб всі попередні дані які містили інформацію про знижку знищились, а далі створюються нові cookie, які вже містять оновленні дані. За таким самим принципом працює і додавання продукції в корзину, за єдиною відмінністю, що робота з таким типом даних набагато складніша та потребує більш продуманих алгоритмів, оскільки працює вже з двома моделями даних, а саме категорії товарів та дані про самі товари.

2.3.3 Робота з сесією

Далі після того як було описано та наведено приклади з реалізацією ключових аспектів створення бекенду клієнтської частини інтернет магазину за допомогою фреймворку Laravel, можна починати над роботою сторінки адміністратора. Звісно дана частина веб-сайту з точки зору роботою з моделями даних та реалізацію до серверу буде набагато складнішою, але за своїми принципами не буде відрізнятись від проектування описаного вище, за

виключенням одного пункту. А саме робота з сесіями для безпечного надання сторінок адміністратора саме адміністраторам веб-сайту [18]. Іншими словами, дана технологія не буде дозволяти користувачам за допомогою посилань відкривати сторінки адміністратора для маніпуляції над даними сайту.

Робота з сесіями та робота з cookie дуже схожа за принципами, і з технічної точки зору реалізувати перевірку активованої сесії адміністратора для інтернет-магазину можна і за допомогою cookie, але оскільки як було сказано раніше ці дані зберігаються на комп'ютері користувача, то було вирішити використовувати саме сесію, адже одна з її переваг та відмінностей це те що на комп'ютері користувача зберігаються тільки ідентифікатори для файлів, які вже у свою чергу розміщуються на сервері [19]. Саме тому можна зробити висновки, що дана технологія є менш ефективною та швидкою, але за те набагато більш безпечніша. В лістингу 2.5 наведений приклад створення сесії після авторизації адміністратора сайту.

Лістинг 2.5 – Створення сесії адміністратора

```
public function Login(Request $request)
  {
    $login = $request->input('login');
    $password = $request->input('password');

    $name = AdminData::where(["log" => $login, "pas" =>
$password])->first();

    if(empty($name)){
      // $error = 'Неправильно введені дані !!!';
      return redirect()->route('admin.login');
    }else {
      Session::put('variableName', 'Active');
      $wert = Session::get('variableName');
      return redirect()->route('admin.index');
    }
  }
  public function DropSession(Request $request)
  {
    Session::forget('variableName');
    return redirect()->route('admin.login');
  }
public function LoginPage()
  {
```

```
        return view('admin/login');
    }
```

Як можна побачити в лістингу в функції Login() перевіряється пароль та логін адміністратора сайту. Далі після цього створюється сесія та її значення становиться Active. З такими даними. Для користувача будуть доступні всі сторінки сайту, включаючи сторінки адміністратора. Таким чином буде реалізована перевірка значення сесії та її видалення з часом або ж коли користувач вийде з панелі адміністратора.

2.4 Проектування інтерфейсу користувача

Даний розділ буде присв’ячений інтерфейсу користувача. Це є один із найважливіших етапів розробки програмного забезпечення, оскільки користувач це єдиний елемент на який користувач буде звертати найбільшу увагу. Інтерфейс користувача – невідємний та найважливіший елемент кожного веб-сайту, адже які б нові та продвинуті технології не використовувались в проекті слід пам’ятати, що для простого користувача важливий тільки фронтент [20]. У межаш даного проекту дизайш та інтерфейс користувача буде втілюватись да допомогою таких засобів як HTML5, CSS3 та JavaScript, а саме такі елементи як контейнери, слайдери, меню, шрифти та медіафайли.

2.4.1 Обґрунтування вибору кольорової схеми

Щодо кольорів які використовуються в дані роботі, було вирішено використовувати біло-чорну кольорову гаму. Також варто зазначити, що всі функціональні елементи сайту, які виконують ту чи іншу функцію – робота з базою даних, або ж навігація по сайту, наділені властивістю при наведені світитися червоним кольром. Значну увагу треба також виділити вибору заднього фону, на рисунку 2.5 навдений приклад реалізації кольорової схеми на базі основної сторінки сайту.

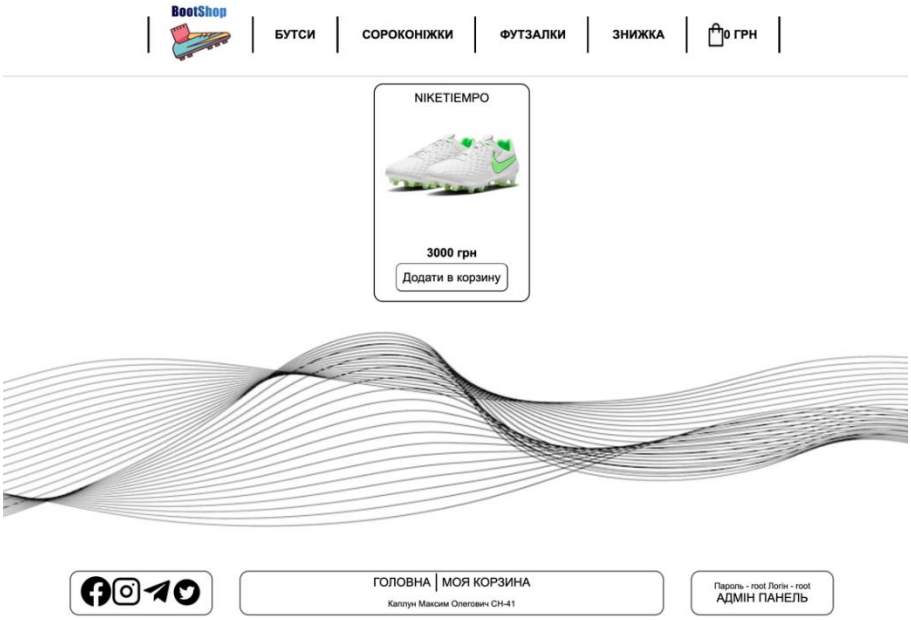


Рисунок 2.5 – Кольорова схема сайту

Як можна побачити основні елементи користувацького інтерфейсу чорного кольору, також при взаємодії користувача з даними елементами, відображаються анімації зв’язані з зміною кольору тексту та заднього фону [21].

2.4.2 Обґрунтування структури шаблонів інтерфейсу

Щодо структури шаблонів інтерфейсу, та в даній роботі основний акцент було взято, на структуру контейнерів, в яких розміщені товари магазину. Структура дизайну контейнера для розміщення товарів в вигляді CSS коду наведено в лістингу 2.6. Стосовно колірної реалізації сайту, єдиний елемент у веб-застосунку, який порушує баланс тонів – логотип, що був розроблений за допомогою програми AdobePhotoshop, для підкреслення оригінальності дизайну та набуття бренду.

Лістинг 2.6 – Створення сесії адміністратора

```
.card {
  margin-bottom: 15px;
  width: 20%;
  padding: 10px;
  border: 2px solid black;
```

```
  border-radius: 15px;
  text-align: center;
  background-color: #fff;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
}
.cart {
  margin-bottom: 10px;
  display: flex;
  justify-content: flex-start;
  align-items: center;
}
.cart-descr {
  margin: 0 10px;
  width: 80%;
}
```

Саме ці класи використовуються в контейнері динамічної генерації товарів. Дані класи та, а саме атрибути в загальному використовувалися при всій розробці дизайну інтерфейсу в веб-застосунку [22]. Більше класів та структур шаблонів інтерфейсу наведені на в розділі додатки.

Стосовно іншої візуальної частини, яка використовувалася про розробці даного програмного ПЗ. Анімації JS – а саме модуль sim-slider.js. Даний модуль використовується для реалізації слайдера фотографій при перегляді товару на сайті. Для цього використаємо скрипт наведений в лістингу 2.7.

Лістинг 2.7 – Основні функції документу sim-slider.js

```
if(that.options.arrows) {  // ініціалізація стрілок
          that.leftArrow.addEventListener('click',  function()
{
               let fnTime = getTime();
               if(fnTime - bgTime > 1000) {
                    bgTime = fnTime; that.elemPrev()
               }
          }, false);
          that.rightArrow.addEventListener('click', function()
{
               let fnTime = getTime();
               if(fnTime - bgTime > 1000) {
                    bgTime = fnTime; that.elemNext()
               }
          }, false) }
          if(!that.options.loop) {
```

```
            that.leftArrow.style.display = 'none';  // виключення
л.с
            that.options.auto   =   false;   //   виключення   авто
прокрутки при наведенні мишкою
        }
```

Як можна побачити в лістингу наведеному вище після ініціалізації елементів контейнера, а саме такі змінні як this.sldrList, this.sldrElements, this.sldrElemFirst, this.leftArrow, this.rightArrow, this.indicatorDots, які відповідають за такі об’єкти як права та ліва стрілка, контейнер слайдера, перший елемент, всі поточні елементи та кнопки сторінок, ми можемо приступити до виконання основних функцій. Перше що потрібно зробити виконати зациклювання функції за допомогою CSS. На рисунку 2.6 зображена реалізація вищенаведених лістингів.

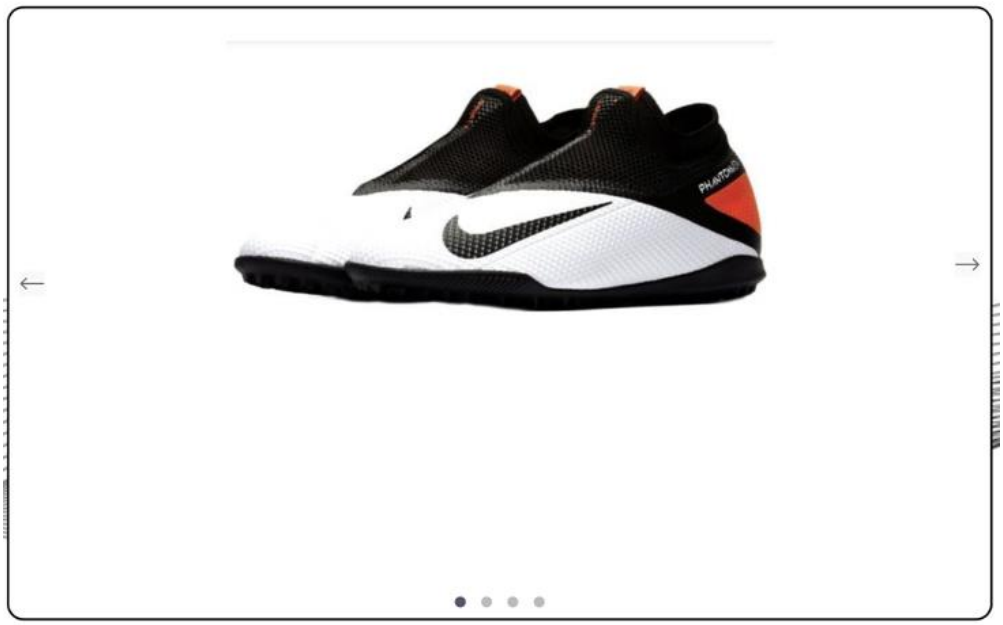


Рисунок 2.6 – Реалізація слайдера

На рисунку видно, що дана анімація включає в себе 4 сторінка, три з яких є забраженням товару, а четверта - відео товару, яке інтегрується за допомогою методу ifreme, що дозволяє в базі даних збурігати лише інтегровані силки. Повний лістинг анімації наведений в додатку В.

2.5 Створення інтегрованих анімацій

Наступним важливим кроком в розробці інтернет магазину – це розробка інтегрованих анімацій. Для реалізації даного завдання було використано в HTML тег canvas, який слугує спеціально для того щоб розміщувати на веб-сайті графіку. А у свою чергу керувати даної анімацією буде мова програмування JavaScript, оскільки CCS не володіє такими широкими можливостями [23]. До прикладу можна навести рух об’єкта по шляху або ж підтримка тієї самої canvas анімації.

Ідея інтегрованої анімації полягає в тому, що в контексті інтернет-магазину вона буде відігравати не тільки візуальну роль, а до неї буде прикріплена функція, яка буде взаємодіяти з користувачем та давати йому, якісь додаткові можливості [24].

Реалізація цієї ідеї така що створено невелику міні-гру, суть якої перелітати перешкоди головним героєм спортивним кросівком. На самому початку було розроблено всі необхідні медіа та аудіо файли для цієї анімації. Більшість картонок було намальовано за допомогою програмного забезпечення Abode Photoshop. Для того щоб розмістити міні-гру в інтернет-магазині було створено окремий вид, контролер та декілька роутів які виконують функції контролера. Тег для тегу canvas було підібрано розміри та розміщено на сторінку. Далі логіка самої анімації, в лістингу 2.8 наведений програмний, який відповідає за генерацію перешкод.

Лістинг 2.8 – Генерація перешкод для гри

```
    for (var i = 0; i < pipes.length; i++) {
        context.drawImage(pipeUp, pipes[i].x, pipes[i].y);
        context.drawImage(sale_persent, pipes[i].x, pipes[i].y +
270, 50, 50);
        context.drawImage(pipeDown, pipes[i].x, pipes[i].y +
          pipeUp.height + distance);
        pipes[i].x--;
        if (pipes[i].x == 125) {
          pipes.push({
            x : canvas.width,
```

```
            y : Math.floor(Math.random() * pipeUp.height) -
pipeUp.height
          });
        }
```

Як можна побачити в лістингу, створено цикл, який випадково генерує перешкоди по їхній висоті. Також в даній анімації було прописано такі умови як виграш, коли змінна знижка рівна – 10% та умова програшу, коли персонаж гри (кросівок) вдаряється в перешкоду. Також важливий етап є передача числа знижки в тег HTML, який буде передавати дане число формою в бекенд. В лістингу 2.9 наведена реалізація даного етапу.

Лістинг 2.9 – Передача результату гри на сторінку

```
if (10 == pipes[i].x){
    points++;
    pointsAudio.play();
    get_sale.innerHTML = 'Отримати: ' + points/2 + '%'
}
```

Після того як саму анімацію та її результат було передано в форму, а з форми дану інформації необхідно перенести в бекенд, де в контролері знижка буде перенесена в cookie, та після того як користувач оформить замовлення на сайті, загальна сума замовлення буде обрахована з урахуванням знижки. Далі після додачі замовлення в базу даних, всі cookies, які містять інформацію про знижку на замовлення автоматично очищаються.

2.6 Наповнення інтернет-магазину

Після етапу додавання всіх анімацій, та проороблення всієї роботи над фонтентом та бекендом користувацької та адмін частини веб, можна починати роботу над наповненням інтернет-магазину. Перш за все необхідно наповнити інтернет магазин всіма необхідними категоріями продукції. Для цього після здійснення міграції в табличку cats було додано три основні категорії футбольного взуття. Оскільки як було сказано в попередніх розділах, всі розділи

та пункти меню хедера генеруються автоматично, то після роботи над бекендом програми, лишається просто отримати результат зображений на рисунку 2.7.

id	name	ua_name	created_at	updated_at
1	buts	Бутси	2022-05-10 12:44:09	2022-05-10 12:44:09
2	40f	Сороконіжки	2022-05-10 12:44:09	2022-05-10 12:44:09
3	footgum	Футзалки	2022-05-10 12:45:19	2022-05-10 12:45:19

Рисунок 2.7 – Наповнення таблиці з категоріями

Як можна побачити на рисунку дана табличка має дуже просту структуру, таку як: дата створення, дата додавання, назва на українській та на англійській мові (щоб не було помилок з кодуванням в бекенді). Після проробленої роботи з категоріями, йде етап наповнення інтернет-магазину продукцією. На даному етапі потрібно вирішити де будуть зберігатись медіа файли до проекту та за допомогою чого вони будуть підключатись. Дану проблему було вирішено за допомогою посилань. Адже за допомогою даного методу файли можна зберігати як локально так і віддалено. В фреймворку Laravel всі медіа файли необхідні для функціонування проекту розміщуються в файлі public. Відповідно в базу даних буде записано тільки посилання на файли в незалежності від того чи є ці посилання локальні чи глобальні. Для того щоб інтегрувати відео файли було використано тег iframe. Для даного тегу було вирішено використати відео з веб-сервісу YouTube. Для інтеграцію відео потрібно зайти на веб-сайт, знайти відео-огляд потрібного продукту та вибрати пункт поділитися, а потім вставити, з спливаючого вікна необхідно скопіювати посилання та додати дане значення в таблицю в базі даних. Також, оскільки в мережі важко знайти відео огляд на деякі одиниці товару, а знімати відео огляд продавцю самому не є зовсім раціонально, тому даний пункт для кожного з товарів було вирішено зробити опціональним. Після проробленої роботи було отримано результат наведений на рисунку 2.8.

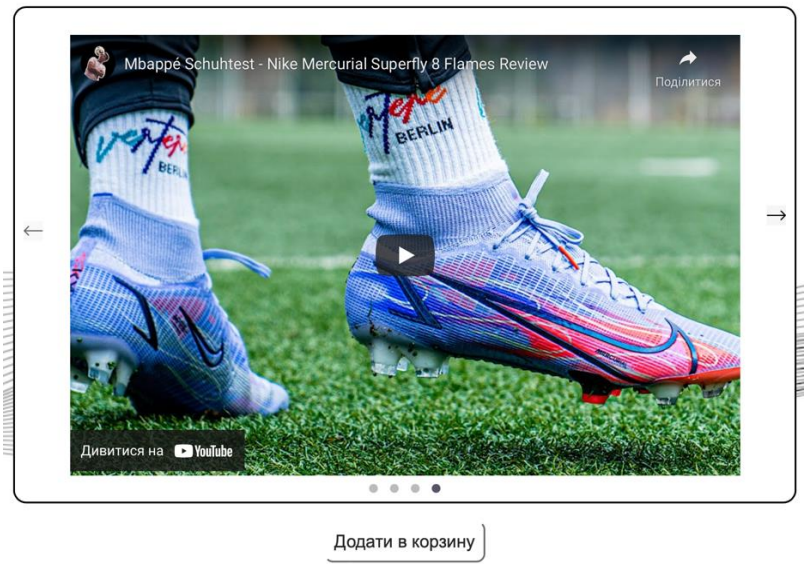


Рисунок 2.8 – Відео-огляд товарної одиниці

Як можна побачити на рисунку тег iframe з посиланням на відео з платформи YouTube підтримує весь функціонал даної платформи. В стилях проекту було визначено лише максимально висоту та ширину даного тега.

2.7 Розміщення інтернет магазину на хостингу

Після завершення тестування та експлуатації програмного забезпечення перейдемо до фінального етапу розробки інтернет магазину – а саме завантаження готового сайту на хостинг. Для цього скористаємося сервісом ho.ua. Єдине питання щодо завантаження веб-сайту – це перенесення бази даних сайту. Для цього на локальному хостингу було використано сторінкою phpMyAdmin та перейшовши на потрібну нам базу даних скористаємося вкладкою – export (Див. рисунок 2.9).

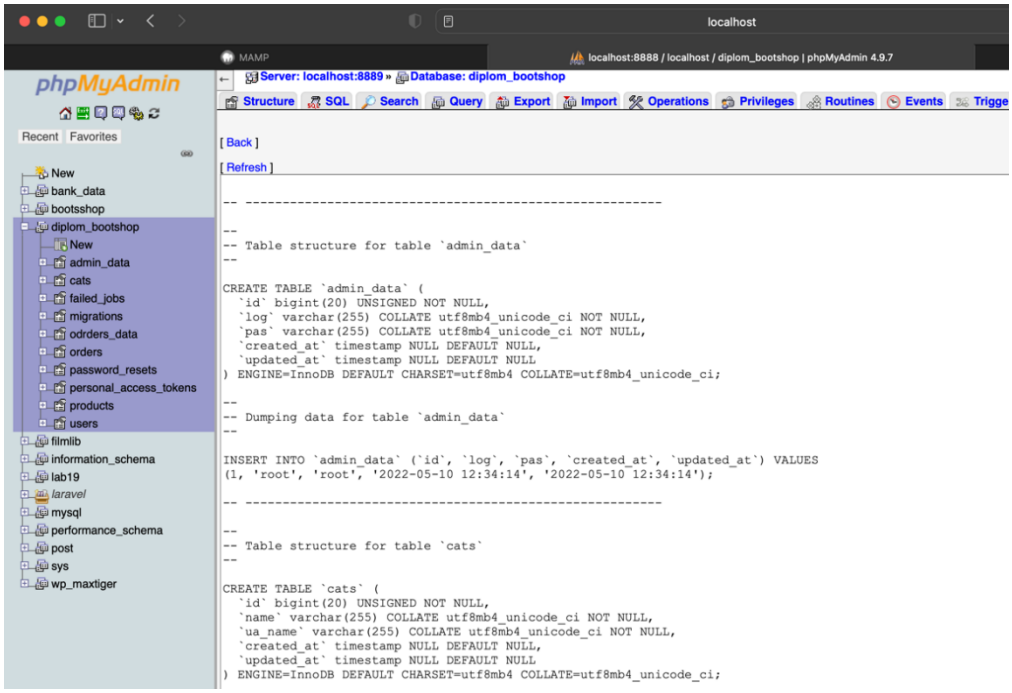


Рисунок 2.9 – Експорт бази даних

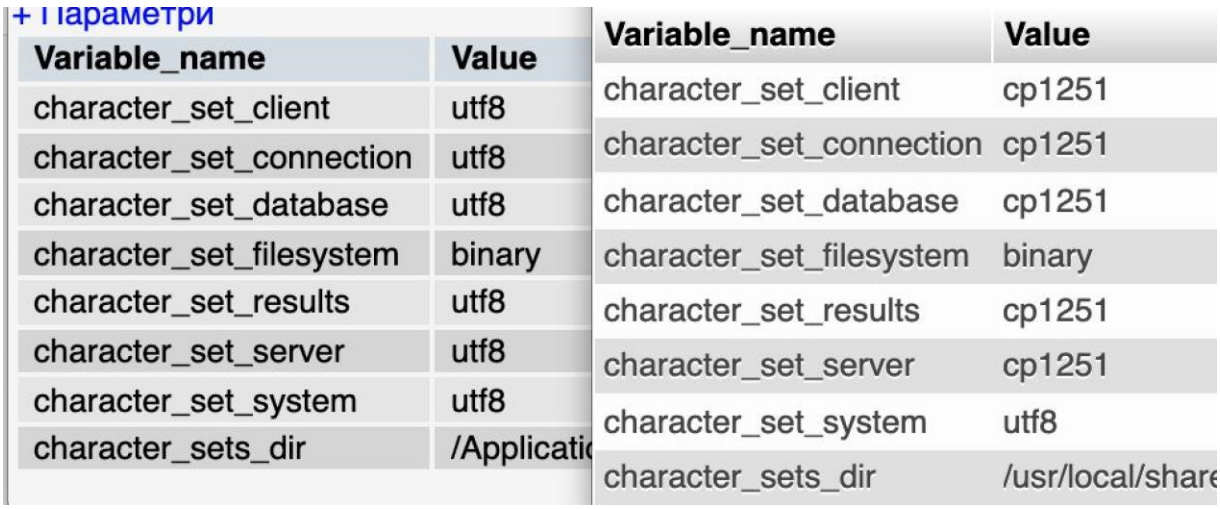
Згенеровані SQL команди потрібно скопіювати та скористатися сторінкою phpMyAdmin вже на вибраному хостингу, перед тим створивши базу даних з кодуванням UTF-8, та вставити дані команди у вкладку SQL [25]. Після проведення даних маніпуляцій база даних імпортується зі всіма необхідними колонками та табличками. Після цього було проведено деяку роботу з модулем підключення до бази даних описаним раніше. В даному документі потрібно поміняти значення назви серверу, логіну на паролю користувача.

```
DB_HOST=host=db1.ho.ua
DB_PORT=8889
DB_DATABASE=diplom_bootshop
```

Далі було проведено тестування програмного забезпечення на хостингу. Після проведеного тестування було виявлено деякі помилки, саме помилки в кодуванні на сервері. Для вирішення цього питання було прийняті такі міри. Написано в технічну підтримку хостингу де це питання до кінця вирішити не змогли, адже постійно перенаправляли на сторінку питань та відповідей роботи з хостингом. Стосовно виправлення даної проблеми SQL запитами також не

вирішило проблеми, саме тому було зроблено припущення, що дану проблему може вирішити тільки компанія. Тому було вирішено поміняти кодування програмно. У цьому помогло скористатися таки функціями PHP як mb_detect_encoding() та mb_convert_encoding() для визначення та задавання кодування відповідно.

Виглядають дані помилки наступним чином – перша частина фотографії – серверне кодування на локальному хостингу (UTF-8), друга кодування на хостингу(sp1251) наведено на рисунок 2.10.



+ Параметри

Variable_name	Value
character_set_client	utf8
character_set_connection	utf8
character_set_database	utf8
character_set_filesystem	binary
character_set_results	utf8
character_set_server	utf8
character_set_system	utf8
character_sets_dir	/Applicati

Variable_name	Value
character_set_client	cp1251
character_set_connection	cp1251
character_set_database	cp1251
character_set_filesystem	binary
character_set_results	cp1251
character_set_server	cp1251
character_set_system	utf8
character_sets_dir	/usr/local/share

Рисунок 2.10 – Відмінність кодування хостингу MAMP від ho.ua

В кінцевому результаті після проведення маніпуляцій над деякими запитами з було отримано повністю готовий до роботи сайт, завантажений на хостинг, але звісно потрібно розуміти що такий вихід з ситуації не є повністю коректний та раціональний, саме тому в майбутньому хостинг потрібно обов’язково змінити на інший.

2.8 Застосування програмного забезпечення

Після етапу розробки програмного забезпечення та завантаження веб-сайту на хостинг, настає етап тестування. Для цього перед початком тестування було додано на сайт декілька товарів, для того щоб подивитись як буде виглядати

наповнена веб-сторінка. Далі можна починати можна перевіряти веб-сайт на працездатність, а саме перевіряти строрінку USER та ADMIN на коректність роботи. На рисунку 2.11 можемо побачити вигляд головної сторінки сайту, його наповнення.

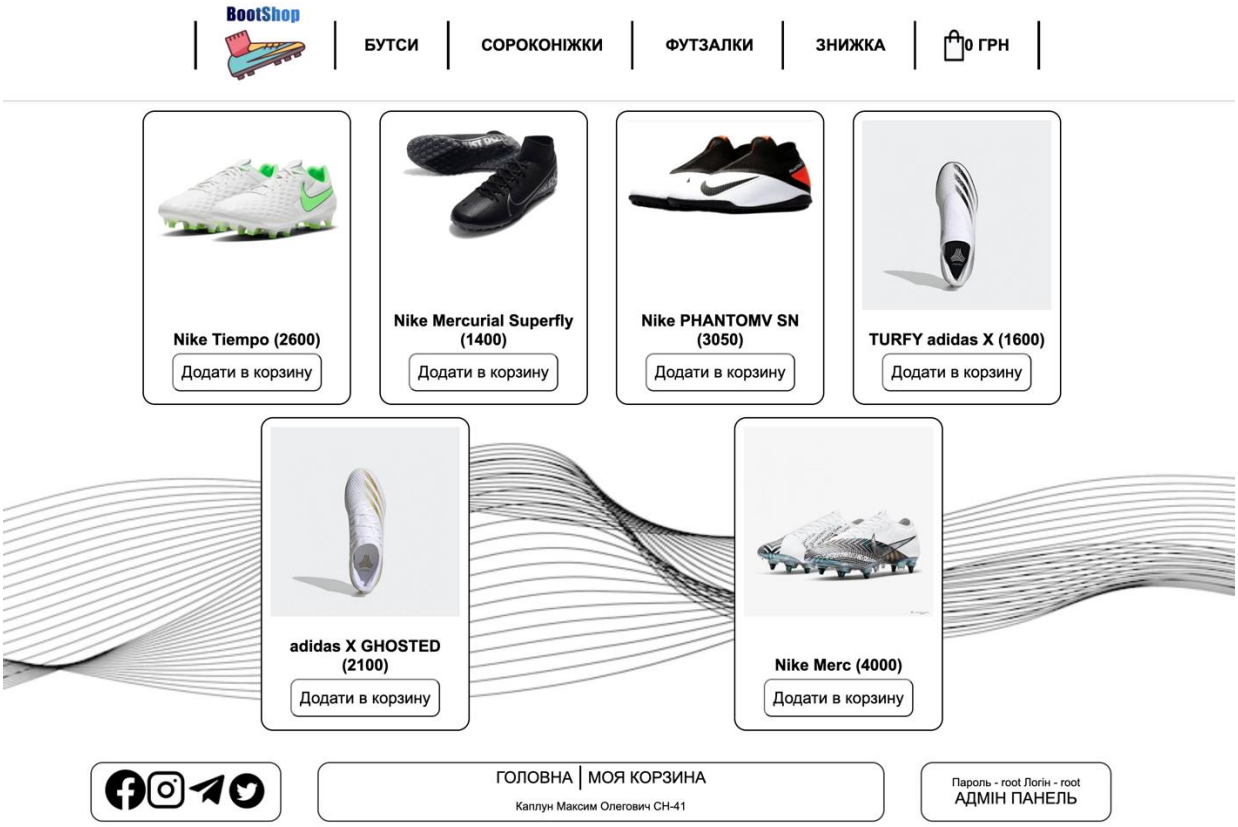


Рисунок 2.11 – Головна сторінка

Як ми можемо бачити на рисунку зображені всі товари, які додаються на сторінку динамічно, та додаються на головну по 10 відсотків заповнення сторінки. Також веб-застосунок дозволяє користувачеві переглядати товари відсортовані по категоріям та переглядати кожен товар окремо. В додачу, звертаючи увагу на футер можна побачити, що він прикріплений саме до низу сторінки. Дане рішення було вибрано не спроста адже в даному випадку футер виступає допомоміжною панеллю, яка навідмінно від хедера буде з користувачем завжди та може містити в собі зручну інформацію та посилання на сторінки, які допомагаю зручніше ззаємодіяти з веб-сайотм.

Неменш важлива частина це сторінка з відображенням безпосередньо одного товару. Дана сторінка містить в собі фотографії одиниці товару та також вона застововує модуль sim-slider.js описаний вище. На рисунку 2.12 зображено реалізацію перегляду товарної пропозиції на сайті.

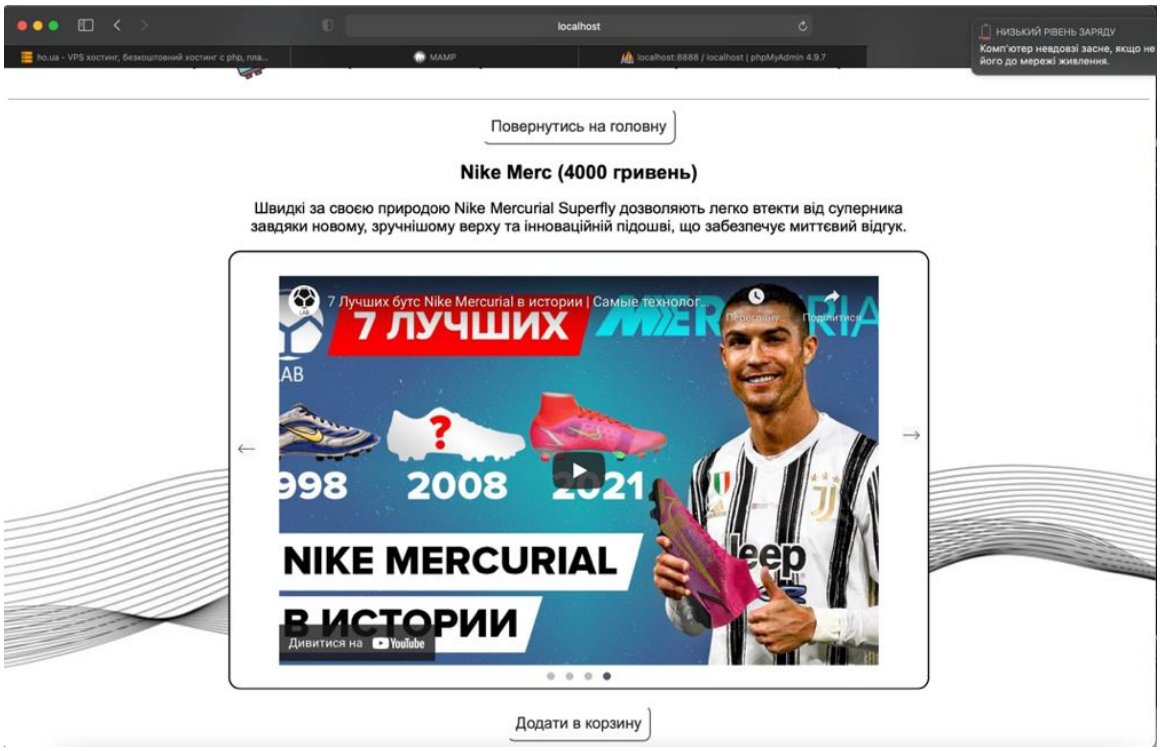


Рисунок 2.12 – Перегляд товару

Як можна побачити сторінка перегляду товару має такі можливості як додавання товару в корзину, перегортання фотографій та повернення на головну сторінку.

Наступна сторінка, якою користується User – це сторінка можливості отримання знижки. Дана сторінка знаходиться в тестовому режиму, оскільки через редизайн, або через ребрендинг сайту її вигляд може змінитися, адже це єдина частина інтернет-магазину, яка працює динамічно. Але з технічної точки зору дана анімація повністю довершена за єдиним виключенням, що не адаптована під мобільні пристрої, адже для цього потрібно стоворювати окремі кнопки, а анімація потребує наявності пробілу в пристрої. На риснку 2.13 наведена реалізація сторінки з веб-грою.

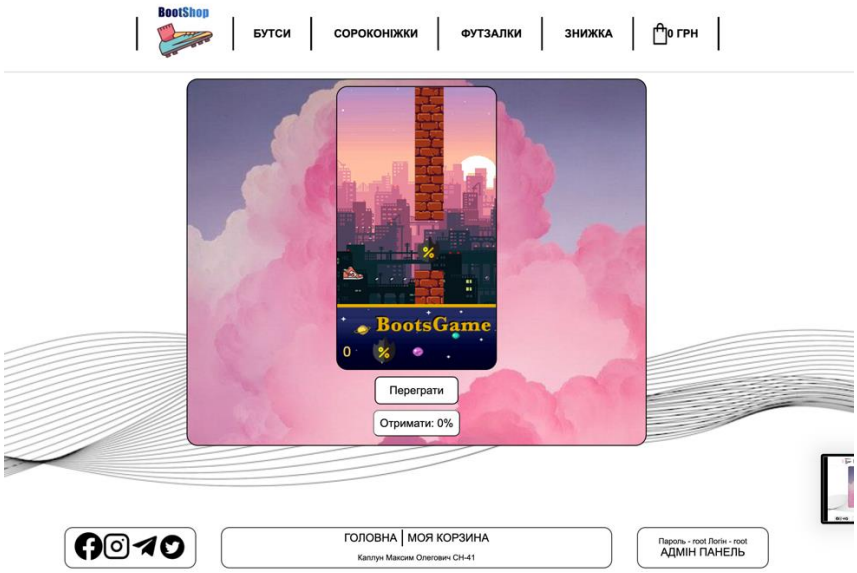


Рисунок 2.13 – Веб-гра з можливістю отримати знижку

Стосовно елементів, які в майбутньому можуть потребувати редагування, то це можу бути задній фон для контейнера в якому розміщається гра та самі елементи гри такі як головний герой перешкоди, вони є гнучні та можуть легко замінитись.

Заключним етапом використання, цього веб-застосунку для клієнта є додавання одиниць продукції в корзину. (Див. рисунок 2.14)

Рисунок 2.14 – Корзина товарів

Дана сторінка володіє можливістю віднімання помилково доданого товару та можливістю заповнення інформації про клієнта, для подальшого надходження цієї інформації в базу даних про замовлення в магазині.

Далі буде розглятута адмін частина сайту яка включає в себе наступний функціонал: редагування товару в магазині, додавання нової категорії додавання нового товару, редагування замовлення клієнта, тобто всі необхідні функції, якими повинна володіти адміністративна панель інтернет-магазину. На рисунку 2.15 наведена реалізація даної частини веб-сайту [26].

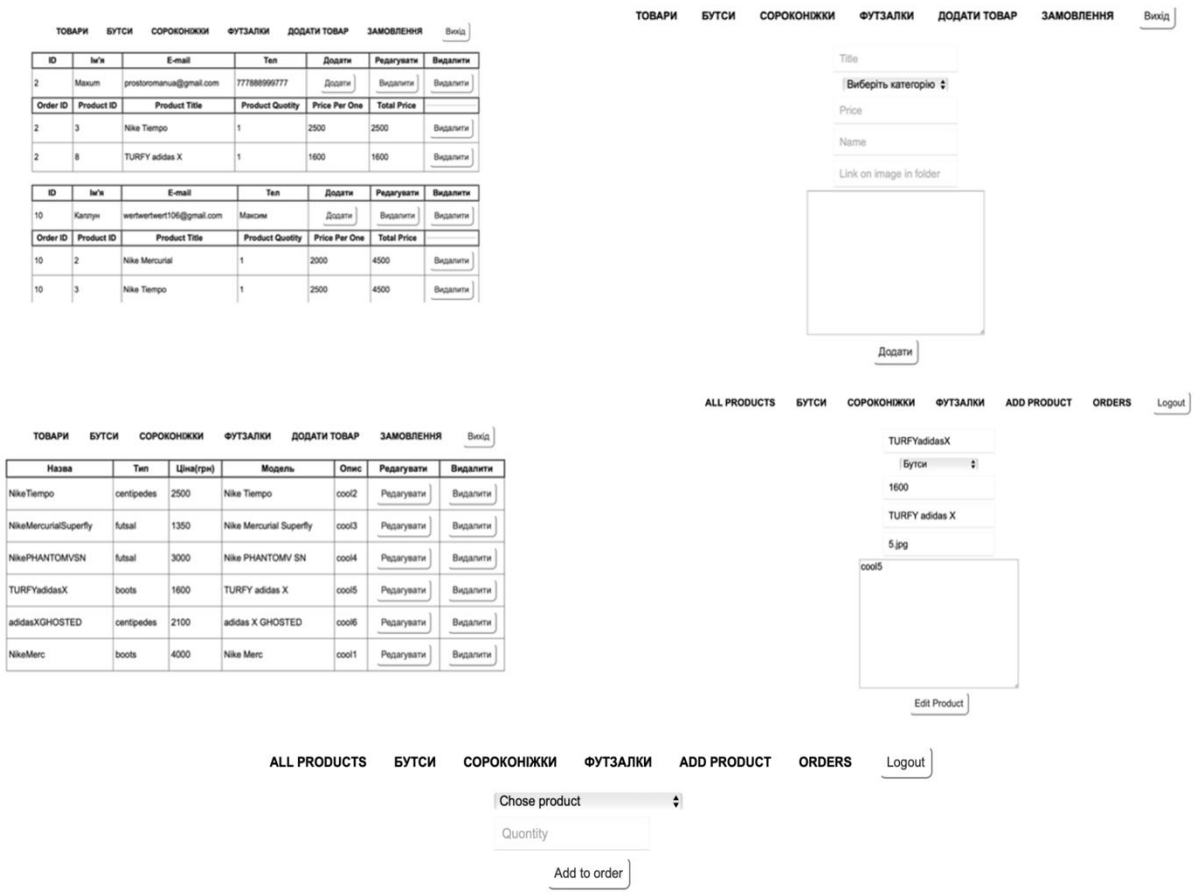


Рисунок 2.15 – Можливості адмін сторінки

Для виконання входу в адмін сторінку сайту, буде використовуватись форма входу з паролем та логіном. Всі паролі та логіни адміністраторів сайту записані в окремій табличці в бази даних. На рисунку 2.16 зображена процедура входу в адмін панель сайту.

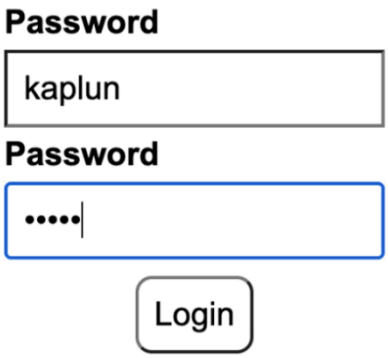


Рисунок 2.16 – Вхід в адмін панель

Для зручності, оскільки веб-сайт зараз перебуває в бета тесті, було вирішено перемістити посилання на форму входу в адмін панель в футері на користувацькій частині. Але звісно з часом дане посилання буде зміщене та вхід адміністратора буде здійснюватись за допомогою окремого посилання.

2.9 Висновок до другого розділу

У другому розділі кваліфікаційної роботи було вибрано та створено архітектуру інтернет-магазину. Було створено модель даних, необхідну для функціональної роботи веб-сайту. Фронтент та бекенд було розділено на структурні елементи та пророблено роботу над їхнім створенням. Також було додано складні анімації, які вміють взаємодіяти з користувачем та несуть за собою корисний функціонал. Після розробки було проведено тестування та розміщення веб-сайту на хостинг та далі описано досвід використання розробленого програмного забезпечення.

РОЗДІЛ 3. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

3.1 Характеристика життєдіяльності людини у системі «людина-машина – середовище існування»

У теперішній час, світ машин та різноманітних механізмів все більше набуває популярності. Він буквально щодня супроводжує людину. Суспільство стає все залежнішим від цього. Сучасна, складна техніка-механізмів машин, включаючи електронно-обчислювальних. Вони тепер очолюють життєво необхідні системи: військові об’єкти, інформаційні системи, наденергоємні-атомні електростанції та штучні космічні об’єкти.

Машина – це те, за допомогою чого виконуються різні операції. Наприклад операції для зміни місця об’єктів, трансформація енергії, матеріалів, збереження та систематизація інформації [27]. Завдяки цьому інтелектальний та фізичний рівень людини зростає. Існує три основних типи машин:

– роботи;
– енергетичні;
– інформаційні.

Машино-двигуни – це пристрої за допомогою яких енергія будь якого виду перетворюється в механічну. Електродвигуни та двигуни внутрішнього згоряння, турбіни, теплові машини. Група машин поділяється на технологічні та транспортні. Технологічні – металообробні верстати, поліграфічне устаткування, прокатні стани. Транспортні – літаки, автомобілі,конвєйєри, підіймачі, водогони.

Інформаційні машини – самі складні пристрої, розраховані на переробку, одержання та збереження інформації. ЕОМ, механічні інтегратори, арифмометри, персональні комп’ютери. Хоча в повному розумінні слова електронно обчислювальні машини не є машиною, тому що в ній механічні пристрої мають допоміжне значення [28].

Слід зазначити , що більшість сучасних машин недосконалі і тому вони, витвори людини, забруднюють довкілля, змінюють його в негативний бік. Дано

трердження звісно схиляється здебільшого до енергетичних або ж роботичних машин, але все ж тему кваліфікафікаційної роботи дана проблема зачіпає в повній мірі. До прикладу можна навести структурну роботу та архітектуру веб-сайтів, що попередньо розглядалось в розділі другому розділі, а саме в пункті про моделювання архітектури та структури інтернет інтернет-мазазину. В даній частині кваліфікаційної роботи сказано, що клієнт-серверна аріхітектура здебільшого розділена на три ланки, одна з яких серверна частина, що містить в собі всю необхідну інформацію для роботи сервісів, інтернет-магазинів та інших різних платформ, що містять в собі виликий обсях даних. Особливо під цю категорію підпадають онлан-сервіси, які спеціалізуються саме на розміщенні користувацьких даних. Дані веб-сайти потребують наявності значних накопичувальних ресурсів, а ці ресурси як привило – це спільні або приватні мережеві ферми.

Такі ферми споживають дуже багато електроенергії та потребують значної кількості комплектуючих, що у свою чергу дуже погано впливає на екологію. Саме тому розробники зі всього світу працюють над алгоритмами компресії файлів для того щоб зменшити ресурс використання накопичувальних пристроїв. Але все ж слід підкреслити, що ідеальних механізмів небуває, тому навіть найсучасніші на найоптимізованіші програми не зможуть відмінити антропогенний вплив інформаційних машин. З цього можна зробити висновок, що неможливо назвати машину, механізм антропогенного походження, який був би ідеальним і не шкодив би здоров'ю людини, не забруднював би середовище життя.

3.2 Психофізіологічне розвантаження для працівників

У сучасному світі все більше набуває затребуваності професія програміста. До прикладу напрямок веб-дизайн, створення веб-сайтів. Це не тільки програмування та розробка дизайну. Сам процес містить в собі детальний аналіз

проекту, співпраця з замовником, пошук рішень для отримання поставлених цілей проекту.

Враховуючи тривалий час роботи за комп'ютером для працівників необхідно забезпечити психофізіологічне розвантаження, що відбувається в спеціально облаштованих для цього приміщеннях (кімната психологічного розвантаження).

Аутогенне тренування – це метод який будується на свідомому застосуванні сукупності пов'язаних між собою технік психічної саморегуляції та виконанні простих фізичних вправ, додаючи словесне самонавіювання [29]. Саме це тренування пропонується для занять з психофізіологічного розвантаження працівників. Також, кімнати для тренувань обладнують із спеціальним інтер'єром та яскравим оформленням. Сеанс поділено на три етапи, що відповідають стадіям відновлювального процесу.

Перший етап – абстрагування. Працівники повинні розслабитись та абстрагуватись від робочої обстановки, що служить етапу залишкового збудження. Грає спокійна, повільна та мелодійна музика. Можливо звуки моря, шум лісу чи пташиний спів. Зайнявши зручну позу, працівники звикають, адаптуються та психологічно настроюються до наступних етапів.

Другий – затишок (заспокоєння). Цей етап відповідає за регенеративне гальмування. На екрані телевізора чи проектора відбувається показ фотографій чи коротких відео з зображенням квітів, степів, лісів, ставків, пустель чи гладких поверхонь. В цей ж час у навушниках звучить заспокійлива музика, на фоні якої тричі повільно повторюється формула аутогенного тренування:

- « я повністю розслаблений, я спокійний»;
- « моє дихання спокійне, воно рівне»;
- « моє тіло важке, гаряче, повністю розслаблене. У мене легка голова, холодний лоб».

Зелене світло служить функціональним освітленням. Протягом усього сеансу воно поступово знижується. На кінці екран та світло зовсім вимикається на декілька хвилин.

Третій – активізація. Третій період відповідає етапу підвищеної збудженості. Спочатку світло вимкнене. За деякий час на екрані чи моніторі з’являється яскрава пляма червоного кольору. Яскравість та розміри якої поступово збільшуються. На кінці сеансу грає весела та бадьора музика. Тричі повторюються формули аутогенного тренування. Формули включають в себе глибоке вдихання та глибоке довге видихання:

– « я бадьорий, повний енергії, у мене хороший настрій»;

– « я веселий, свіжий та готовий діяти».

За потреби та бажання під час музичної програми пропонується вимовляти окремі фрази. Наприклад, навіювання гарного настрою, відпочинку, сили чи бадьорості.

Заняття з психофізіологічних розвантажень можуть бути проведені за єдиною програмою та складатись із двох етапів. Кожен із них по 5 хвилин:

– перший- абсолютне розслаблення;

– другий- активізація працездатності.

Після занять психофізіологічного розвантаження зменшується втома та з’являються нові сили, бадьорість та гарний настрій, що допомагає веб-розробникам продовжувати працювати над своїми задачами [30].

Ватро звернути увагу, якими методами психофізіологічного розвантаження сучасні IT компанії користуються. До прикладу можна навести компанію Google, яка підтримує найпопулярнішу пошукову систему, займається розробкою програмного забезпечення та володіє найпопулярнішими сервісами, які використовують люди в всіх куточках світу. Люди, які працюють в цій компанії здебільшого розробники програмного забезпечення, зокрема веб-сервісів. Вони дійсно працюють над тим щоб робити життя своїх користувачів краще. Але інша сторона медалі – це психологічне вигорання працівників, адже ця робота вимагаю значну кількість розумових ресурсів, що власне викликає даний наслідок. Для вирішення даної проблеми компанія вжила чимало заходів. До прикладу можна навести:

– безплатна їжа для всіх працівників, що сприяє повноцінному харчуванню та економія часу, який би був витрачений на пошук та замовлення;

– зони відпочинку де працівники можуть відпочити та зайнятись психофізіологічним розвантаженням;

– безплатні спортзали, басейни та все необхідне для заняття спортом, що теж сприяє психофізіологічному розвантаженню;

– можливість працювати в домашніх умовах, у випадках якщо працівникам комфортніше працювати саме так [31].

Саме такий підхід дозволяє Google залучати найкращих. Та оприумувати від працівників набагато більш позитивні відгуки про досвід роботи в прорівнянні з іншими світовими компаніями. Виходячи з даної інформації було зроблено висновок, що рівень психофізичного розвантаження для працівників на дуже високому рівні.

3.3 Висновок до третього розділу

В третьому розділі кваліфікаційної роботи розглянуто таке питання як – характеристика життєдіяльності людини у системі «людина-машина – середовище існування», а саме важливість машин в житті сучасних людей. Також особливу увагу приділено такому типу машин як – інформаційні та їхньому техногенному впливу на екологію.

Висвітлено питання – психофізіологічне розвантаження для працівників, що є дуже важливим, особливо для такої виснажливої роботи як програміст. Подано один із методів психофізичного розвантаження, а саме – аутогенне тренування. Крім того, на прикладі однієї з найвідоміших ІТ компаній, було наведено приклади, того які ідеальні умови для працівників можна зробити, щоб покращувати їхній емоційний стан та давати можливість для психофізіологічного розвантаження і підвищити продуктивність праці.

ВИСНОВКИ

Виконуючи кваліфікаційну роботу було розроблено інтернет-магазин спортивного взуття з використанням таких технологій як HTML5, CSS3, мови програмування PHP на основі фреймворку Laravel та JavaScript для інтеграції складної анімації. Умовно проектування веб-сайту можна поділити на три частини. Це проектування програмного забезпечення, реалізація програмного забезпечення та його тестування.

В першому розділі кваліфікаційної роботи освітнього рівня «Бакалавр»:

– Подано загальну інформацію про найкращі та найпопулярніші існуючі рішення та проаналізовано специфіку їх роботи.

– Висвітлено можливі варіанти і актанти використання та сплановано розробки інтернет магазину на етапи.

– Вибрано середовище розробки та програми за допомогою який буде реалізований інтернет -магазин та обґрунтовано використання технологій наведених вище.

В другому розділі кваліфікаційної роботи:

– Розроблено модель даних для роботи веб-сайту та вибрано архітектуру, яка найбільше підходить для роботи інтернет-магазину.

– Спроектовано серверну та користувацьку частину веб-сайту, подано приклади реалізації сторінок магазину .

– Інтегровано складну інтерактивну анімацію для взаємодії з користувачем, яка дає йому можливість отримати знижку на замовлення.

– Описано процедуру розміщення проекту на один із хостингів з урахуванням можливих помилок та проведено тестування програмного забезпечення.

У розділі «Безпека життєдіяльності, основи хорони праці» було висвітлені проблеми життєдіяльності людини у системі «людина-машина – середовище існування» та описані методи психофізіологічне розвантаження для працівників.

ПЕРЕЛІК ДЖЕРЕЛ

1 Переваги і недоліки інтернет магазину для ведення бізнесу [Електронний ресурс]. – 2013. – Режим доступу до ресурсу: https://officem.com.ua/uk/poleznaja-informatsija/preimuschestva-i-nedostatki-internet-magazina-dlja-vedenija-biznesa.

2 Етапи створення інтернет-магазину [Електронний ресурс]. – 2014. – Режим доступу до ресурсу: https://essuir.sumdu.edu.ua/bitstream-download/123456789/38235/1/Ivanova.pdf.

3 Тестування програмного забезпечення [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення.

4 Що таке SEO оптимізація? [Електронний ресурс]. – 2012. – Режим доступу до ресурсу: https://www.taina.com.ua/shho-take-seo-optymizacija/.

5 Анімація, звук і відео у Веб [Електронний ресурс] – Режим доступу до ресурсу: https://web-design.okis.ru/lektsiia-8-animatsiia-zvuk-ta-video-u-web.

6 PHP [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/PHP.

7 Переваги та недоліки JavaScript [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://hackit-ukraine.com/627-the-advantages-and-disadvantages-of-javascript.

8 Laravel — лідер серед PHP фреймворків, схвалений розробниками [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: http://savelink.org.ua/laravel-lider-sered-php-frejmvorkiv-shvalenij-rozrobnikami/.

9 Синтаксичний цукор [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Синтаксичний_цукор.

10 MySQL/Руководство для начинающих [Електронний ресурс]. – Режим доступу до ресурсу: https://wiki.gentoo.org/wiki/MySQL/Startup_Guide/ru.

11 Основні поняття реляційних БД: нормалізація, зв’язок та ключі [Електронний ресурс]. – 2011. – Режим доступу до ресурсу:

https://bondarenko.dn.ua/osnovni-ponyattya-relyatsijnih-bd-normalizatsiya-zv-yazok-ta-klyuchi/.

12 Объекты данных PHP [Електронний ресурс] – Режим доступу до ресурсу: https://www.php.net/manual/ru/book.pdo.php.

13 Що таке архітектура програмного забезпечення [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: https://soandso.biz/blog/software-engineering/arhitektura-programnogo-zabezpechennya.html.

14 Змерзлий І. Клієнт-серверна архітектура та ролі серверів. [Електронний ресурс] / Іван Змерзлий. – 2017. – Режим доступу до ресурсу: https://medium.com/@IvanZmerzlyi/клієнт-серверна-архітектура-та-роль-серверів-9893d8048229.

15 HTTP [Електронний ресурс] // 2022 – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/HTTP.

16 Структурне програмування [Електронний ресурс] – Режим доступу до ресурсу: https://ua5.org/osnprog/200-strukturne-programuvannja.html.

17 Что такое файлы cookies и зачем они нужны [Електронний ресурс] – Режим доступу до ресурсу: https://ssl.com.ua/blog/what-are-cookies/.

18 HTTP Session [Електронний ресурс] – Режим доступу до ресурсу: https://laravel.com/docs/9.x/session.

19 Різниця між cookie та сесією [Електронний ресурс] – Режим доступу до ресурсу: https://myrusakov.ru/php-cookie-session.html.

20 Проектування інтерфейсу для сайту та онлайн сервісу [Електронний ресурс] – Режим доступу до ресурсу: https://dlogic.com.ua/ua/service/proektuvannya-interfeisiv/.

21 Кольори для дизайну сайту та їх вплив на користувачів [Електронний ресурс] – Режим доступу до ресурсу: https://www.itk-agency.com/kolori-dlya-dizajnu-sajtu-ta-yih-vpliv-na-koristuvachiv/.

22 Структура веб-сайтів. Етапи створення сайту. Засоби автоматизованої розробки веб-сайтів, редактор веб-сайтів з графічним інтерфейсом. Хостинг.

Створення та наповнення веб-сторінки. [Електронний ресурс] – Режим доступу до ресурсу: http://www.vpu20.lviv.ua/images/library/informatyka/okg02.pdf.

23 Руководство по Canvas [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://developer.mozilla.org/ru/docs/Web/API/Canvas_API/Tutorial.

24 JavaScript-анимации [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://learn.javascript.ru/js-animation.

25 phpMyAdmin [Електронний ресурс] – Режим доступу до ресурсу: https://docs.phpmyadmin.net/uk/latest/intro.html.

26 Що таке адмін-панель і навіщо вона потрібна [Електронний ресурс] – Режим доступу до ресурсу: https://hostiq.ua/blog/ukr/admin-panel/.

27 Машина – елемент системи "Людина-Машина-Середовище" [Електронний ресурс] – Режим доступу до ресурсу: https://pidru4niki.com/16280414/bzhd/mashina_element_sistemi_lyudina-mashina-seredovische.

28 Желібо Є. П. Безпека життєдіяльності / Є. П. Желібо, В. В. Зацарний. – Київ: Каравела, 2008. – 344 с.

29 Курдибаха О. М. Вплив аутогенного тренування на психічні процеси [Електронний ресурс] / Оксана Миколаївна Курдибаха. – 2016. – Режим доступу до ресурсу: https://rehab.kyiv.ua/vpliv-autogennogo-trenuvannya-na-psihichni-protsesi/.

30 Основи Охорони Праці / [М. П. Купчик, М. П. Гандзюк, І. Ф. Степанець та ін.]. – Київ, 2000. – 416 с.

31 Мотивація персоналу в Google: секрети успіху [Електронний ресурс]. – 2014. – Режим доступу до ресурсу: https://www.management.com.ua/blog/1509.

ДОДАТКИ

Додаток А

Користувацький інтерфейс

addProdToOrder.blade.php

```
@include('admin.parts.header')
<div class="main">
        <form action="{{route('AddProdQuonInOrder')}}"
method="POST">
            @csrf
            <select name="selet_item">
              @foreach($products as $prod)
                <option value='{ "id": {{$prod->id}}, "title":
"{{$prod->title}}", "price": {{$prod->price}} }' placeholder>{{
$prod->title }}</option>
              @endforeach
            </select><br>

            <input type="number" name='quontity'
placeholder="Quontity" required><br>
            <input type="hidden" name='order_id'
value="{{$order_id}}" placeholder="Quontity"><br>
            <input type="submit" name="minusFromOrder"
value="Edit">
        </form>
    </div>
</body>
</html>
```

addProduct.blade.php

```
@include('admin.parts.header')

<div class="main">
    <form action="{{route('addProductToData')}}" method="POST"
class="order">
        @csrf
        <input type="text" name="title" required=""
placeholder="Title" required>
        <select name='category'>
            @foreach($cats as $cat)
              <option value='{{$cat->id}}' placeholder>{{ $cat-
>ua_name }}</option>
            @endforeach
        </select>
        <input type="text" name="price" required=""
placeholder="Price" required>
        <input type="text" name="name" required=""
placeholder="Name" required>
        <input type="text" name="image" placeholder="Link on image
in folder" required>
```

```
        <input type="text" name="video" placeholder="Link on video
in folder" required>
        <textarea rows="11" cols="37" name="desctiption"
required></textarea>
        <input type="submit" name="addProduct" value="Додати">
    </form>
</div>
</body>
</html>
```

edit.blade.php

```
@include('admin.parts.header')

<div class="main">
        <form action="{{route('EditProd')}}" method="POST"
class="order">
            @csrf
            <input type="text" name="title" required=""
placeholder="Title" value=" {{$edit_product[0]->title}} ">
            <select name='category'>
              <option value="{{$edit_product[0]->cat}}"
placeholder>{{$edit_product[0]->cat}}</option>
              @foreach($cats as $cat)
                <option value="{{$cat->id}}" placeholder>{{$cat-
>id}}</option>
              @endforeach
            </select>
            <input type="text" name="price" required=""
placeholder="Price" value="{{$edit_product[0]->price}}">
            <input type="text" name="name" required=""
placeholder="Name" value="{{$edit_product[0]->name}}">
            <input type="text" name="image" placeholder="Link on
image in folder" value=" {{$edit_product[0]->img}} ">
            <input type="text" name="video" placeholder="Link on
video in folder" value=" {{$edit_product[0]->video}} ">
            <textarea rows="11" cols="37" required=""
name="desctiption"> {{$edit_product[0]->descr}} </textarea>
            <input type="hidden" name="idNumber" value="
{{$edit_product[0]->id}} ">
            <input type="submit" name="editProduct" value="Edit
Product">
        </form>
    </div>
</body>
</html>
```

editOrder.blade.php

```
@include('admin.parts.header')

<div class="main">
```

```
        @foreach($order as $item)
            <table class="table">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>Ім'я</th>
                        <th>Тел</th>
                        <th>E-mail</th>
                        <th>Сума</th>
                        <th>Додати</th>
                        <th>Редагувати</th>
                        <th>Видалити</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>{{ $item->id }}</td>
                        <td>{{ $item->username }}</td>
                        <td>{{ $item->email }}</td>
                        <td>{{ $item->phone }}</td>
                        <td>{{ $item->total_prise }}</td>
                        <td>
                          <form action="{{route('addOrderForm')}}"
method="POST">
                            @csrf
                              <input type="hidden" name="id"
value="{{ $item->id }}">
                              <input type="submit" value="Додати">
                          </form>
                        </td>
                        <td>
                          <form action="{{ route('editOrderForm')
}}" method="POST">
                            @csrf
                              <input type="hidden" name="id"
value="{{ $item->id }}">
                              <input type="submit"
value="Редагувати">
                          </form>
                        </td>
                        <td>
                          <form action="{{ route('dropOrder') }}"
method="POST">
                            @csrf
                              <input type="hidden" name="id"
value="{{ $item->id }}">
                              <input type="submit"
value="Видалити">
                          </form>
                        </td>
                    </tr>
                </tbody>
                <thead>
```

```
                    <tr>
                        <th>Order ID</th>
                        <th>ProdID</th>
                        <th>Title</th>
                        <th>Quotity</th>
                        <th>Price</th>
                        <th>Total Price</th>
                        <th>Drop</th>
                    </tr>
                </thead>
                <tbody>
                    @foreach($odrdersData as $order)
                      @if($item->id == $order->order_id)
                       <tr>
                         <td>{{ $order->id }}</td>
                         <td>{{ $order->product_id }}</td>
                         <td>{{ $order->product_title }}</td>
                         <td>{{ $order->product_quotity }}</td>
                         <td>{{ $order->price }}</td>
                         <td>{{ $order->total_prise }}</td>
                         <td>
                          <form action="{{
route('dropItemInOrder') }}" method="POST">
                               @csrf
                               <input type="hidden" name="id_order"
value="{{ $item->id }}">
                               <input type="hidden"
name="order_total" value="{{ $item->total_prise }}">
                               <input type="hidden" name="id"
value="{{ $order->id }}">
                               <input type="hidden"
name="price_prod" value="{{ $order->total_prise }}">

                               <input type="submit"
value="Видалити">
                          </form>
                         </td>
                       </tr>
                       @endif
                     @endforeach
                </tbody>
            </table>
        @endforeach
    </div>
</body>
</html>
```

editProdInOrder.blade.php

```
@include('admin.parts.header')

<div class="main">
```

```
        <form action="{{ route('EditProdQuonInOrder') }}"
method="POST">
            @csrf
            <select name="selet_item">
              @foreach($products as $prod)
                <option value='{ "prise": {{$prod->price}}, "id":
{{$prod->id}}, "order_id": {{$prod->order_id}}, "old_quontity":
{{$prod->product_quotity}} }' placeholder>{{ $prod->product_title
}}</option>
              @endforeach
            </select><br>

            <input type="number" name='quontity'
placeholder="Quontity" required><br>
            <input type="submit" name="minusFromOrder"
value="Edit">
        </form>
    </div>
</body>
</html>
```

index.blade.php

```
@include('admin.parts.header')

<div class="main">
        <table class="table">
          <thead>
            <tr>
              <th>Назва</th>
              <th>Тип</th>
              <th>Ціна(грн)</th>
              <th>Модель</th>
              <th>Опис</th>
              <th>Редагувати</th>
              <th>Видалити</th>
            </tr>
          </thead>
<!-- foreach ($products as $product) -->
          @foreach($products as $product)
          <tbody>
            <tr>
              <td>{{$product->title}}</td>
              <td>{{$product->cat}}</td>
              <td>{{$product->price}}</td>
              <td>{{$product->name}}</td>
              <td>{{$product->descr}}</td>
                    <td>
                      <form action="{{route('EditItemProd')}}"
method="post">
                          @csrf
```

```
                            <input type="hidden" name="id"
value="{{$product->id}}">
                            <input type="submit" value="Редагувати">
                          </form>
                        </td>
                        <td>
                          <form action="{{route('dropItemProd')}}"
method="POST">
                            @csrf
                            <input type="hidden" name="id"
value="{{$product->id}}">
                            <input type="submit" value="Видалити">
                          </form>
                        </td>
            </tr>
          </tbody>
<!-- end -->
          @endforeach
        </table>
</div>
</body>
</html>
```

login.blade.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet" href="{{ asset('/css/app.css') }}">
    <link rel="stylesheet" href="{{ asset('/css/admin.css') }}">
    <title>Admin Login</title>
</head>
<body>

          <div class="logDiv">
            <br><br><br>
            @if(isset($error))
            <h1>{{$error}}</h1>
            @endif
            <form action="{{ route('admin.var') }}" method="POST">
                @csrf
                <label>Login</label><br>
                <input type="text" name="login"><br>
                <label>Password</label><br>
                <input type="password" name="password"><br>
                <input type="submit" value="Login"
name="loginButton">
            </form>
```

```
            <br>
            <a href="/">Вихід</a>
          </div>
</body>
</html>
```

game.blade.php

```
@include('parts.header')
<div class="main">
  <div class="game_elements">
    <div class="boot_game">
      <canvas id="canvas" class="game_canvas" width="288"
height="512"></canvas>
    </div>
    <button id="game_relode_but" class="game_relode_but"
type="button" name="button">Переграти</button>
    <form action="{{ route('sale') }}" method="POST">
      @csrf
      <input id="score_of_fb" type="hidden" name="sale" value="">
      <button id="get_sale" type="submit" name="button">Отримати:
0%</button>
    </form>
  </div>
    <script src="JS/FB.js">
    </script>
</div>

@include('parts.footer')
```

index.blade.php

```
@include('parts.header')
@include('parts.products')
@include('parts.footer')
```

Додаток Б

Контролери

AdminController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\AdminData;
use Illuminate\Support\Facades\Session;

class AdminController extends Controller
{
  public function LoginPage()
  {
    return view('admin/login');
  }
  public function Login(Request $request)
  {
    $login = $request->input('login');
    $password = $request->input('password');

    $name = AdminData::where(["log" => $login, "pas" =>
$password])->first();

    if(empty($name)){
      // $error = 'Неправильно введені дані !!!';
      return redirect()->route('admin.login');
    }else {
      Session::put('variableName', 'Active');
      $wert = Session::get('variableName');
      return redirect()->route('admin.index');
    }
  }
  public function DropSession(Request $request)
  {
    Session::forget('variableName');
    return redirect()->route('admin.login');
  }
}
```

AdminOrderController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Cats;
use App\Models\OdrdersData;
```

```
use App\Models\Order;
use App\Models\Products;


class AdminOrderController extends Controller
{
  public function AdiminPageOrders()
  {
    $cats = Cats::all();
    $odrdersData = OdrdersData::all();
    $order = Order::all();
    return view('admin/editOrder' , compact('cats', 'odrdersData',
'order'));
  }
  public function DropItemInOrder(Request $request)
  {
    $id_order = $request->input('id_order');
    $order_total = $request->input('order_total');
    $id_prod = $request->input('id');
    $price_prod = $request->input('price_prod');
    $new_prise = floatval($order_total) - (floatval($price_prod));
    Order::where('id', $id_order)->update(['total_prise' =>
$new_prise]);
    OdrdersData::where('id', $id_prod)->delete();
    return redirect('/EditOrder');
  }
  public function DropOrder(Request $request)
  {
    $id = $request->input('id');
    Order::where('id', $id)->delete();
    OdrdersData::where('order_id', $id)->delete();
    return redirect('/EditOrder');
  }
  public function EditOrderForm(Request $request)
  {
    $id = $request->input('id');
    $cats = Cats::all();
    $products = OdrdersData::where(["order_id" => $id])->get();
    return view('admin/editProdInOrder' , compact('cats',
'products'));
  }
  public function EditProdQuonInOrder(Request $request)
  {
    $values = $request->input('selet_item');
    $new_quon = $request->input('quontity');
    $arr = json_decode($values);
    $order = Order::where(["id" => $arr->{'order_id'}])-
>first('total_prise');
    $new_total_prod = $arr->{'prise'} * $new_quon;
    $new_total_order = $order->{'total_prise'} - ($arr->{'prise'}
* $arr->{'old_quontity'}) + $new_total_prod;
    Order::where('id', $arr->{'order_id'})->update(['total_prise'
=> $new_total_order]);
```

```
    OdrdersData::where('id', $arr->{'id'})->update(['total_prise'
=> $new_total_prod, 'product_quotity' => $new_quon]);
    return redirect('/EditOrder');
  }
  public function AddOrderForm(Request $request)
  {

    $order_id = $request->input('id');
    $cats = Cats::all();
    $products = Products::all();
    return view('admin/addProdToOrder' , compact('cats',
'products', 'order_id'));

  }
  public function AddProdQuonInOrder(Request $request)
  {
    $values = $request->input('selet_item');
    $arr = json_decode($values);
    $quontity = $request->input('quontity');
    $order_id = $request->input('order_id');
    $total_prise = $quontity * $arr->{'price'};
    $order = Order::where(["id" => $order_id])-
>first('total_prise');
    $new_order_total_prise = $order->{'total_prise'} +
$total_prise;

    OdrdersData::create([
    'order_id' => $order_id,
    'product_id' => $arr->{'id'},
    'product_title' => $arr->{'title'},
    'product_quotity' => $quontity,
    'price' => $arr->{'price'},
    'total_prise' => $total_prise
    ]);
    Order::where('id', $order_id)->update(['total_prise' =>
$new_order_total_prise]);
    return redirect('/EditOrder');
  }
}
```

CatsController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Cats;
use App\Models\Products;


class CatsController extends Controller
```

```
{
  public function allDataCats()
  {
     $cats = Cats::all();
     $products = Products::all();
     return view('index', compact('cats', 'products'));
  }
  public function sortByCats($slug)
  {
    $cats = Cats::all();
    $products = Products::where('cat', $slug)->get();
    return view('index', compact('cats', 'products'));
  }
  public function oneProduct($slug)
  {
    $cats = Cats::all();
    $product = Products::where('id', $slug)->firstOrFail();
    return view('oneProduct', compact('cats', 'product'));
  }
  public function game()
  {
     $cats = Cats::all();
     $products = Products::all();
     return view('game', compact('cats', 'products'));
  }
  public function cart()
  {
    if(isset($_COOKIE['product'])){
      $data = json_decode($_COOKIE['product'], true);
    }else {
      $data = [];
    }
     $cats = Cats::all();
     $products = Products::all();
     return view('cart', compact('cats', 'products', 'data'));
  }
}
```

OrderController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Order;
use App\Models\OdrdersData;

class OrderController extends Controller
{
  public function AddOrder(Request $request)
  {
```

```
    $data = json_decode($_COOKIE['product'], true);
    $total_prise = intval(json_decode($_COOKIE['total_prise'],
true));

    $username = $request->input('username');
    $phone = $request->input('phone');
    $email = $request->input('email');

    Order::create([
    'username' => $username,
    'email' => $phone,
    'phone' => $email,
    'total_prise' => $total_prise
    ]);

    $last_id = Order::orderBy('created_at', 'desc')->first();
    $last_id = $last_id['id'];

    for ($i=0; $i < count($data); $i++) {
      $total_prise = $data[$i][3] * $data[$i][2];
      OdrdersData::create([
      'order_id' => $last_id,
      'product_id' => $data[$i][0],
      'product_title' => $data[$i][1],
      'product_quotity' => $data[$i][3],
      'price' => $data[$i][2],
      'total_prise' => $total_prise
      ]);
    }
    unset($_COOKIE['product']);
    setcookie('product', '', time() - 3600, '/');
    unset($_COOKIE['total_prise']);
    setcookie('total_prise', '', time() - 3600, '/');
    unset($_COOKIE['sale']);
    setcookie('sale', '', time() - 3600, '/');
    return redirect('/');
  }
}
```

Додаток В

Анімації інтернет-магазину

FB.js

```
var canvas = document.getElementById("canvas");
var context = canvas.getContext("2d");

var bird = new Image();
var background = new Image();
var foreground = new Image();
var pipeUp = new Image();
var pipeDown = new Image();
var sale_persent = new Image();
var game_lose = new Image();
var winner = new Image();
var upAudio = new Audio();
var pointsAudio = new Audio();
var score_for_sale = document.getElementById('score_of_fb');
var game_relode_but = document.getElementById('game_relode_but');
var get_sale = document.getElementById('get_sale');

bird.src = "img_game/boot.png";
background.src = "img_game/background.png";
foreground.src = "img_game/fg.png";
pipeUp.src = "img_game/down.png";
pipeDown.src = "img_game/up.png";
sale_persent.src = "img_game/sale.png";
game_lose.src = "img_game/lose.png";
winner.src = "img_game/win.png";
upAudio.src = "audio_game/fly.mp3";
pointsAudio.src = "audio_game/score.mp3";

var distance = 90;
var xb = 10;
var yb = 150;
var grav = 2;
var points = 0;

document.addEventListener("keydown", function(event) {
  if (event.code == 'Space') {
    yb -= 40;
  }
})

var pipes = [];
pipes[0] = {
  x : canvas.width,
  y : 0
}

function draw() {
```

```
  context.drawImage(background, 0, 0);
  for (var i = 0; i < pipes.length; i++) {
    context.drawImage(pipeUp, pipes[i].x, pipes[i].y);
    context.drawImage(sale_persent, pipes[i].x, pipes[i].y + 270,
50, 50);
    context.drawImage(pipeDown, pipes[i].x, pipes[i].y +
      pipeUp.height + distance);
    pipes[i].x--;
    if (pipes[i].x == 125) {
      pipes.push({
        x : canvas.width,
        y : Math.floor(Math.random() * pipeUp.height) -
pipeUp.height
      });
    }
    if (xb + bird.width >= pipes[i].x &&
    xb <= pipes[i].x + pipeUp.width &&
    (yb <= pipes[i].y + pipeUp.height ||
    yb + bird.height >= pipes[i].y + pipeUp.height + distance)
    || yb + bird.height  >= canvas.height - foreground.height){
      return context.drawImage(game_lose, 0, 0, 288, 512)
      // location.reload();
    }
    if(points == 20){
      return context.drawImage(winner, 0, 0, 288, 512)
    }
    if (10 == pipes[i].x){
      points++;
      pointsAudio.play();
      get_sale.innerHTML = 'Отримати: ' + points/2 + '%'
    }
  }
  context.drawImage(foreground, 0, canvas.height -
foreground.height);
  context.drawImage(bird, xb, yb, 38, 26);
  yb += grav;
  context.fillStyle = "#ffd359"
  context.font = "24px Verdana"
  context.fillText(points/2, 10 , canvas.height - 23)
  if(points > 0){
    score_for_sale.value = points/2;
  }
  context.drawImage(sale_persent, 60 , canvas.height - 60, 50, 50)
  requestAnimationFrame(draw);
}
game_relode_but.onclick = function(){
  location.reload();
};
pipeDown.onload = draw;
```

sim-slider.js

```
function Sim(sldrId) {
    let id = document.getElementById(sldrId);
    if(id) {
        this.sldrRoot = id
    }
    else {
        this.sldrRoot = document.querySelector('.sim-slider')
    };
    // Carousel objects
    this.sldrList = this.sldrRoot.querySelector('.sim-slider-
list');
    this.sldrElements = this.sldrList.querySelectorAll('.sim-
slider-element');
    this.sldrElemFirst = this.sldrList.querySelector('.sim-slider-
element');
    this.leftArrow = this.sldrRoot.querySelector('div.sim-slider-
arrow-left');
    this.rightArrow = this.sldrRoot.querySelector('div.sim-slider-
arrow-right');
    this.indicatorDots = this.sldrRoot.querySelector('div.sim-
slider-dots');
    // Initialization
    this.options = Sim.defaults;
    Sim.initialize(this)
};

Sim.defaults = {

    loop: true,     // Бесконечное зацикливание слайдера
    auto: true,     // Автоматическое пролистывание
    interval: 5000, // Интервал между пролистыванием элементов
(мс)
    arrows: true,   // Пролистывание стрелками
    dots: true      // Индикаторные точки
};

Sim.prototype.elemPrev = function(num) {
    num = num || 1;

    let prevElement = this.currentElement;
    this.currentElement -= num;
    if(this.currentElement < 0) this.currentElement =
this.elemCount-1;
    if(!this.options.loop) {
        if(this.currentElement == 0) {
            this.leftArrow.style.display = 'none'
        };
        this.rightArrow.style.display = 'block'
    };

    this.sldrElements[this.currentElement].style.opacity = '1';
    this.sldrElements[prevElement].style.opacity = '0';
```

```
    if(this.options.dots) {
        this.dotOn(prevElement); this.dotOff(this.currentElement)
    }
};
```

Додаток Г

Стилі використані в проекті

app.css

```
* {
  font-family: "Arial", sans-serif;
  font-size: 20px;
}
body {
  margin-top: 20px;
  /* background-color: white; */
  background-image: url("https://image.freepik.com/vector-
gratis/visualizacion-datos-dinamico-vector-patron-onda_53876-
43475.jpg");
  /* background-image: url("https://image.freepik.com/free-
vector/white-abstract-background-theme_23-2148830883.jpg"); */
  background-repeat: no-repeat;
  background-attachment: fixed;
  -webkit-background-size: 100%;
}

a {
  text-decoration: none;
  text-transform: uppercase;
  color: #000000;
  -webkit-transition: all 0.3s;
  -moz-transition: all 0.3s;
  -ms-transition: all 0.3s;
  -o-transition: all 0.3s;
  transition: all 0.3s;
}

a:hover {
  color: red;
}

select {
  margin: 5px auto;
  border: none;
  padding: 10px;
}

input,
button {
  margin: 5px auto;
  padding: 10px;
  background-color: white;
  -webkit-transition: all 0.3s;
  -moz-transition: all 0.3s;
  -ms-transition: all 0.3s;
  -o-transition: all 0.3s;
  transition: all 0.3s;
```

```
}

.small-input {
  padding: 3px;
}

input[type="submit"],
button {
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
  display: block;
  cursor: pointer;
  background-color: white;
}

input[type="submit"]:hover,
button:hover {
  background-color: red;
}

.delete {
  display: inline-block;
}

label,
.label {
  margin-top: 10px;
  font-weight: 700;
}
.textPic{
  margin-top: 10px;
  font-weight: 700;
}

h2 {
  font-size: 25px;
}

img {
  width: 100%;
}
.main{
  padding-bottom: 80px;
}
.main,
nav {
  width: 1170px;
  margin: auto;
  display: flex;
  flex-wrap: wrap;
  justify-content: space-around;
}
```

```
nav ul {
  margin: 10px;
  display: flex;
  list-style: none;
  width: 100%;
  justify-content: space-around;
  align-items: center;
  font-weight: 700;
}

.card {
  margin-bottom: 15px;
  width: 20%;
  padding: 10px;
  border: 2px solid black;
  border-radius: 15px;
  text-align: center;
  background-color: #fff;
  display: flex;
  flex-direction: column;
  justify-content: space-between;
  /* background-color: pink; */
}

.cart,
.product-card {
  text-align: center;
  width: 900px;
  margin: auto;
}

.cart-title {
  text-align: center;
}

.descr {
  margin-bottom: 20px;
}

.product-card img {
  width: 500px;
}

.cart {
  margin-bottom: 10px;
  display: flex;
  justify-content: flex-start;
  align-items: center;
}

.cart img {
  width: 100px;
  align-self: center;
```

```
}

.cart-descr {
  margin: 0 10px;
  width: 80%;
}

.order {
  display: flex;
  flex-direction: column;
}

.back {
  display: block;
  margin: 10px auto;
  text-align: center;
}
.seper{
  width: 3px;
  height: 60px;
  background-color: black;
  margin-left: 30px;
  margin-right: 30px;
}
.imgshop{
  height: 40px;
  width:  25px;
}
.shopdiv{
  display: flex;
  flex-direction: row;
  align-items: center;
}

.table {
  width: 100%;
  margin-bottom: 30px;
  border-collapse: collapse;
}
.table th {
  font-weight: bold;
  padding: 5px;
  background: white;
  border: 3px solid #363636;
}
.table td {
  border: 1px solid #363636;
  padding: 5px;
}
.headerImg{
  width: 100px;
}
.footdiv{
```

```
  height: 70px;
  width: 70%;
  display: flex;
  justify-content: flex-start;
}
.social1{
  height: 100%;
  width: 20%;
  border: 2px solid #363636;
  border-radius: 15px;
  background: white;
  margin-left: 1.7%;
  margin-right: 1.7%;
  display: flex;
  justify-content: center;
  align-items: center;
  font-size: 14px;
}
.social1IMG{
  height: 50px;
  width: 50px;
}
.social{
  height: 100%;
  width: 60%;
  border: 2px solid #363636;
  border-radius: 15px;
  background: white;
  margin-left: 1.7%;
  margin-right: 1.7%;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
.social4{
  height: 100%;
  width: 60%;
  margin-left: 1.7%;
  margin-right: 1.7%;
  display: flex;
  flex-direction: row;
  align-items: center;
  justify-content: center;
}
.footseperatordiv{
  height: 30px;
  width: 2px;
  background: black;
  margin-left: 2%;
  margin-right: 2%;
}
.adminL a{
```

```
  font-size: 20px;
}
.social4{
  font-size: 14px;
}
.logDiv{
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
.social3{
  height: 100%;
  width: 20%;
  border: 2px solid #363636;
  border-radius: 15px;
  background: white;
  margin-left: 1.7%;
  margin-right: 1.7%;
  font-size: 14px;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
}
footer {
    position: fixed; /* Фиксированное положение */
    display: flex;
    align-items: center;
    justify-content: center;
    left: 0; bottom: 0; /* Левый нижний угол *//* Цвет фона */
    width: 100%; /* Ширина слоя */
    padding-bottom: 10px;
}
.class_before_futter{
  height: 100px;
}
```

fb.css

```
.game_elements{
  width: 800px;

  display: flex;
  flex-direction: column;
  justify-content: space-between;
  align-items: center;

  padding: 10px;
```

```
  /* background: url("https://media.istockphoto.com/vectors/pixel-art-style-noise-seamless-vector-background-colorful-vector-id530769666"); */
  background: url("https://pbs.twimg.com/media/EZ783JZXQAYMXI2.jpg");

  background-repeat: no-repeat;
  background-size: cover;
  border-radius: 20px;
  border: 2px solid black;
}
.boot_game{
  width: 288px;
  height: 512px;

  display: flex;
  flex-direction: column;
  justify-content: center;
}
.game_canvas{
  border-radius: 20px;
  border: 2px solid black;
}
.game_relode_but{
  margin-top: 10px;
  width: 150px;
  height: 50px;
  background-color: white;
  border-radius: 10px;
  border: 2px solid black;
  align-items: center;
}
.game_relode_but:hover{
  background-color: green;
  color: white;
}
```

sim-slider-styles.css

```
img {
    width: 100%;
}

.sim-slider {
    max-width: 1000px;
    min-width: 320px;
    height: 500px;
    margin: 20px auto;
    padding: 30px 50px;
    border: 2px solid black;
    border-radius: 15px;
    background-color: white;
```

```
}

.sim-slider {
    position: relative;
}

.sim-slider-list {
    margin: 0;
    padding: 0;
    list-style-type: none;
    position: relative;
}

.sim-slider-element {
    width: 100%;
    transition: opacity 1s ease-in;
    opacity: 0;
    position: absolute;
    z-index: 2;
    left: 0;
    top: 0;
    display: block;
}

/* Navigation item styles */

div.sim-slider-arrow-left,
div.sim-slider-arrow-right {
    width: 22px;
    height: 40px;
    position: absolute;
    cursor: pointer;
    opacity: 0.6;
    z-index: 4;
}

div.sim-slider-arrow-left {
    left: 10px;
    top: 40%;
    display: block;
    /* background: url("http://pvbk.spb.ru/inc/slider/sim-
files/sim-arrow-left.png%22") no-repeat; */
    /* background-color: black; */

    background: url("../img/right.png") no-repeat;
    -webkit-transform: rotate(180deg);
    background-size: 100%;
}

div.sim-slider-arrow-right {
    right: 10px;
    top: 40%;
    display: block;
```

```
    /* background: url("img/right.png) no-repeat; */

    background: url("../img/right.png") no-repeat;
    background-size: 100%;
}

div.sim-slider-arrow-left:hover {
    opacity: 1.0;
}

div.sim-slider-arrow-right:hover {
    opacity: 1.0;
}

div.sim-slider-dots {
    width: 100%;
    height: auto;
    position: absolute;
    left: 0;
    bottom: 0;
    z-index: 3;
    text-align: center;
}
```