ДИПЛОМНА РОБОТА

ДР.ПІс – 18.00.000 ПЗ

Група ПІс-2015

Савчин І. В.

2019

ПВНЗ Університет Короля Данила

Кафедра Інформаційних технологій та програмної інженерії

УДК 004

ДИПЛОМНА РОБОТА

Тема ***Розробка інтернет-магазину брендованої продукці для Університету Короля Данила***

Спеціальність ***6.050103 «Програмна інженерія»***
(код і назва спеціальності)

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДР.ПІс – 18.00.000 ПЗ
(позначення)

Студент

Савчин І.В.
(підпис) (дата) (розшифрування підпису)

Керівник проекту
к.т.н. Пашкевич О.П.
(посада) (підпис) (дата) (розшифрування підпису)

Нормоконтроль
к.т.н. МанулякІ.З.
(посада) (підпис) (дата) (розшифрування підпису)

Допускається до захисту
Завідувач кафедри
д.т.н.,доц. Мельничук С.І.
(посада) (підпис) (дата) (розшифрування підпису)

2019

ПВНЗ УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА

Факультет *Інформаційних технологій*
Кафедра *Інформаційних технологій та програмної інженерії*
Спеціальність *6.050103 «Програмна інженерія»*

ЗАТВЕРДЖУЮ:
Завідувач кафедри ІТПІ
_____ *С.І. Мельничук*
“____” _____ 2019 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Студенту *Савчину Івану Васильовичу*

1. Тема проекту (роботи) *Розробка інтернет-магазину брендованої продукції для Університету Короля Данила*

Затверджена наказом ректора Університету Короля Данила від *15.11.2018 р. № 20/4*
2. Термін здачі студентом закінченого проекту (роботи) *10.06.2019 р.*
3. Вихідні дані до проекту (роботи) *JavaScript, VueJs фреймворк, база даних.*

4. Зміст пояснювальної записки (перелік питань, що їх належить розробити) *1. Загальний огляд інтернет-магазинів. Розгляд аналогів. 2. Розробка алгоритмів та взаємозв’язків між функціональними об’єктами. 3. Розробка основних компонентів, методів та огляд інтерфейсу адміністратора таі користувача.*

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень) *Постановка завдання, структура веб-додатку, головна сторінка, структура бази даних, вигляд особистого кабінету адміністратора і користувача.*

6. Консультанти з проекту (роботи), із зазначенням розділів проекту, що стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 30.10.2018

Керівник _____ *Пашкевич О. П*

Завдання прийняв до виконання _____ Савчин І. В.

КАЛЕНДАРНИЙ ПЛАН

Пор №	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1.	Опис наявних сайтів аналогів, їх алгоритми роботи та опис використаних технологій	30.11.2018	
2.	Розробка алгоритмів та взаємозв’язків між функціональними об’єктами	25.12.2018	
3.	Розробка допоміжних функцій та інтерфейсів для веб-сайту наукової організації	21.02.2019	
4.	Оформлення пояснювальної записки	31.05.2019	
5.	Оформлення графічного матеріалу та підготовка до захисту роботи	10.06.2019	

Студент-дипломник _____ *Савчин І. В.*
(підпис) (розшифровка підпису)

Керівник проекту _____ *Пашкевич О. П.*
підпис) (розшифровка підпису)

АНОТАЦІЯ

Розрахунково-пояснювальна записка: 89 сторінки, 26 рисунків, 1 таблиць, 2 додатки.

Об'єкт дослідження – Інтернет магазин брендованої продукції для Університету Короля Данила.

Мета роботи - проаналізувати специфіку діяльності Інтернет магазинів, розробити web-сайт та базу даних «Інтернет-магазину брендованої продукції для Університете Короля Данила».

Відповідно до поставленого завдання в роботі проводиться дослідження процесу створення інтернет-магазину. Виконання поставлених завдань а також вирішень певних питань. Для полегшення даного процесу та зменшення рутинної роботи та затрачуваного часу розроблено веб-сайт для створення, продажу брендовоної продукції для Університету Короля Данила.

SUMMARY

Settlement and explanatory note: 89 pages, 26 drawings, 1 table, 2 attachments.

The object of the research is the Internet store of branded products for the King Danylo University.

The purpose of the work is to analyze the specifics of the activities of Internet shops, to develop a website and database of the «Internet store of branded products for the King Danylo University».

In accordance with the set task in the work is conducted research of the process of creating an online store. Completing tasks as well as solving certain issues. To facilitate this process and reduce routine work and time spent developing a website for the creation, sale of branded products for the King Danylo University.

РЕФЕРАТ

Інтернет-магазин - це реалізоване в мережі Інтернет представництво, шляхом створення Web-сервера для продажу товарів і послуг іншим користувачам мережі Інтернет. Інакше кажучи, Інтернет-магазин - це співтовариство територіально роз'єднаних співробітників магазина (продавців, касирів) і покупців, які можуть спілкуватися і обмінюватися інформацією через електронні засоби зв'язку при повній (або мінімальній) відсутності особистого прямого контакту. Електронна торгівля у Інтернет-магазині ґрунтується на тій же структурі, що і традиційна торгівля. Переваги Інтернет-магазина перед реальним очевидні. Зменшується чисельність персоналу за рахунок скорочення об'єму взаємодії з клієнтами, оренда дискового простору і розміщення «електронної вітрини» дешевше і простіше за оренду торгових приміщень і розміщення товарів на полицях, немає потреби в касовому обслуговуванні і т.д.

Електронні магазини істотно зменшують витрати виробника, заощадивши на утриманні звичайного магазину, розширюють ринки збуту, так само як і розширює можливість покупця - купувати будь-який товар в будь-якій країні, в будь-якому місті, в будь-який час доби, в будь-який час року. Це дає електронним магазинам переваги перед звичайними магазинами. Цей момент є істотним при переході виробників з «звичайною» торгівлі на «електронну».

Інтернет-ресурс дозволяє зблизитись з потенційним клієнтом, завдяки структурі web-порталу, дозволяє розмістити величезну кількість різних товарів і опис кожного товару, відгуки та пропозиції по кожному продукту.

Користуються величезною популярністю Інтернет-магазини одягу, меблів, книг і багатьох інших товарів. Тому що для покупця дуже зручно вибирати і замовляти товар, не виходячи з дому. Більш того, створюючи Інтернет-магазин, ви економите на оренді торгового приміщення, товар знаходиться на складі. Це успішний початок бізнесу, що розвивається. Обрана мною тема вважається актуальною на сьогоднішній день, так як сьогодні мільйони людей щодня, не виходячи з дому, купують різні товари в електронних магазинах. У світі, а зокрема в України величезними темпами зростає кількість користувачів Internet і як наслідок кількість «електронних» покупців, потенційних «електронних» покупців.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ 10
ВСТУП 11
1 ЗАГАЛЬНИЙ ОГЛЯД ІНТЕРНЕТ-МАГАЗИНІВ. РОЗГЛЯД АНАЛОГІВ . 13
1.1 Призначення та мета проекту 13
1.2 Переваги та недоліки Інтернет-магазинів 15
1.3.1 Переваги інтернет-магазинів 16
1.3.2 Недоліки інтернет-магазинів 17
1.4 Визначення технологій 18
1.5 Розгляд аналогів 19
1.5.1 Розгляд інтернет-магазину «Сайтег» 19
1.5.2 Розгляд інтернет-магазину «Розетка» 21
2 РОЗРОБКА АЛГОРИТМІВ ТА ВЗАЄМОЗВ'ЯЗКІВ МІЖ ФУНКЦІОНАЛЬНИМИ ОБ'ЄКТАМИ 25
2.1 MySQL. Основні можливості MySQL 25
2.2 Опис структури бази даних 29
3 РОЗРОБКА ОСНОВНИХ КОМПОНЕНТІВ, МЕТОДІВ ТА ОГЛЯД ІНТЕРФЕЙСУ АДМІНІСТРАТОРА ТА КОРИСТУВАЧА 35
3.1 Опис інтерфейсу інтернет -магазину 35
3.1.1 Опис інтерфейсу для адміністратора та продавця 43
3.1.2 Опис інтерфейсу для користувача 49
ВИСНОВКИ 54
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 56
Додаток А 58
Додаток Б 74

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

CSS – Cascading Style Sheets

HTML – HyperText Markup Language

JS – JavaScript

PHP – Hypertext Preprocessor

MySQL – система управління базами даних (СУБД)

ORM – Object-Relational Mapping

ВСТУП

Розробка Інтернет-магазину та його використання є актуальним питанням на сьогоднішній день, оскільки мільйони людей щодня, не виходячи з дому, купують різні товари в електронних магазинах. В світі, а зокрема в Україні величезними темпами росте кількість користувачів Інтернет і, як наслідок, кількість «електронних» покупців.

Інтернет зближує, реакція на будь-яку подію поступає практично негайно, а відстані зникають. Інтернет-магазини істотно зменшують витрати виробника, заощадивши на утриманні звичайного магазина, розширюють ринки збуту, так само як і розширює можливість покупця - купувати будь-який товар у будь-який час в будь-якій країні, в будь-якому місті, у будь-який час доби. Це дає Інтернет-магазинам перевагу перед звичайними магазинами. Цей момент є істотним під час переходу виробників із «звичайної» торгівлі на «електронну».

Інтернет - магазини, що пережили кризу, укріпили свої позиції на ринку, за рахунок скуповування менш вдалих конкурентів. Гіганти оффлайнового бізнесу, в 1999 році що коштували в десятки разів дешевше за своїх онлайнових побратимів, після кризи теж дістали можливість вийти на ринок Інтернету, запропонувавши вищу якість послуг і гучне ім'я.

В Україні у сфері електронної комерції традиційно працюють фірми по наданню послуг доступу в Інтернет: web-портали, пошукові машини, служби web-пошти, нові Інтернет-агентства. Тут немає ні лідерів оффлайнового ринку, ні представництв крупних міжнародних Інтернет-компаній .

Традиційний ринок Інтернет-комерції зароджувався як ринок роздрібної торгівлі. Але поступово на ринку стали з'являєтися рішення, орієнтовані не на кінцевих споживачів, а на організації, так званий ринок B2b, або business-to-business.

Спочатку, на ринку B2b пропонувалися продукти, пов'язані власне з організацією роздрібної торгівлі (готові Інтернет-магазини, послуги з реклами, впровадження Інтернет-торгівлі в традиційні бізнес-процеси і тому подібне).

Але поступово через Інтернет почали продаватися рішення, безпосередньо з Інтернет не зв'язані (комерційне програмне забезпечення, послуги з автоматизації, оптова торгівля, брокерські послуги, і тому подібне).

З всього вище сказаного однозначно зрозуміла актуальність питання дослідження Інтернет-магазину та його розробки з використанням сучасних Internet-технологій. В рамках дипломного проекту перед нами було поставлено завдання розробити Web-сайт «Інтернет-магазин».

1 ЗАГАЛЬНИЙ ОГЛЯД ІНТЕРНЕТ-МАГАЗИНІВ. РОЗГЛЯД АНАЛОГІВ

1.1 Призначення та мета проекту

Основним призначенням проекту, є надання університету, можливість просто та швидко продавати брендовану продукцію Короля Данила (одяг, символіку університету, т. д.).

Також для іншої сторони (покупців) створено зручну систему відслідковування замовлень та керування ними.

Всі організаційні моменти по доставці та комунікації з клієнтами лягають на плечі продавців, так як проект представляє з себе лише зручний та якісний інструмент, а як ним розпоряджатися – вибирають продавці.

1.2 Інтернет-магазини та їх особливості

Інтернет-магазин (електронний магазин, онлайновий магазин) – це програмний комплекс, який дозволяє продавати товари чи послуги через мережу Інтернет та автоматизувати управління бізнес-процесами. Електронні магазини об'єднують елементи прямого маркетингу та традиційної торгівлі. Основними відмінностями Інтернет-магазину від традиційного є інтерактивність, велика кількість інформації та асортименту продукції і персоналізований підхід до кожного відвідувача. Найбільшим недоліком електронних магазинів є те, що не можна торкнутися товару та оцінити його візуально. Проте, цей недолік з успіхом компенсується великою кількістю інформації, яку не зможе надати продавець в традиційному магазині.

Електронна торгівля у віртуальному магазині ґрунтується на тій же структурі, що і традиційна торгівля.

Таблиця 1.1 – Порівняльна характеристика традиційної і електронної торгівлі

Традиційний магазин	***Інтернет - магазин***
Торговий зал	Інтернет - магазин
Ходьба покупця по торговому залу і огляд товарів на полицях магазина	Перегляд покупцем сторінок сервера
Особистий контакт покупця з продавцем (консультація)	Консультація у продавця (при необхідності) по комп'ютерній мережі або по телефону
Вибір покупцем товару	Вибір покупцем товару
Замовлення товару	Замовлення товару через сервер
Виписка продавцем і вручення покупцеві рахунку на оплату	Пересилка продавцем по комп'ютерних мережах покупцеві рахунку на оплату
Оплата покупцем рахунку на товар в касі магазина готівкою або банківською картою	Оплата покупцем рахунку по якій-небудь системі електронних платежів (банківська карта, електронний чек, цифрові гроші, електронні гроші)

Основними функціями електронного магазину є:

– надання якомога повнішої інформації про представлені товари та послуги;

– прийом та обробка замовлень;

– персоналізація відвідувачів;

– проведення платежів (за умови підключення до платіжної системи);

– збір та аналіз статистичної інформації.

Інтернет-магазин – це сайт, оснащений он-лайн-вітриною, що дозволяє приймати замовлення не тільки по телефону, але і через сайт на електронну пошту.

Інтернет-магазин дозволяє оформляти взаєморозрахунки з покупцями, як за готівку, так і за безготівковим розрахунком з можливістю роздруківки рахунку-фактури і вибором «з ПДВ» і «без ПДВ»

Інтернет-магазин також включає всі функції вирішення корпоративного сайту, що дозволяє в рамках цього рішення сумістити повноцінний корпоративний сайт з роздрібним або оптовим Інтернет-магазином.

Інтернет-магазин підійде для тих, хто хоче ефективно почати продажі, швидко і недорого організувавши он-лайн-вітрину з функцією замовлення товарів через Інтернет.

1.2 Переваги та недоліки Інтернет-магазинів

За останні роки практично повсюдного поширення набуло придбання будь-якої продукції в Інтернет-магазинах. Це дуже серйозний крок на шляху до прогресу, перш за все тому, що він дає людям можливість уникнути необхідності поїздок по стаціонарних торгових точках. Звичайно, прості магазини як існували раніше, так й існують зараз і вони завжди будуть в наявності, поки є люди, які отримують задоволення від «шопінгу». Проте така можливість далеко не у всіх. Наприклад, більшість людей настільки зайнята своєю роботою і вирішенням якихось побутових проблем, що у них просто немає вільного часу на походи по салонах і магазинах. Але що ж робити, якщо будь-яку серйозну покупку, наприклад, комп'ютер все ж таки необхідно зробити? Скористатися послугою Інтернет-магазина. По-перше, це гарантія отримання дійсно якісного сертифікованого товару. По-друге, це шанс уникнути виснажливих пошуків. По-третє, можна заощадити, оскільки, найчастіше вартість товару в Інтернет-магазинах хоч трохи, але нижче звичайної. І, нарешті, по-четверте, можна знайти все, що завгодно, оскільки в Інтернеті можна придбати практично все. Так само не варто недооцінювати можливість безготівкового розрахунку, адже це дуже зручно, але на жаль, далеко не всі стаціонарні торгові точки, дають своїм покупцям такий шанс. Інтернет-магазин, дуже потрібна і важлива річ. Відповідно, якщо є попит, значить, буде і пропозиція. Чисельність будь-яких

Інтернет-магазинів росте в мережі Інтернет з кожним днем. Це дуже вигідна можливість розширення власного бізнесу шляхом збільшення об'єму продажі.

1.3.1 Переваги інтернет-магазинів

Найбільш очевидною перевагою електронної торгівлі є можливість «проходжуватися» по магазинах в Інтернет, не виходячи при цьому з будинку.

Які ж мотиви рухають людьми що визначають можливість зробити замовлення в магазині, не виходячи з будинку, як головна перевага Інтернет-торгівлі. Таким мотивом виявилося небажання витрачати час на відвідини звичайних магазинів, тобто економія часу.

Економія часу. У Інтернет можна вибирати і порівнювати характеристики товарів серед декількох магазинів, і дана процедура займає хвилини на відміну від утомливих походів по традиційних магазинах. Покупка необхідного товару через Інтернет економить такий вельми дорогий час. Інтернет-магазин доступний 24 години в добу, 7 днів в тиждень, 365 днів в році. Працює без свят і вихідних, черг там немає. Покупець може у будь-який час відірватися від вибору товару, якщо в даний момент у нього немає часу, і повернутися до вибору навіть через декілька діб (всі дані по вибраних позиціях залишаються в аккаунті користувача). Продовжити вибір можна у вільний час - на робочому місці або удома, увечері або вночі.

Низькі ціни. Більшість товарів в мережі Інтернет можна придбати за нижчу ціну, ніж в звичайному супермаркеті. Чим це обумовлено? По-перше, у Інтернет-магазині немає необхідності орендувати торгові площі, по-друге, відсутні витрати по охороні і змісту торгових залів, немає потреби витрачати гроші на найм і навчання продавців-консультантів, по-третє, не потрібно оплачувати послуги посередників.

Свобода вибору. У відвідувачів Інтернет - магазинів існує повна, нічим не обмежена свобода вибору. Сформована заявка поступає не на склад рядового

роздрібного магазину, а на найбільший оптовий склад, де є величезний вибір товарів. У Інтернет - магазині можна ознайомитися з інструкціями по застосуванню і подивитися фотографії товарів, не покидаючи улюбленого крісла перед монітором комп'ютера. Звичайно, не можна ознайомитися з товаром «живцем», відчути його руками, але є можливість сходити в звичайний магазин, подивитися на уподобаний товар, ще раз зважити все «за і проти», й істотно заощадити значну суму грошей при покупці через Інтернет - магазин.

Зручність оплати. Клієнти можуть вибрати найоптимальніший спосіб оплати вибраного товару: готівкою при доставці кур'єром, поштовий або банківський переказ, оплата кредитною картою або електронними грошима (Webmoney, та ін.). Вони можуть відмовитися від покупки, навіть не оплачуючи витрати при доставці.

1.3.2 Недоліки інтернет-магазинів

Складна система замовлення. Складна система замовлення полягає в довгому і заплутаному процесі оформлення покупки. Покупцеві необхідно заповнити довгу і часто не зовсім зрозумілою форму, внаслідок чого витрачається багато часу в Інтернет, який не завжди дешево стоїть.

Також можна відзначити повільну швидкість завантаження деяких магазинів, що може послужити причиною відходу з магазину.

Нестабільний асортимент. Відповідно, отримати конкурентну перевагу зможе той магазин, який запропонує користувачам весь необхідний ним асортимент.

Крім того, нестабільний асортимент деяких магазинів, тобто коли після вибору товару і оформлення покупки виявлялося, що потрібного товару немає на складі або просто магазин так і не зв'язався з покупцем.

Необхідність реєстрації. Деякі магазини примушують покупця проходити дуже довгий покроковий процес реєстрації. Це відлякує потенційних покупців

здійснювати покупки в даному магазині, особливо коли покупець цінує час, що витрачається ним.

Проблема віртуальності. Неможливість відчути і потримати товар, що купується (на картинці він може виглядати відмінно, але в реалі зовсім інакше).

Невиконання вибору замовлення. Можуть переплутати колір товару, модель, вигляд і так далі, або і в загалі замовлення може втратитися, але це буває дуже рідко.

1.4 Визначення технологій

Прості проекти це зазвичай сайти візитки, лендінги, прості інтернет-магазини, прості програми. Для таких типів проектів зазвича вже є готові рішення - CMS або шаблони. Середні проекти - складні інтернет-магазини і маркетплейси, портали національного масштабу, різноманітні сервіси. Зазвичай використовуються фреймворки. Складні - величезні портали, соціальні мережі, інноваційні та нетипові рішення. Ядро таких проектів зазвичай розробляється на чистій (нативній) мові програмування:

– швидкість розробки, тобто терміни, які відведені на розробку проекту;

– вартість, слід аналізувати бюджет відведений під даний проект;

– доступні інструменти розробки, вибираючи технології перш за все.

Вибираючи технологію за такими критеріями, можливо домогтися об'єктивного вибору і тим самим заощадити час і гроші.

Для розробки даного програмного забезпечення було обрано наступний стек технологій:

– vueJS (версія 2.5.2), це прогресивний фреймворк JS для створення користувацьких інтерфейсів;

– vue.cli – фреймворк що використовує шаблон MVVM для створення інтерфейсів користувача на основі моделей даних [4], через реактивне зв'язування даних;

– webpack (стилізований веб-пакет) – модуль з відкритим вихідним кодом JavaScript;

– spring framework – це програмний каркас (фреймворк) з відкритим кодом та контейнери з підтримкою інверсії управління для платформи Java;

– apache tomcat – контейнер сервлетів, розроблений Apache Software Foundation. Повністю написаний мовою програмування Java та реалізує специфікацію сервлетів;

– hibernate – засіб відображення між об'єктами та реляційними структурами (object-relational mapping, ORM) для платформи Java.

Фреймворки надають чітку структуру програми та реалізуються з використанням так званих «патернів проектування». Найбільш широко розповсюджені наступні патерни: MVC (Model-View-Controller), MVP (Model-View-Presenter) і MVVM (Model-View-ViewModel).

Серед усіх доступних фреймворків було обрано VueJS. VueJS – це прогресивний веб фреймворк JavaScript для створення користувацьких інтерфейсів. В відмінності від феймворків-монолітів, Vue створено придатним для поступового впровадження. Його ядро в першу чергу вирішує завдання рівня представлення (view), що спрощує інтеграцію з іншими бібліотеками та існуючими проектами. З іншого боку, Vue повністю підходить і для створення складних односторонніх додатків (SPA, Single-Page Applications), якщо використовувати його спільно з сучасними інструментами та додатковими бібліотеками.

1.5 Розгляд аналогів

1.5.1 Розгляд інтернет-магазину «Сайтег»

Оскільки на даний час я не знаю аналогічних інтернет-магазинів від університетів я порівняю свій інтернет-магазин комп'ютерної техніки з Рівненським інтернет-магазином «Сайтег» (рис. 1.1).

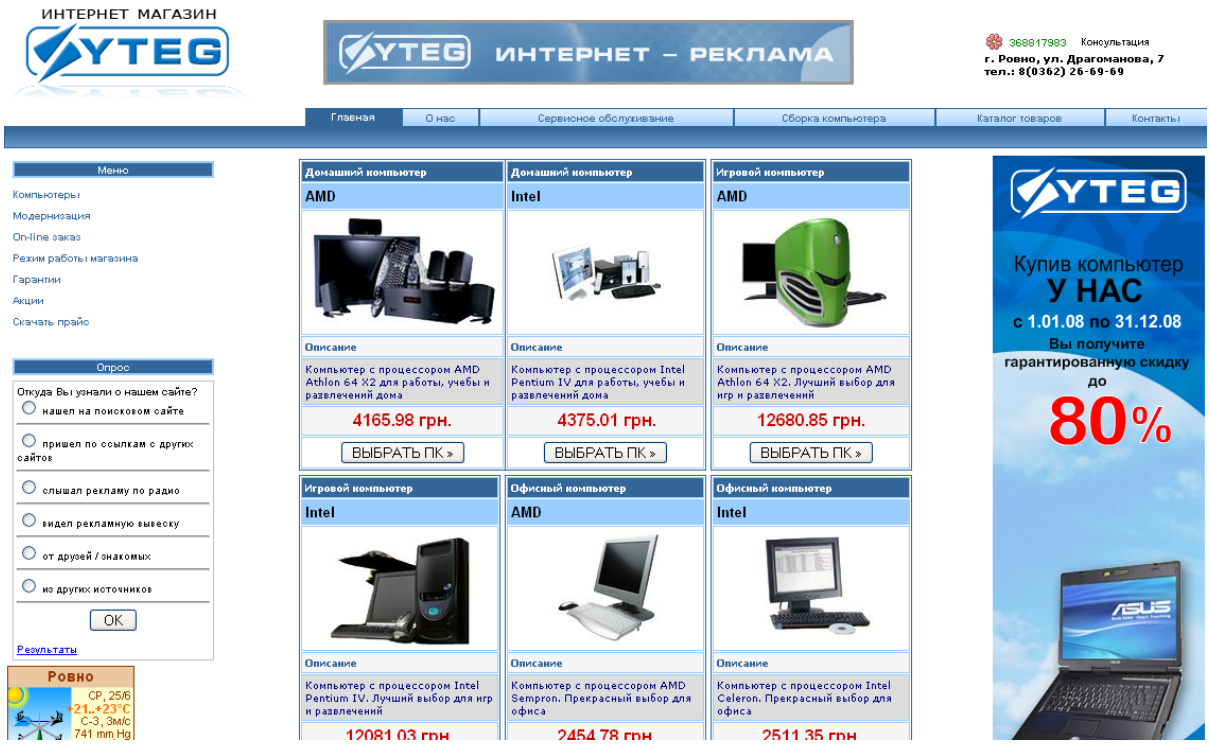

Рисунок 1.1 – Рівненський Інтернет-магазин «Сайтег»

ТОВ «Сайтег» – це зареєстрована організація, об'єднуюча в собі професіоналів своєї справи. Така організація існує на законних підставах і несе відповідальність за свою роботу за заздалегідь підготовленим договором. ТОВ «Сайтег» надає послуги з модернізації комп'ютерної техніки. Гарантує найвищу якість і максимально доступні ціни. Варіанти конфігурацій можна вибрати автоматично (знаходяться під головним горизонтальним меню) або власноручно скласти конфігурацію. Замовлення доставляються в перебігу декількох робочих днів після підтвердження. У описі кожної конкретної моделі товару вказаний термін гарантійного обслуговування, яке забезпечується магазином або ж безпосередньо сервісними центрами виробників. Працездатність товарів перевіряється співробітниками. ТОВ «Сайтег» займається оптовим і роздрібним продажем комп'ютерів, що їх комплектують. Компанія успішно працює у різних

напрямах, і це дозволяє ставити найнижчі ціни на продукцію світових брендів. ТОВ «Сайтег» пропонує широкий спектр послуг для споживачів комп'ютерного устаткування і офісної техніки. Нижче перерахований список послуг з сервісного обслуговування, які надає ТОВ «Сайтег»:

– гарантійне і післягарантійне обслуговування, ремонт комп'ютерів всіх модифікацій;

– ремонт відеоадаптерів і материнських плат;

– ремонт мобільних комп'ютерів notebook всіх модифікацій;

– проектування, прокладка і модернізація локальних мереж;

– ремонт моніторів всіх видів; обслуговування і ремонт всіх видів периферійних пристроїв - принтерів, сканерів, джойстиків, мультимедійних колонок і т.д;

– гарантійний і післягарантійний ремонт модемів;

– установка і настройка МІНІ-АТС;

– ремонт джерел безперебійного живлення;

– ремонт і обслуговування копіювальних апаратів;

– ремонт акустики, DVD апаратури;

– заправка копіювальних апаратів;

– заправка і регенерація картриджів;

– консультації по будь-яких спеціалізованих питаннях.

1.5.2 Розгляд інтернет-магазину «Розетка»

Рисунок 1.2 – Головна сторінка Інтернет-магазину Rozetka.com.ua

Проаналізуємо діалогове вікно Інтернет-магазину, який дозволяє придбати різні сучасні товари. Весь асортимент, пропонованих товарів, представлений в каталозі. У ньому міститься повна, достовірна і доступна інформація, що характеризує пропонований товар, а саме: фірмове найменування, зображення, опис товару, ціна товару та інше представлено на рисунку 1.2.

Рисунок 1.3 – Комп'ютерна техніка з магазину Rozetka.com.ua

На даному рисунку зображено діалогове вікно Головної сторінка Інтернет-магазину Rozetka. На ній розміщується пошукова панель по сайту, курс долара, розділи різних товарів від найбільш відвідуваних до найменш відвідуваних, акції

які можуть зацікавити покупця. Також коли проходить велике обновлення товару у певному розділі над ним з'являється напис «Нові товари» (рис 1.3).

Обравши найбільш відвідуваний розділ «Комп'ютери і ноутбуки», перед нами з'являються підрозділи різних ноутбуків, комп'ютерів, комплектуючих, програмного забезпечення, організаційної техніки. Можна побачити що навпроти певних підрозділів є надпис «NEW» – це означає що в них з'явилися нові товари, і покупець може їх переглянути.

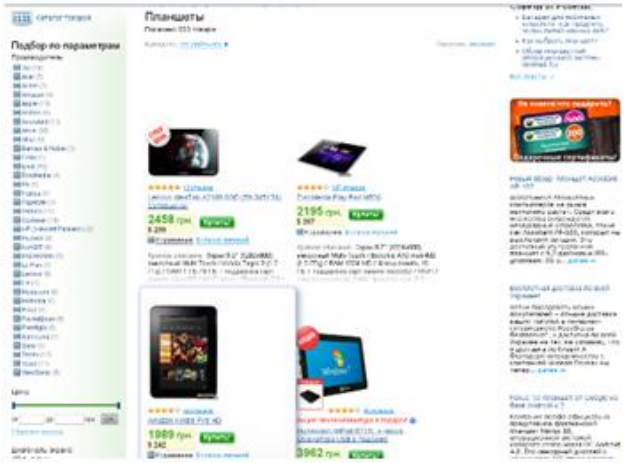

Рисунок 1.4 – Каталог товарів що є в наявності

Перейшовши до певного підрозділу, покупець має змогу вибрати певного виробника, ціну яка його влаштовує – це головне до кожного підрозділу (рис 1.4)

Рисунок 1.5 – Огляд товару

Вибравши певний товар, покупець має змогу прочитати його характеристику, подивитися детально його на фото, дізнатися про способи доставки. Також ознайомившись з написом «Аукціоне пропозиція» покупець має змогу купити зі скидкою потрібні йому аксесуари до товару за вигідною ціною (рис. 1.5).

Після того як покупець натискає на кнопку «Купити» у нього з'являється вікно «Корзина» у ній він має змогу побачити який товар обрано, кількість товару і суму яку має оплатити за товар.

Рисунок 1.6 – Огляд товару перед купівлею

Купуючи товар як подарок другу, подрузі можна поставити галочку «Це для подарунку», тоді до покупця зателефонують за номером який вказаний в оформлені замовлення і його не будуть турбувати, щоб він написав відгук про товар представлено на рисунку 1.6.

Придбання товару можна продовжити або оформити замовлення.

2 РОЗРОБКА АЛГОРИТМІВ ТА ВЗАЄМОЗВ'ЯЗКІВ МІЖ ФУНКЦІОНАЛЬНИМИ ОБ'ЄКТАМИ

2.1 MySQL. Основні можливості MySQL

База даних слугує основою для створення і розвитку всіх необхідних функції веб-додатку. Також важливо розробити базу даних, яка буде захищена від можливих атак, збережені дані повинні бути добре зашифровані.

Для проекту я використав СУБД MySQL, за допомогою якої здійснювалось зберігання та упорядкування даних.

MySQL – вільна система управління базами даних (СУБД). MySQL є власністю компанії Sun Microsystems, здійснюючу розробку й підтримку додатка. Поширюється під GNU General Public License і під власною комерційною ліцензією, на вибір. Крім цього компанія MySQL AB розробляє функціональність за замовленням ліцензійних користувачів, саме завдяки такому замовленню майже в самих ранніх версіях з'явився механізм реплікацій.

Основні можливості MySQL

Нижче представлений список найбільш важливих характеристик програмного забезпечення баз даних MySQL.

– внутрішні характеристики;

– написаний на мовах C и C++;

– протестований на широкому спектрі різних компіляторів;

– працює на безлічі різних платформ;

– доступні API-Інтерфейси для З, C++, Eiffel, Java, Perl, PHP, Python, Ruby і Tel;

– повністю багато поточний з використанням потоків ядра. Може працювати в багатопроцесорних системах;

– забезпечує транзакцію і не транзакцію механізми зберігання;

– використовує дуже швидкі дискові таблиці (MylSAM) зі стиском індексів на основі бінарних дерев (B-Дерев);

– порівняно просте додавання іншого механізму зберігання. Це зручно, якщо потрібно додати SQL-Інтерфейс до бази даних власної розробки;

– дуже швидка система розподілу пам'яті, заснована на потоках;

– дуже швидкі з'єднання, що використовують оптимізовані однопрохідні мульти-з’єднання;

– збережені в пам'яті хеш-таблиці, які використовуються як тимчасові таблиці;

– функції SQL реалізовані з використанням високо оптимізованої бібліотеки класів і повинні виконуватися гранично швидко. Як правило, будь-якого розподілу пам'яті після ініціалізації запиту не виконується;

– код MySQL протестований за допомогою інструментів пошуку витоку пам'яті, як комерційних, так і з відкритим вихідним кодом;

– сервер доступний як окрема програма для використання в клієнт-серверній мережному середовищу. Крім того, він також поставляється у вигляді бібліотеки, що може бути убудована в окремі автономні додатки. Такі додатки можуть застосовуватися в ізольованому середовищі або середовищі, що не має доступ до мережі;

– типи стовпців;

– безліч типів даних для стовпців таблиць: знакові/беззнакові цілі довжиною в 1, 2, 3, 4 і 8 байт; типи FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM і просторові типи OpenGIS;

– запису фіксованої й змінної довжини;

– оператори й функції;

– повна підтримка операцій і функцій у конструкціях SELECT і WHERE запитів, наприклад: mysql> SELECT CONCAT(first_name, ' ', last_name) -> FROM citizen -> WHERE income/dependents > 10000 AND age > 30;

– повна підтримка конструкцій GROUP BY і ORDER BY. Підтримка групових функцій (COUNT (), COUNT (DISTINCT . . .), AVG(), STD(), SUM(), MAX(), MIN() I GROUP_CONCAT());

– підтримка LEFT OUTER JOIN і RIGHT OUTER JOIN як із синтаксисом SQL, так і із синтаксисом ODBC;

– підтримка псевдонімів для таблиць і стовпців, як того вимагає стандарт SQL;

– оператори DELETE, insert, REPLACE і UPDATE повертають кількість рядків, які були змінені. Замість цього можна задати повернення кількості рядків, що відповідають запиту, для чого буде потрібно встановити відповідний прапор при підключенні до сервера;

– специфічна для MySQL команда SHOW може бути використана для добування інформації про бази даних, таблицях і індексах. Команда EXPLAIN дозволяє переглянути, як оптимізатор виконує запит;

– імена функцій не конфліктують з іменами таблиць і стовпців. Наприклад, ABS — абсолютно коректне ім'я стовпця. Єдине обмеження, що накладається на виклик функцій, - це те, що між ім'ям функції й наступної за ним відкриваючою дужкою не повинне бути пробілів;

– можна змішувати таблиці з різних баз даних в одному запиті (як в MySQL 3.22);

– безпека;

– система, заснована на паролях і привілеях, є винятково гнучкою і безпечною й дозволяє організувати верифікацію засобами хоста. Паролі захищені, оскільки весь трафік паролів під час з'єднання із сервером шифрується;

– масштабованість і обмеження;

– підтримує роботу баз даних величезних обсягів. Наприклад, компанія MySQL AB застосовує сервер MySQL для обслуговування бази даних, утримуючих 50 мільйонів записів. Відома також організація, що використовує

сервер MySQL для обслуговування бази даних з 60 000 таблиць, що зберігає близько 5 мільярдів записів;

– дозволяється мати до 64 індексів на таблицю (у версіях, що передують MySQL 4.1.2, допускалося до 32 індексів). Кожний індекс може містити від 1 до 16 стовпців або частин стовпців. Максимальна ширина індексу становить 1000 байт (500 байт у версіях, що передують MySQL 4.1.2). Для індексу може застосовуватися префікс стовпців з типами CHAR, VARCHAR, BLOB і TEXT;

– мережна зв'язність;

– клієнти можуть підключатися до сервера MySQL, використовуючи сокети TCP/IP на будь-якій платформі. В Windows-Системах сімейства NT (NT, 2000 або XP) клієнти можуть підключатися з використанням іменованих каналів. У системах на базі UNIX клієнти можуть підключатися через файли сокетів UNIX-Доменів;

– інтерфейс Connector/ODBC дозволяє MySQL підтримувати клієнтські програми, які використовують ODBC-З'єднання. Наприклад, для підключення до сервера MySQL можна використовувати MS Access. Клієнтське програмне забезпечення може виконуватися під керуванням Windows або UNIX. Вихідні тексти інтерфейсу Connector/ODBC доступні. Підтримуються всі функції ODBC 2.5, так само як і безліч інших;

– інтерфейс Connector/JDBC дозволяє MySQL взаємодіяти із клієнтськими програмами на Java, у яких використовуються JDBC-Підключення. Клієнтське програмне забезпечення може виконуватися під керуванням Windows або UNIX. Вихідні тексти інтерфейсу Connector/JDBC доступні;

– локалізація;

– сервер може видавати клієнтам повідомлення про помилки на різних мовах;

– повністю підтримуються кілька кодових таблиць, включаючи latinl (ISO-8859- 1), german, big5, ujis і інші. Наприклад, в іменах таблиць і стовпців дозволяється застосовувати скандинавські символи на зразок «а», «а» і «б». Починаючи з версії MySQL 4.1, також забезпечується підтримка Unicode;

– всі дані зберігаються в обраному кодуванні. Всі порівняння стовпців з нормальними рядками чутливі до регістра;

–сортування виконується відповідно до обраного кодування (за замовчуванням використовуються шведський набір). Під час запуску сервера MySQL це можна змінити. Як приклад досить зробленого сортування рекомендується звернути увагу на код сортування для чеської мови. Сервер MySQL підтримує безліч різних кодувань, причому вони можуть бути зазначені як під час компіляції, так і під час виконання;

– клієнти й інструменти;

– сервер MySQL має убудовану підтримку SQL-Операторів для перевірки, оптимізації й відновлення таблиць. Ці оператори можна виконувати в режимі командного рядка, використовуючи клієнтський додаток mysqlcheck. MySQL включає також myisamchk - дуже швидку утиліту командного рядка для реалізації тих же операцій над таблицями MylSAM;

– всі програми MySQL можна запускати на виконання з опцією -help або -? для одержання швидкої підказки.

2.2 Опис структури бази даних

В дипломному проекті MySQL використовувалась в парі з Hibernate ORM, яка значно пришвидшує процес створення програми, так як дозволяє швидко створити всі таблиці бази даних майже не використовуючи будь-які можливості середовища проектування баз даних MySQL. Це зменшило час розробки структури бази даних в рази, а також дозволило набагато краще зпроектувати БД, за рахунок продумування зв’язків між об’єктами напряму в коді, а все інше виконує ORM.

ORM (Object-Relational Mapping) – технологія програмування, яка зв'язує базу даних з концепціями об'єктно-орієнтованих мов програмування, в даному випадку мовою програмування є Java, створюючи віртуальну об'єктну базу

даних. Схема на рис. 2.1 фактично результат роботи даної системи керування зв'язками між об'ктами, так як всі параметри, кількість символів, та все інше було прописано напряму в коді. Також всі зв'язки між таблицями, унікальні значення та автоінкрементація ідентифікаторів у таблицях теж генеруються за допомогою ORM у коді сервера.

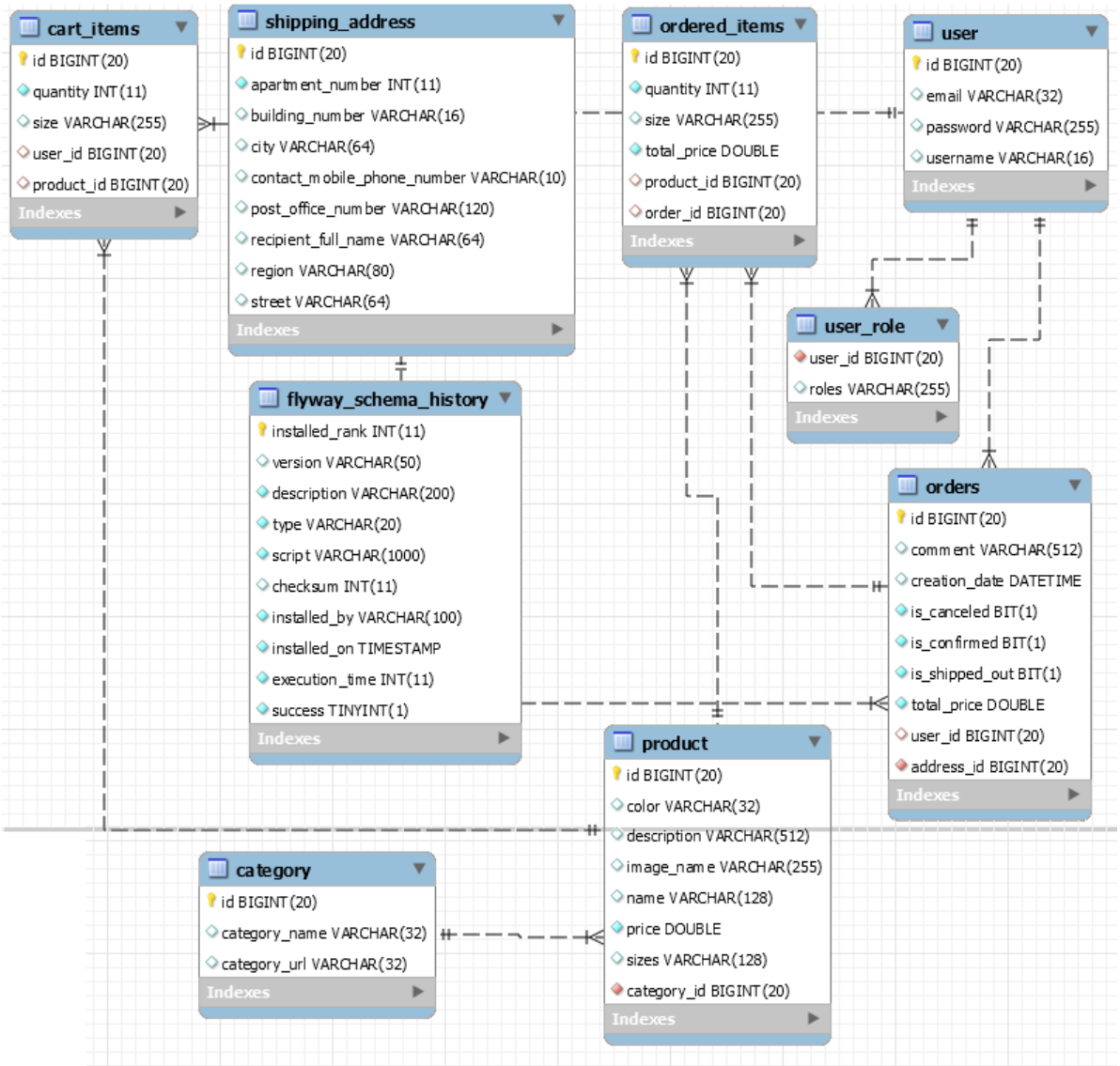

Рисунок 2.1 – ERD діаграма бази даних

Одною з найважливіших таблиць у проекті, є таблиця «user», в якій зберігаються основні дані, такі як: логін, пароль (який зберігається у зашифрованому вигляді), ел. адреса, та ім'я представлено на рисунку 2.2.

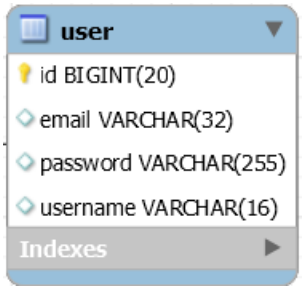

Рисунок 2.2 – ERD діаграма таблиці user

В скрипті описане створення таблиці user з полями (рис 2.2):

– id – ідентифікатор, по якому буде здіснюватись вибірка полів з таблиці;

– email – електронна адреса користувача;

– password – пароль;

– username – логін користувача, який використовується для входу на сайт; Максимальна дожна колинки де міститься пароль встановлено на 512 символів, оскільки паролі збирігаються у шифрованому вигляді.

Опис колонок таблиці shipping_address представлено на рис. 2.3.

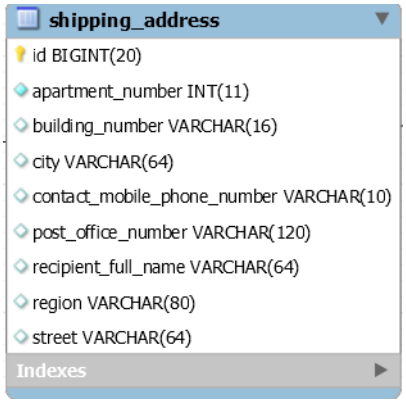

Рисунок 2.3 – ERD діаграма таблиці shipping_address

В даній таблиці зберігаються дані покупця, які він вносив при реєстрації в своєму особистому кабінеті. Дані покупця автоматично запам’ятовуюсь і при повторному замовленні товару користувачу, вже не обов'язково їх вводити повторно, оскільки ці дані вже є збереженні в БД (рис 2.3).

– id – id користувача;

– appatmente_number - номер квартири користувача;

– building_number – номер будинку користувача;

– city – місто доставки;

– contact_mobile_phone_numbet – контактний номер мобільного, за допомогою якого буде прадавець здійснювати пітвердженняамовлення;

– post_office_number – номер поштового віділення, для того щоб легше можна орінтуватися куди потрібно доставити замовдення;

– recipient_full_name – П. І. Б. отримувача;

– region – область;

– street – вулиця на яку буде доставлятися замовлення;

Опис колонок таблиці product представлено на рис 2.4:

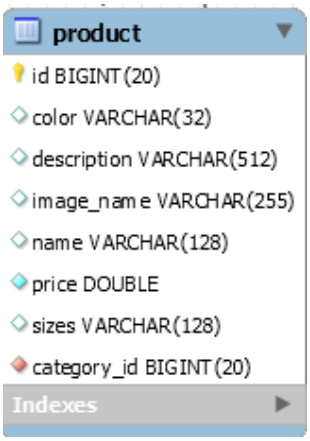

Рисунок 2.4 – ERD діаграма таблиці product

В даній таблиці описуєть товар який буде продаватися, його основні параметри, та його загольний опис (рис 2.4).

– id – індифікатор товару;

– color – колір товару або наявність для продажі кольорової гами;

– description – опис товару, де можна більш детальніше описати товар, розмір і т. д. але не є обов’язковим;

– image_name – фото (картинка) представленого товару розташовоного у каталозі на диску;

– name – назва товару;

– price – ціна на одну одиниць товару, вказана в національні валюті;

– sizes – розміри даного товару;

– category_id – ідентифікатор каталогу, до якого відноситься замовлений товар;

Опис колонок таблиці orders представлено на рис 2.5:

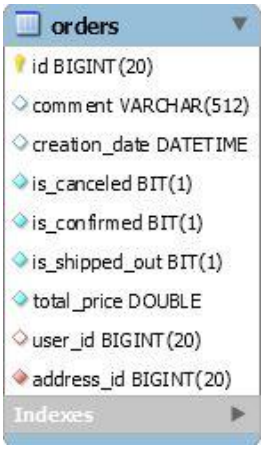

Рисунок 2.5 – ERD діаграма таблиці orders

– id – ідентифікатор замовлення;

– comment – коментар, який може залишити користувач при замовленні товару;

– creation_date – дата створення замовлення;

– is_cancel – вказує статус замовлення «Скасовано»;

– is_confirmed – вказує статус замовлення «Підтверджено»;

– is_shipped_out – вказує статус замовлення «Відправлено»;

– total_price –вартість товару за 1 одинцю товару.

– user_id – ідентифікатор користувача який оформив замовлення;

– addres_id – ідентифікатор адреси.

На рисунку 2.6 подано ERD діаграма таблиці ordered_items

Рисунок 2.6 – ERD діаграма таблиці ordered_items

Опис колонок таблиці ordered_items (рис 2.6):

– id – ідентифікатор товару у замовленні;

– quantity – кількість обраного товару;

– size – розмір або варіації розмірів товару;

– total_price – ціна за вибрану кількість товару;;

– product_id – ідентифікатор замовленого товару;

– order_id - ідентифікатор замовлення, до якого відноситься замовлений товар;

3 РОЗРОБКА ОСНОВНИХ КОМПОНЕНТІВ, МЕТОДІВ ТА ОГЛЯД ІНТЕРФЕЙСУ АДМІНІСТРАТОРА ТА КОРИСТУВАЧА

3.1 Опис інтерфейсу інтернет -магазину

Створення власного Інтернет - магазину зв'язане з масою проблем:

По-перше, необхідно розробити даний проект.

По-друге, необхідно його просунути в пошукових системах, що займає, не мало часу, але принципово важливо, адже потрібно, щоб ви скрізь були на перших сторінках, інакше потенційні споживачі просто не зуміють вас знайти.

Крім того, варто враховувати, що тут, так само як і в реальному житті існує вельми солідна конкуренція, тобто необхідно детально продумати стратегію своєї роботи, щоб отримати перемогу в цій конкурентній боротьбі.

У наше століття швидко розвиваючих комп'ютерних технологій, комп'ютери стали незамінними атрибутами практично в кожному будинку, і вже звичайно в офісах, а тому даний вид бізнесу просто від початку приречений на успіх, але зусилля прикласти у будь-якому випадку доведеться. Крім того, постійно з'являються в даній області нововведення, що комплектуючі комп'ютерів оновлюються і удосконалюються, таким чином, ті люди, які своєчасно зорієнтувалися в цій ситуації, і відкрили власний бізнес, пов'язаний з торгівлею даним видом товарів, не помилилися. Але завжди потрібно прагнути до чогось більшого, до підкорення нових висот, саме таким шансом в даній конкретній ситуації і є створення інтернет-магазину. Він відкриває величезні перспективи для підприємців всіх сфер діяльності і дає шанс розширення бізнесу і власного кар'єрного зростання.

Кожен алгоритм є списком добре визначених інструкцій для розв'язання задачі. Починаючи з початкового стану, інструкції алгоритму описують процес обчислення, які відбуваються через послідовність станів, які, зрештою, завершуються кінцевим станом. Кожен алгоритм передбачає існування

початкових (вхідних) даних та в результаті роботи призводить до отримання певного результату. Робота кожного алгоритму відбувається шляхом виконання послідовності деяких елементарних дій. Ці дії називають кроками, а процес їхнього виконання називають алгоритмічним процесом

Важливою властивістю алгоритмів є масовість, або можливість застосування до різних вхідних даних. Тобто, кожен алгоритм покликаний розв'язувати клас однотипних задач.

Виконання команд алгоритму відбувається у єдиний спосіб та призводить до однакового результату для однакових вхідних даних. Вхідні дані алгоритму можуть бути обмежені набором припустимих вхідних даних. Застосування алгоритму до неприпустимих вхідних даних може призводити до того, що алгоритм ніколи не зупиниться, або потрапить в тупиковий стан (зависання) з якого не зможе продовжити виконання.

Для розробки данного сайту використаємо такий алгоритм:

–етап 1 – визначення цілей створення сайту. Це найважливіший етап в створенні сайту, тому що не можна досягти мети, якщо її немає або якщо вона визначена неправильно. Від цілей буде залежати весь подальший процес створення сайту, кожен його етап. Тому до постановки мети потрібно підходити з максимальною серйозністю. Правильно поставлена мета - це вже половина успіху.

–етап 2 – визначення типу сайту, розробка ТЗ і структури. На цьому етапі складається ТЗ (технічне завдання) на створення сайту. ТЗ, як мінімум, повинне включати наступні пункти: Тип сайту, функціонал сайту (пошук, каталог, стрічка новин і т.д.). Стиль дизайну (строгий, веселий, похмурий і т.д.). Структура сайту (які сторінки мають бути на сайті). Структура сторінок (які блоки мають бути на сторінках і в якому вигляді). При складанні ТЗ обов'язково потрібно пам'ятати про цілі створення сайту. Якщо яка-небудь функція сайту не сприяє досягненню поставлених цілей (або тим більше - заважає) – необхідно від неї відмовитися: на сайті вона буде зайвою. Наприклад, якщо мета сайту – продати якомога більше одиниць товару, то стрічка новин на сайті не потрібна.

–етап 3 – розробка макету дизайну сайту. Цей етап ділиться на кілька підетапів: Генерація ідей дизайну. Розробка попереднього макета дизайну головної сторінки. Розробка внутрішніх сторінок. Природно, кожен крок розробки макету дизайну повинен виконуватися згідно з ТЗ і цілями сайту.

–етап 4 – HTML-CSS верстка. В залежності від цілей і завдань сайту, верстка повинна задовольняти деяким вимогам. Зазвичай ці вимоги такі: Кросбраузерність - сторінки повинні однаково відображатися різних браузерах (Mozilla Firefox, Google Chrome, Opera, Internet Explorer, Safari і т.д.). Гнучкість верстки – можливість легко додавати / видаляти інформацію на сторінки. Швидкість обробки коду браузером. Валідність – відповідність стандартам. Семантична коректність - логічне і правильне використання елементів HTML

–етап 5 – програмування. Це чисто технічний етап, на якому реалізується весь функціонал сайту. Вимоги до цього етапу визначаються технічним завданням.

–етап 6 – заповнення сайту контентом (інформацією). Всі матеріали сайту, будь то тексти чи графіка, також повинні вписуватися в загальну концепцію сайту, відповідати його цілям і завданням.

Спробуємо визначити основні елементи, які нам необхідні для створення Інтернет-магазину:

–розробка функціонуючої структури Інтернет-магазину (визначення основних принципів роботи);

– розробка структури сайту і схем сторінок;

– створення дизайн-макета;

– підготовку прав на різні типи операцій на сайті для груп користувачів;

– створення необхідної кількості розділів і інформаційних блоків;

– розробку структури товарного каталога;

– настройку експорту і імпорту даних;

– узгодження необхідних характеристик товарів для порівняння і виводу;

– настройка форм замовлення і зворотного зв'язку;

– обробка зображень, наповнення сайту.;

Велику увагу при створенні Інтернет-магазину потрібно виділяти розробці інтерфейсу. Інтерфейс - в широкому сенсі - визначена стандартами межа між взаємодіючими незалежними об'єктами. Інтерфейс задає параметри, процедури і характеристики взаємодії об'єктів.

Інтерфейс користувача - елементи і компоненти програми, які здатні робити вплив на взаємодію користувача з програмним забезпеченням. Зокрема:

- засоби відображення інформації, інформація, що відображається, формати і коди;
- командні режими, мова користувач-інтерфейс;
- пристрої і технології введення даних;
- діалоги, взаємодія і транзакції між користувачем і комп'ютером;
- зворотний зв'язок з користувачем;
- підтримка ухвалення рішень в конкретній наочній області;
- порядок використання програми і документація на неї.

Є декілька простих правил, що дозволяють інтерфейсу магазину бути зрозумілим клієнтові.

Чим простіше, тим краще. Це зовсім не означає, що повинен бути тільки текст і контактна інформація, але на сайті не повинно бути даремної інформації, шрифт повинен легко читатися. Графічні елементи повинні бути були чіткими, виразними і швидко завантажуватися. На сайті магазину неприпустимо використовувати анімацію і звук, які довго завантажуються і відволікають увагу покупця.

Людське око проглядає сторінки сайту зверху вниз. Найбільша увага зосереджується на верхній лівій частині сторінки. Тому у верхній частині сторінки сайту, як правило, розміщується найбільш важлива інформація: назва фірми, логотип, сама назва сайту і так далі. Чим простіше виконана верхня частина сторінки, тим легко запам'ятати назву сайту і саму фірму.

Типовою помилкою багатьох сайтів є нагромадження у верхній частині сторінки сайту авангардної і складної композиції з багатьох малюнків, серед яких ледве помітно розташовується назви фірми і назву сайту. В результаті

людське око, скануючи дану частину сторінки сайту, не встигає розпізнати і запам'ятати потрібну інформацію. Відбувається перенасичення. Замість збільшення запам’ятованості, на таких сайтах, відбувається зниження запам’ятованості основних написів (малюнків) сайту, назви фірми і назва сайту.

Легкість пошуку потрібного товару. Покупець повинен без зусиль знаходити товар, що цікавить його, і мати можливість отримати про нього вичерпну інформацію (опис у вигляді тексту плюс декілька фотографій). На вітрині бажано розмістити дані про діяльність фірми-продавця.

Розподіл товарів по групах. Товари повинні бути розподілені по групах. Необхідно забезпечити можливість пошуку товарів по частині назви і опису.

Якісне оформлення замовлення. При оформленні замовлення покупець повинен ввести контактну інформацію, після перевірки даних клієнт повен отримати підтвердження про отримане замовлення.

Оплата. Розрахунок вартості і виведення цін повинні здійснюватися в національній валюті.

У електронному магазині можуть бути і інформаційні розділи:

- з даними про магазин (сфера діяльності, адреса, контактні телефони і так далі);

- новини магазину;

- статті (системи управління новинами і статтями надають можливість використовувати Інтернет-магазин як справжній інформаційний портал);

- інша корисна інформація.

Збільшенню відвідуваності інтернет-магазина і кількості постійних клієнтів сприяє також наявність зворотного зв'язку адміністратора з клієнтами. Важливо, щоб адміністратори щонайшвидше і максимально повно відповідали на листи.

Зворотною, невидимою покупцеві, стороною інтернет-магазину є система управління. Вхід в систему адміністрування здійснюється тільки після введення адміністратором логіна і пароля. Адміністратор має можливість повністю управляти вмістом Інтернет-магазину:

- додавати або видаляти товари, описи і фотографії до них, змінювати їх вартість, умови доставки товарів і рівень знижок;

- редагувати розділи магазина;

- редагувати контактну інформацію Інтернет-магазину;

- редагувати зміст заголовків і текстів листів, що відправляються покупцеві при реєстрації і покупці товару;

- проглядати історію замовлень і статистику покупців та інше.

Сайт – сукупність веб-сторінок, доступних у мережі Інтернет, які об'єднані як за змістом, так і навігаційно. Фізично сайт може розміщуватися як на одному, так і на кількох серверах.

Грамотна розробка сайту – це складний багатоетапний процес. Верстання сайтів є невід'ємною частиною процесу виготовлення сайтів. Верстання сайту – це процес формування веб-сторінки по готовому макету. Цей процес містить в собі створення коду сторінки за допомогою зрозумілої браузерам мови розмітки гіпертексту html, і оформлення її за допомогою каскадних таблиць стилів (CSS). Не треба самому намагатися верстати сайт, доцільніше замовити послуги верстальника.

Адже професійне верстання сайтів повинне враховувати численні особливості, як правил верстання, так і параметрів відображення документів в різних браузерах.

Верстання і дизайн ширини сторінки

Розробка сайту, щоб не зламати продуманий дизайн, повинна враховувати той факт, що користувачі мережі Інтернет мають монітори з різним дозволом. В цьому випадку існує два варіанти дизайн-макета сайту: сайт з фіксованою шириною і гумовий сайт. Ширина фіксованого сайту визначається з розрахунку того, що за статистичними даними більшість користувачів використовують монітори з дозволом 1024x768 пікселів.

Отже, для того, щоб уникнути появи горизонтальної смуги прокрутки і спотворення дизайну, треба в середньому брати ширину макету приблизно 800–990 пікселів. Недолік при такому методі верстання – це неефективне

використання вільної площі. Гумовий макет сайту виходить за рахунок того, що одиницями виміру при верстанні дизайну у більшості своїй виступають відсотки, що дозволяє розтягнути макет на усю ширину вікна браузеру. Верстання і дизайн гумового макету найбільш прийнятні. Верстання сайтів з урахуванням кросбраузерності.

Сьогодні існує цілий ряд популярних браузерів, використовуваних людьми для роботи в мережі Інтернет. До них відносяться: Opera, Mozilla Firefox, Google Chrome, Netscape Navigator, Safari, Internet Explorer та інші. Кожен з них має декілька версій, і кожен може відображати одну і ту ж сторінку, представлену у форматі html, по-різному. Усі творці браузерів прагнуть прийти до єдиного стандарту використання мови гіпертекстової розмітки, але доки існують розбіжності, тому верстання сайтів повинне враховувати кросбраузерність.

Грамотне верстання сайтів з чистим кодом

Ще одне правило хорошого тону при верстанні сайту – це чистий і красивий код. Що це означає? Це означає, що код html розмітки сторінки красиво відформатований, містить коментарі і виділені логічні блоки. Використання каскадних таблиць стилів (CSS) дозволить очистити код від зайвих тегів і зробити вагу сайту в цілому значно менше. Верстання сайтів з використанням усіх цих маніпуляцій потрібне для того, щоб спростити код, і зіткнутись з найменшою кількістю проблем при його виправленні, модифікації і доопрацюванні. Правильний компактний код полегшить індексацію сайту пошуковими системами, а мала вага скоротить час завантаження сайту.

Табличне і блокове верстання сайтів

Верстати сайт можливо двома принципово різними способами. Перший спосіб – за допомогою таблиць. Табличне верстання використовує елементи розмітки, що не відповідають тій семантиці, яку вони несуть з метою отримання зовнішніх ефектів, постільки, поскільки спочатку таблиці не були призначені для побудови каркаса сторінок, а служили для стандартного введення даних. Для того, щоб верстати сайт за допомогою табличного верстання, частенько виникає необхідність використання безлічі вкладених таблиць. Такий підхід нерідко

сильно утруднює обробку браузером сторінок, збільшує розмір документу знижує швидкість завантаження файлів, і може привести до появи помилок в коді. Зручніше і правильно, з точки зору переконань про розділення змісту і оформлення, верстати сайт за допомогою шарів (блоків). Блокове верстання сайтів дає більше можливостей, дозволяє зробити код компактніше, і збільшити швидкість завантаження веб-сторінки. Через CSS властивості шарів настроювати набагато зручніше. За допомогою верстання шарами можна набагато ефективніше розробити сайт, який коректно відображатиметься у браузерах.

Веб-юзабіліті (англ. web-usability – дослівно «зручність користування веб») – це застосування принципів юзабіліті в тих областях, де перегляд веб-сторінок можна розглядати як загальну парадигму (або «метафору») для побудови GUI (графічний інтерфейс користувача). Для зручності користування системою необхідно використовувати прості випадаючі меню для представлення змісту розділів. Об'єм сторінки не може перевищувати 500–900 Кб.

Верстка повинна бути кросбраузерна. Вміст веб-сторінок має однаково відображатися в популярних браузерах, таких як:

– google chrome;

– mozilla firefox;

– internet explorer версій не нижче 7;

– opera;

– safari;

– lunascape.

Структуру Web-сайту Інтернет-магазину представимо наступним чином:
Із структури видно представленого на рисунку 3.1 що в інтернет-магазині пропонується 2 робочих частини: адміністративна та клієнтська.

Адміністративна частина дозволяє адміністратору Інтернет-магазину (після входження в систему адміністрування) здійснювати редагування бази даних та внесення додаткових елементів (товарів) в базу даних. Клієнтська частина даної можливості немає, тобто доступ до ресурсів Інтернет-магазину закритий для загального огляду.

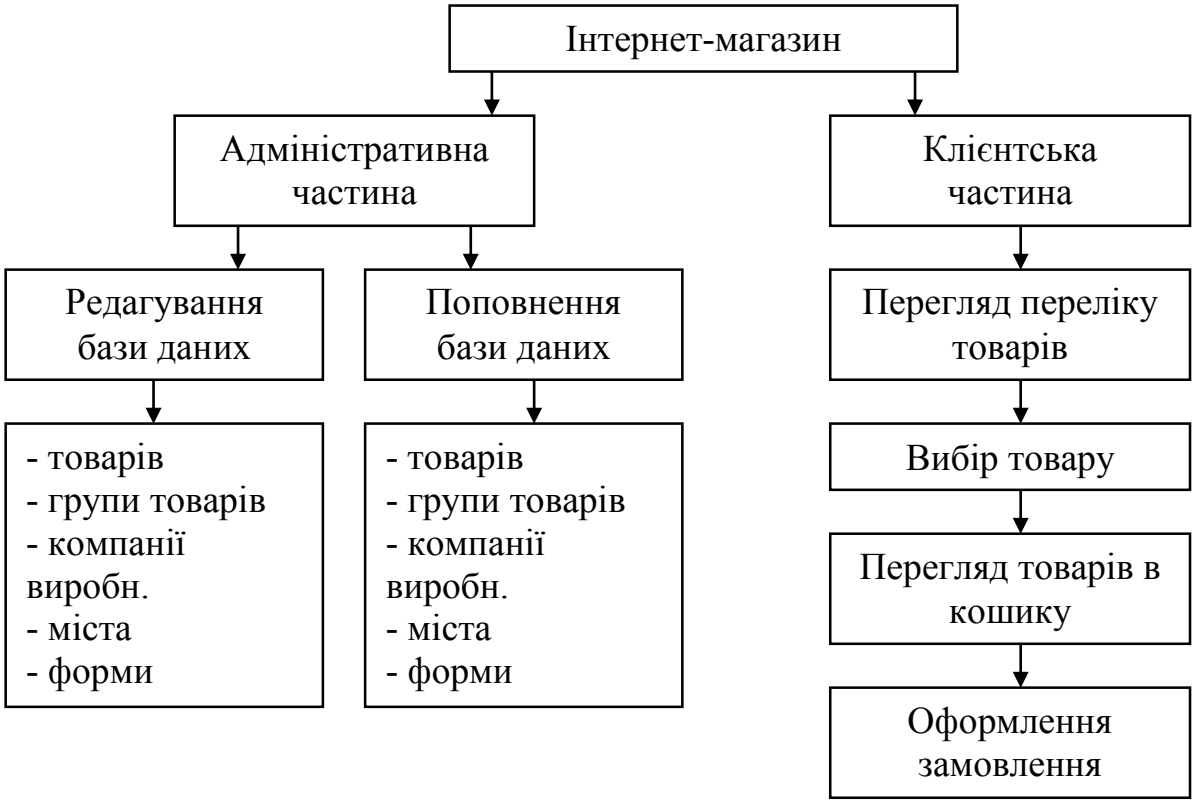

Рисунок 3.1 – Структура Web-сайту Інтернет-магазину

Клієнтська частина доступна всім користувачам і відвідувачам Інтернет-магазину. Дозволяє клієнтам здійснювати перегляд, пошук, вибір потрібного товару, який розміщений в різних групах. Це значно полегшує роботу користувача при пошуку певної інформації. Клієнт має змогу оформити замовлення товару, вибрати тип оплати та доставки товару.

3.1.1 Опис інтерфейсу для адміністратора та продавця

При завантаженні Web-сайту «Інтернет-магазину брендованої продукції Короля Данила» перед користувачем відкривається головна сторінка (рис.3.2):

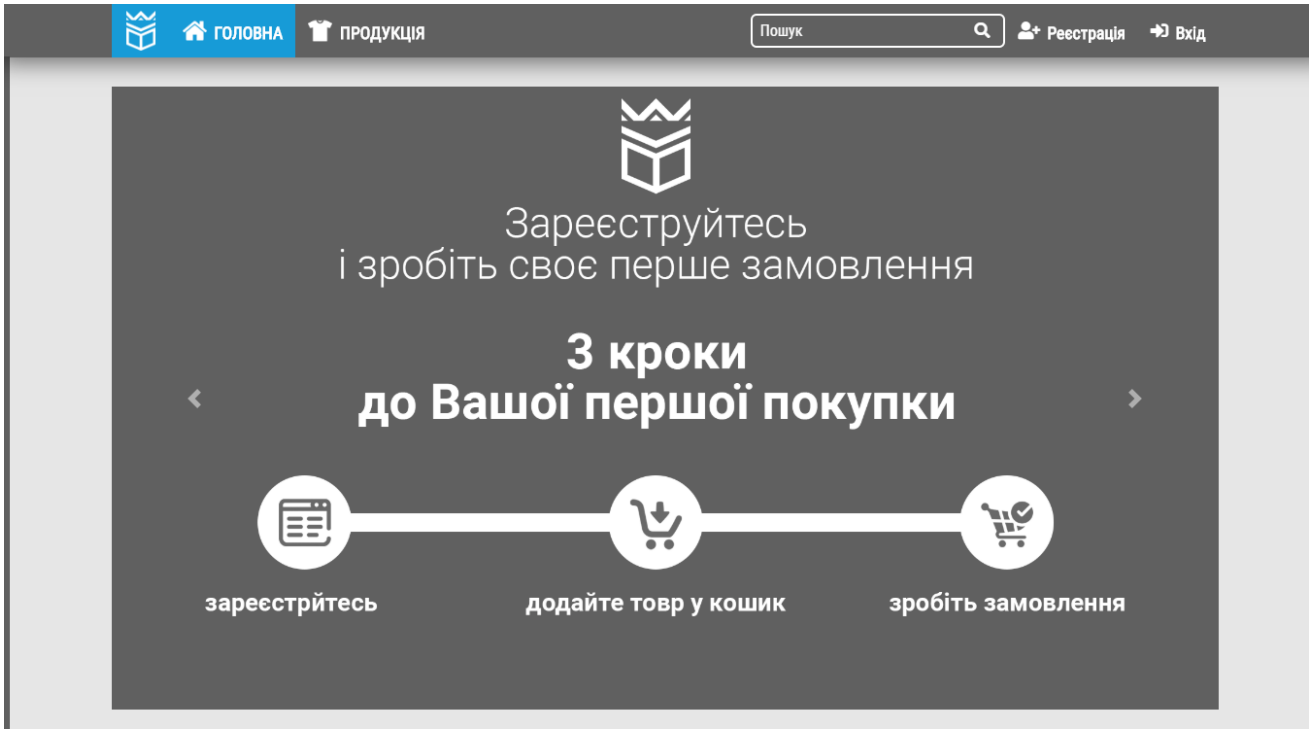

Рисунок 3.2 – Головна сторінка Web-сайту «Інтернет-магазину продукції Короля Данила»

На даній сторінці розміщені меню користувача (рис 3.3), категорії товарів (рис 3.4), швидкий пошук товару, реєстрація або вхід до свого профілю.

Розробка інтернет-магазину здійснюється шляхом переходу від інфологічної моделі до даталогічної, суть якої полягає в тому, що навідмінну від інфологічної, якою займається адміністратор вищого рівня – проектувальник, то на цьому етапі здійснюється вибір шляхів та методів проектування моделі за допомогою програмних засобів та сервісів.

Програмними засобами вступають мови програмування високого рівня, можливості об’єктно-орієнтованого програмування, зокрема ірархічний полеморфний та метод наслідуваня які визначатимуть структуру сайта.

Даний сервіс дає можливості працювати у двох режимах:

1. Адміністративний, який дозволяє створювати структуру сайт, редагувати товар, створювати новий. Також у режимі адміністратора ми можемо проектувати базу, проводити зміни, здійснювати оновлення, тощо.

2. Режим представлення, який призначений кінцевих користувачів тобто клієнтів, що дозволено даним сервісом створювати клієнт-серверну систему. За допомогою даного режиму клієнти формують запити і сервіс.

Рисунок 3.3 – Меню користувача

Проаналізуємо діалогове вікно головної сторінки інтернет-магазину. На ній розміщується пошукова панель по сайту, контактний номер телефону, ссилки на соціальні мережі, каталог товарів і детальна характеристика товару (рис. 3.2).

Вхід в адміністративну частину сайту здійснюється при натисканні пункту ВХІД меню користувача. При цьому адміністратору пропонується пройти перевірку особистих даних (логіна і пароля) (рис. 3.5).

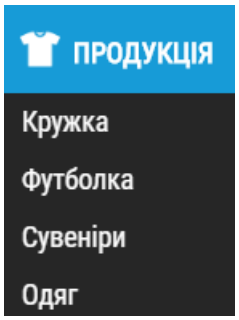

Рисунок 3.4 – Категорії товарів

При вірному введенні даних в поля логін і пароль відбувається входження в систему і перед адміністратором інтернет-магазину відкривається вікно з наступними можливостями:

– добавити товари і розподілити їх по категоріям;

– подивитися замовлення;

– заблокувати користувачів;

– подивитися відвідування Інтернет-магазину;

–надавання іншим користувачам адміністраторських прав;

–редагувати «категорії товарів» (рис 3.4);

–створювати нові категорії товарів (рис 3.4);

– перегляд вмісту товарів в базі даних (рис 3.6);

– додавання нового товару в базу даних (рис 3.7);

–редагування існуючого товару в базі даних – при виборі даної опції надається моєжливісь коригувати товари частково або повністю (рис 3.8);

У розділі «Користувачі» надає можливість продивитися зареєстрованих користувачів, переглянути як за часто вони відвідують інтернет-магазин, як за часто здійснюють покупки а також змінити їх реєстраційні дані, а також заблокувати їх.

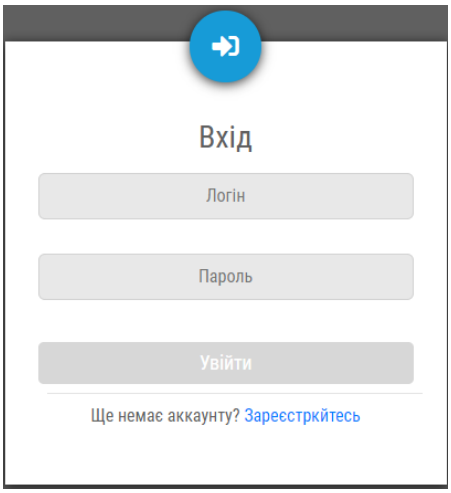

Рисунок 3.5 – Вхід в систему

Адміністратор має можливість додавати знижки до товару, причому ця знижка може бути багаторівневою, тобто залежить від розміру товару.

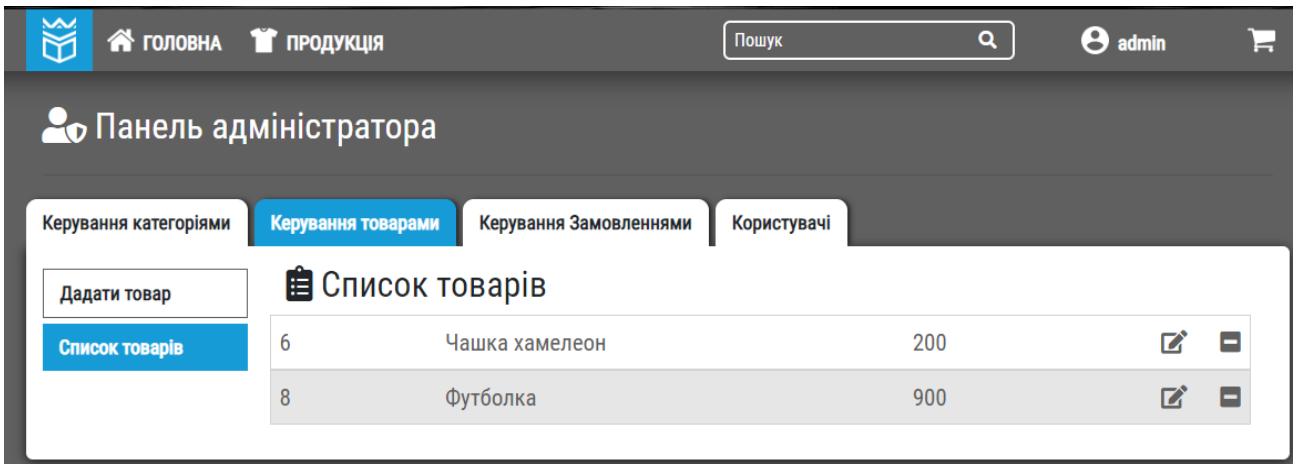

Рисунок 3.6 – Перегляд вмісту товарів в базі даних

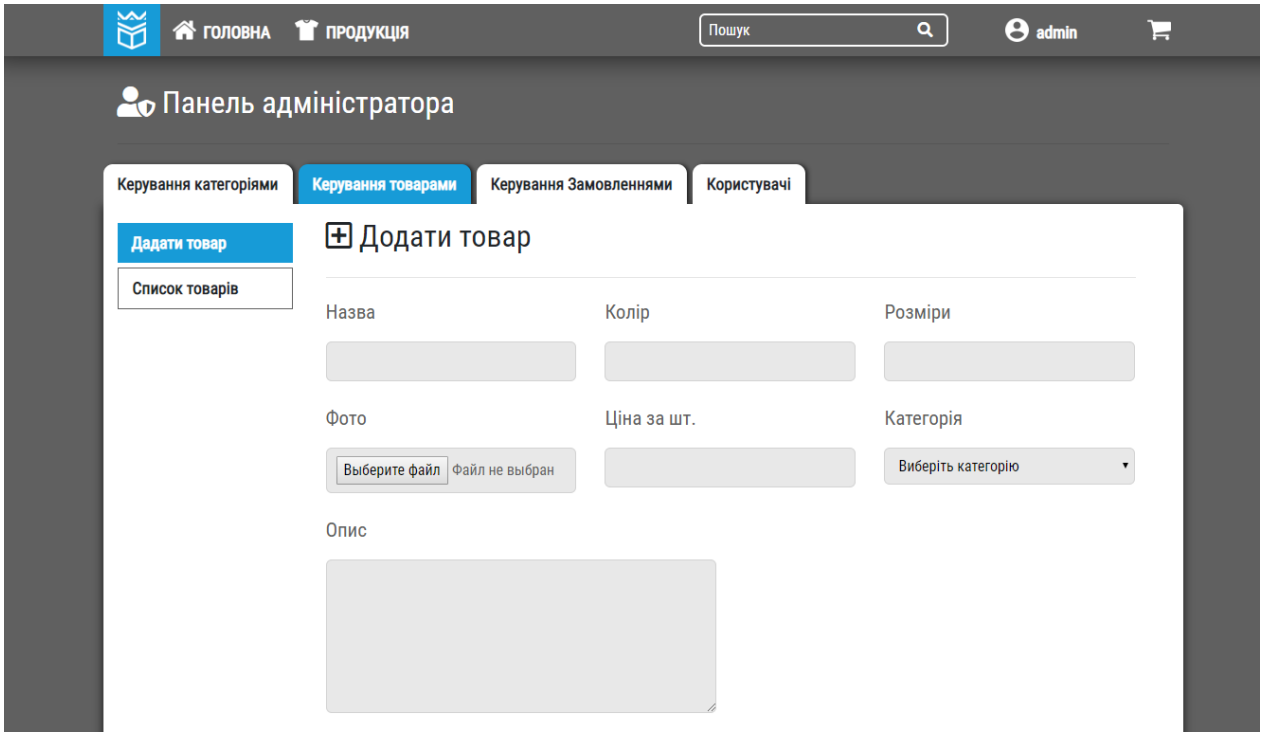

Рисунок 3.7 – Додавання нового товару в базу даних

Редагування або створення нового товару здійснюється адміністратором або користувачем який наділений цим повноваженням. Він має можливість редагувати всі пункти або взагалі видалити товар з бази даних.

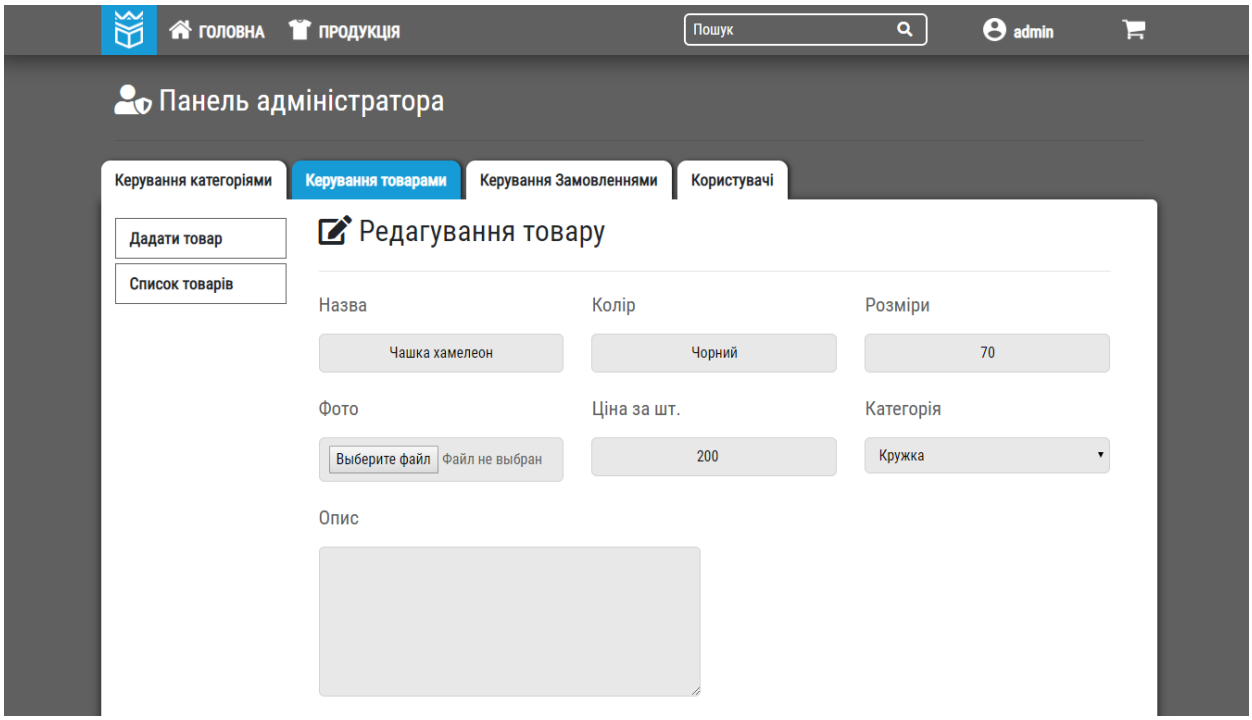

Рисунок 3.8 – Редагування існуючого товару в базі даних

Адміністратор або продавець чи користувач який має повноваження адміністратора при отриманні замовлення приходить повідомлення про надходження нового замовлення, розглядачи зомовлення мають можливість підтвердити або скасувати його замовлення, бачить всі товари які замовив покупець та адресу куди потрібно відправити товар (рис 3.9).

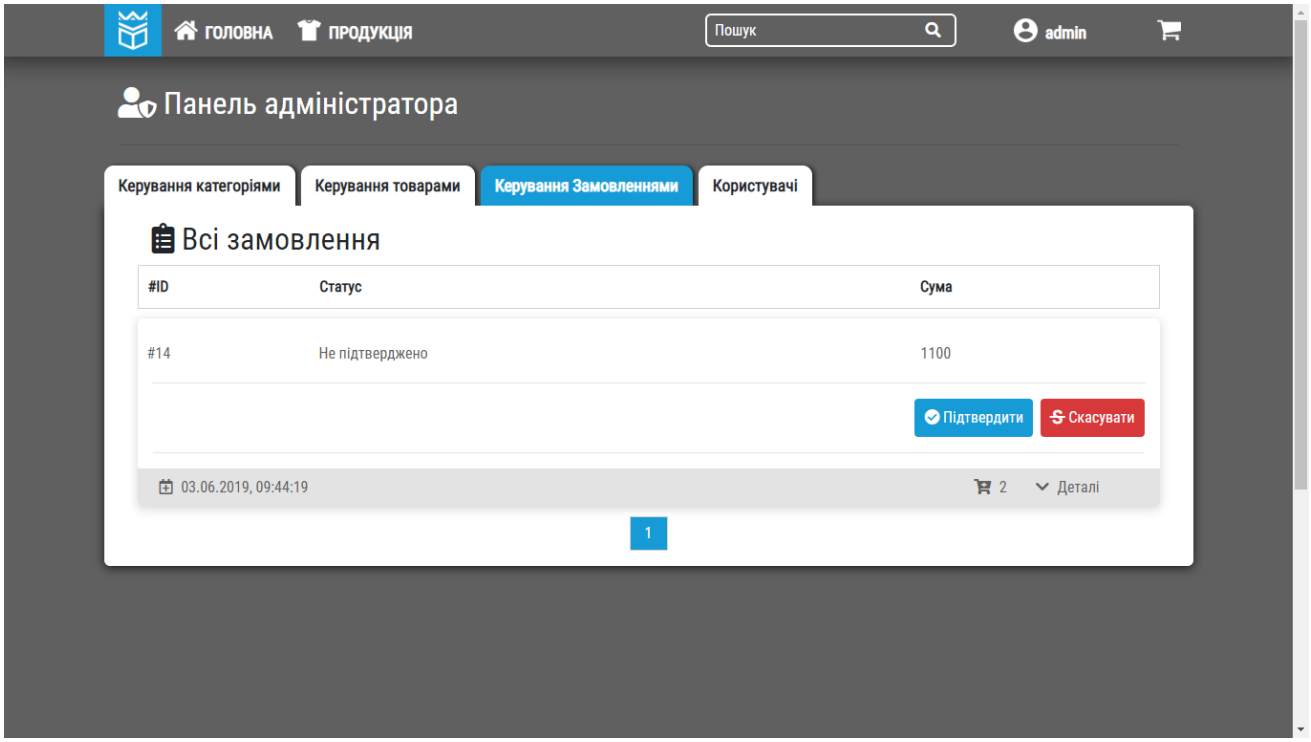

Рисунок 3.9 – Вікно про підтвердження або скасування замовлення

Для виходу з адміністративної частини в клієнтську на меню користувача необхідно натиснути кнопку ВИХІД (рис 3.10).

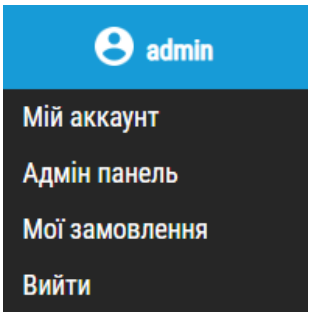

Рисунок 3.10 – Вихід з адміністративної частини

Вхід в клієнтську частину здійснюється автоматично при завантаженні інтернет-магазину. Користувач може здійснити пошук потрібного йому товару скориставшись КАТЕГОРІЯМИ, що значно пришвидшить його роботу. Вибравши потрібну категорію, перед користувачем відкривається вікно з переліком товарів та коротким описом про даний товар із вказанням його ціни (ціна всіх товарів вказана в національній валюті).

3.1.2 Опис інтерфейсу для користувача

Інтерфейс для користувача для користування значно спрощується оскільки сайт має інтуїтивно зрозумілий інтерфейс і може бути використаним будь-якою людиною, що має навики роботи з комп'ютером. Серед основних моментів можна виділити загальний вигляд сайту

Для того щоб купити обраний товар, клієнту потрібно зареєструвати тобі профіль в магазині (рис 3.11).

Для цього потрібно заповнити 4 поля:

– логін;

– емайл;

– пароль;

– пітвердження паролю;

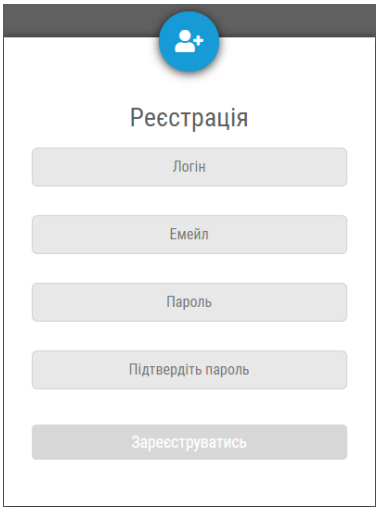

Рисунок 3.11 – Вікно реєстрації нового клієнта

Для кожного поля встановленна валідацію (рис 3.12):

– логін – логін має містити як мінімум 4 символи і обов’язково написаний на англійські мові;

– емайл – має обов’язково містити знак «@»;

– пароль – має містити як мінімум 6 символів, різних регістрів;

–пітвердження паролю – потрібно повторити пароль користувачу індентично, як він вводив в полі «пароль»;

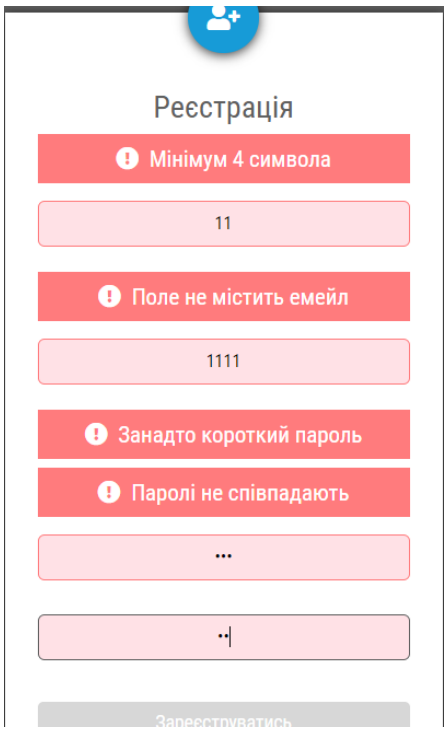

Рисунок 3.12– Вікно реєстрації ного клієнта

Щоб забезпечити максимально яскраве представлення товару, інформацію розділено на інформаційні категорії у вигляді ієрархічних зв’язків. Обравши відповідну категорію на екрані з’являються під категорії товарів, комп’ютерів, ноутбуків, мобільних телефонів. Можна побачити що над товаром є надпис «Новинка» - це означає що на сайті з’явився новий товар, «Хіт продаж» - це означає що товар має велику популярність, і покупець має можливість його

переглянути. Також можна побачити каталог товарів певних виробників, ціну яка влаштовує – це головне до кожного підрозділа.

Після реєстрації, входу в особистий кабінет клієнт має можливість обрати відповідний товар і добавити його собі в кошик.

Після вибору товару клієнт може переглянути вибрані товари в кошику а також замінити кількість товару якщо це потрібно (рис 3.13). Обравши певний товар є можливість подивитися чи є він в наявності, дізнатися ціну, прочитати його характеристику, подивитися детально його на фото, при потребі змінюючи маштаб фото.

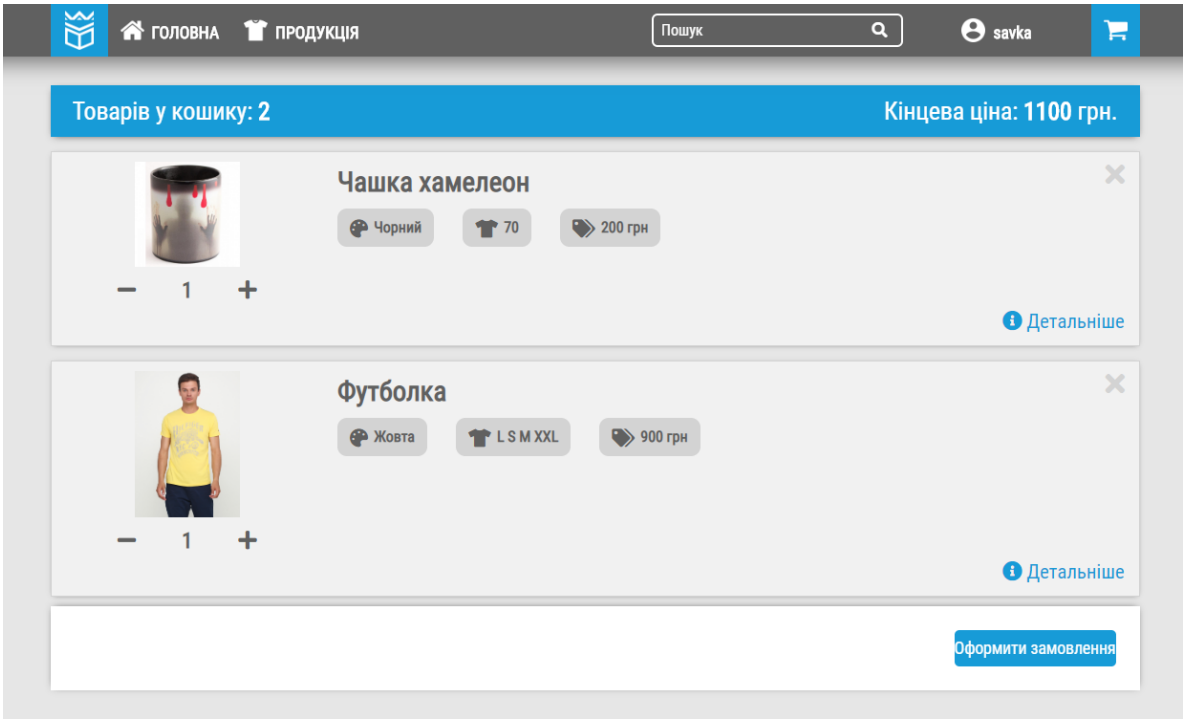

Рисунок 3.13 – Вікно обраного товару «Кошик»

Натиснувши на кнопку «Кошик» з’являється вікно «Ваша корзина» у ньому є можливість побачити який товар обрали, кількість товару і суму яку мають оплатити за товар.

Після цього можливе продовження покупки натиснувши «Продовжити покупки» або «Оформити замовлення».

Після чого він заповнює поля в яких вказується наступні дані (рис 3.14):

– п. і. б. отримувача;

– контактний номер телефону (мобільний, домашній, робочий на вибір);

– місто;

– область;

– вулиця;

– номер будинку;

– квартира;

– номер поштового відділення або інша уточнююча інформація;

– коментар до замовлення (не є обов’язковим для заповнення);

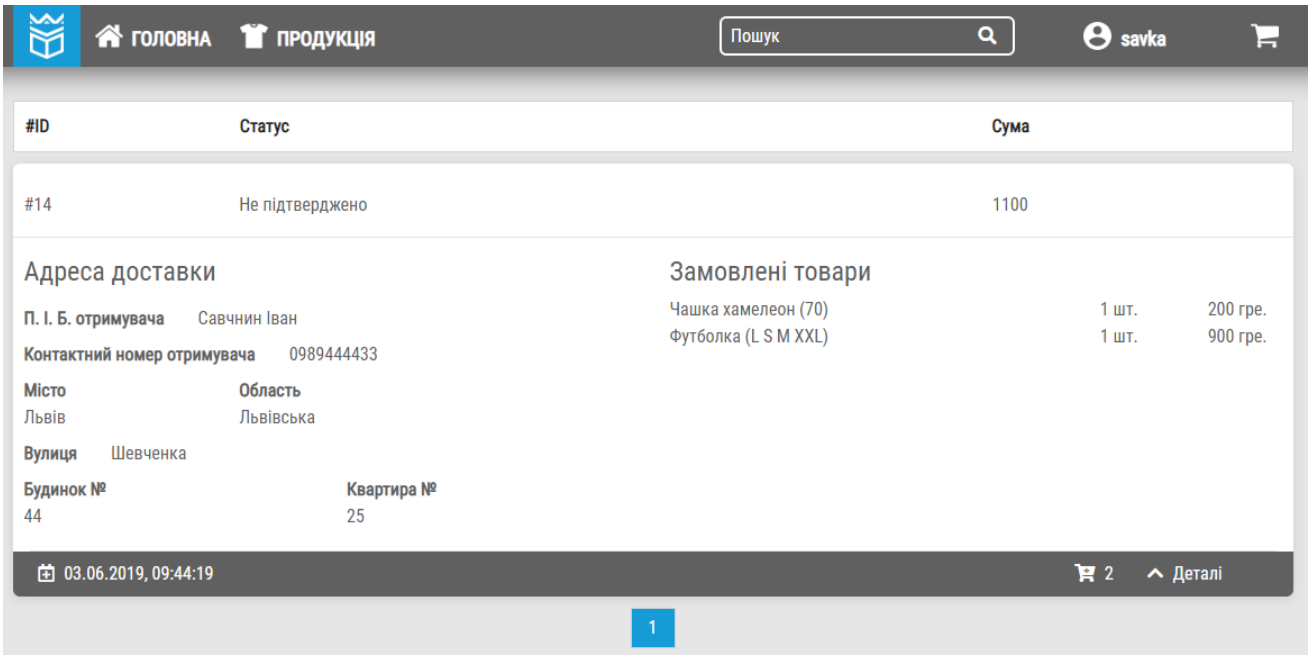

Рисунок 3.15 – Вікно перегляду статусу замовлення

Клінєнт має можливість преглядати статус свого замовлення, дату створення замовлення а також переглянути деталі свого замовлення (рис 3.15). Відповідно до стадії розгляду замовлення воно міняється, а разі потреби замовлення можна відмінити якщо товар ще не був надісланий покупцю:

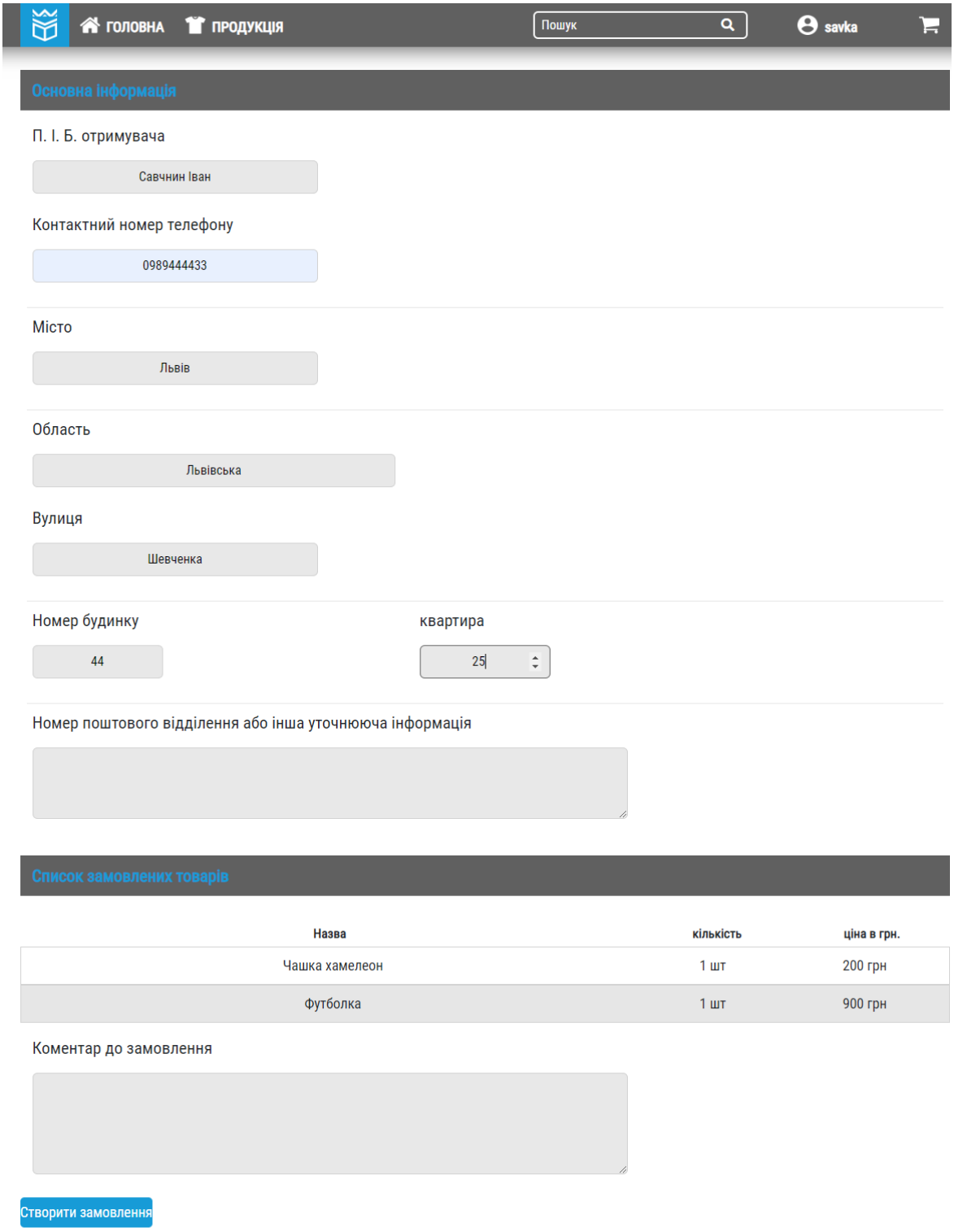

Рисунок 3.14 – Вікно оформлення замавлення

– не підвердженно;

– замовлення підтвердженно, очікуйте відправки;

– замовлення відправлено;

ВИСНОВКИ

Інтернет-магазин (електронний магазин, онлайновий магазин) - це програмний комплекс, який дозволяє продавати товари чи послуги через мережу Інтернет та автоматизувати управління бізнес-процесами.

Електронна торгівля у віртуальному магазині ґрунтується на тій же структурі, що і традиційна торгівля.

Класифікують Інтернет-магазини по різних критеріях: по моделі бізнесу, по відношенню з постачальниками, по товарному асортименту, по методах роздрібного продажу товарів в мережі.

Інтернет-магазини мають свої як позитивні так і негативні сторони. До переваг відносять: економія часу, низькі ціни, безкоштовна доставка, свобода вибору, психологічний комфорт, зручність оплати, анонімність. До недоліків Інтернет-магазину відносять: недосконала система доставки, незручна система оплати, складна система замовлення, нестабільний асортимент, необхідність реєстрації, проблема віртуальності, невиконання вибору замовлення, втрата замовлення.

Типовий варіант Інтернет-магазину складається з наступних функціональних частин: каталог товарів, пошукова система, корзина, реєстраційна форма, форма відправки замовлення.

Враховуючи основні вимоги до створення Інтернет-магазину та за допомогою сучасних Інтернет технологій мови програмування PHP та системи управління базами даних MySQL, було розробленно Web-сайт Інтернет-магазин, функціональна структура якого складається з Web-сайту та бази даних, дані елементи взаємодіють між собою.

Структура бази даних складається з 9 таблиць, які пов’язані між собою зв’язком типу один до багатьох, що забезпечує збереження цілісності даних. Структура Web-сайту базується на адміністративній та клієнтській частині.

Позитивними якостями розробленого Інтернет-магазину є:

– зручний інтерфейс користувача;

–відсутність реєстрації клієнтів магазину, що забезпечує їхню анонімність;

–сортування товарів по групах, що пришвидшує пошук потрібного товару;

– вибір оплати та доставки товару користувачем;

– невелика за розміром форма для замовлення;

– автоматична зміна даних в базі даних магазину при здійсненні замовлення обраного товару.

Негативними якостями Інтернет-магазину є:

– неповна база товарів;

– відсутність можливості оплати новітніми засобами;

– ціни товарів не представленні в іноземній валюті.

Враховуючи вище сказане можна зробити висновок, що всі поставленні завдання в дипломній роботі виконані в повному обсязі.

Список використаних джерел

1. JavaScript. Матеріал з habr. URL: https://habr.com/ru/company/ruvds/blog/429552 (дата звернення: 15.02.2019).

2. Vue Js. Матеріал з Вікіпедії – вільної енциклопедії URL: http://uk.wikipedia.org/wiki C_Sharp (дата звернення 09.01.2019).

3. Айверсон У. Популярные Web-сервисы: практика использования. 2005. 240 с.

4. Акила К. Интернет-магазин с нуля. Полное пошаговое руководство – Питер, 2013. 176 с.

5. Алексеев А.А. Как сделать рекламу Вашей фирмы в сети Интернет: Пошаговое руководство для фирм, желающих сделать рекламу своей продукции в сети Интернет. 1997. 237 с.

6. Бейли. Л. Изучаем SQL. Санкт-Петербург: Питер, 2012. – 276 с.

7. Вигерс Карл. Разработка требований к программному обеспечению ИД «Русская редакция» Вільна енциклопедія Вікіпедія URL: http://uk.wikipedia.org (дата звернення 19.12.2018).

8. Гарретт, Д. Веб-дизайн. Элементы опыта взаимодействия, Одеса, 2015. 192 с.

9. Глинський.Я.М. Internet, Сервіси, HTML і web дизайн. Деол, 2000 з. 168 с.

10. ГОСТ 2.004–88 ЕСКД. Общие требования к вьполнению конструкторских и технологических документов на печатающих и графических устройствах вывода ЭВМ.

11. Гради Буч. Об'єктно-орієнтований аналіз і проектування. Одеса, 2000. 560с.

12. Давыдов В.И.б Кунгурцев А.Б. Об'єктно-орієнтоване програмування. Одеса, 2003. 124с.

13. Дакетт, Д., М.: Эксмо HTML и CSS. Разработка и дизайн веб-сайтов, 2015. 480 с.

14. Довідник по HTML та CSS. URL: http://htmlbook.ru (дата звернення: 12.01.2019).

15. Дэвид Сойер Макфарланд. JavaScript и jQuery. Исчерпывающее руководство. Пер. с англ. — М.: Издательский дом «Вильямс», 2017. 450

16. Соммервилл Иан. Інженерія програмного забезпечення. М. Вильямс, 2002. 624с.

17. Дэвид Флэнаган, JavaScript – Подробное руководство 6-ое издание Пер. с англ. М.: Издательский дом «Вильямс», 2015. 300 с.

18. Дэвид Флэнаган, JavaScript. Карманный справочник. Пер. з англ. «Питер», 2015. 600с.

19. Кристиан Дари, Эмилиан Баланеску. PHP и MySQL: создание интернет-магазина. Вильямс, 2010. 640 с.

20. Леонтьев Б. PHP 5.0 для начинающих, или как создать динамичный web-сайт. Одеса, 2006. 176 с.

21. Липпман С., Язык программирования JavaScript. Вводный курс, Невский проспект, 2006. 1406 с.

22. Малышенко А.А. WEB. Дизайн и коммерция. Конспект программиста. Одеса, 2000. 192 с.

23. Орлов С.А. Технология розробки програмного забезпечення. СПБ. Питер, 2002. 464 с.

24. Писарев А.А., Соловьев Д.С. Интернет-магазин без правил, Питер, 2013. 176 с.

25. Седерхольм, Д. Пуленепробиваемый веб-дизайн. Библиотека специалиста. Питер, 2012. 304 с.

26. Энди Харрис PHP/MySQL для початківців, Одеса, 2005. 384 с.

Додаток А

Файл Build.js

```
'use strict'
require('./check-versions')()

process.env.NODE_ENV = 'production'

const ora = require('ora')
const rm = require('rimraf')
const path = require('path')
const chalk = require('chalk')
const webpack = require('webpack')
const config = require('../config')
const webpackConfig = require('./webpack.prod.conf')

const spinner = ora('building for production...')
spinner.start()

rm(path.join(config.build.assetsRoot,
config.build.assetsSubDirectory), err => {
  if (err) throw err
  webpack(webpackConfig, (err, stats) => {
    spinner.stop()
    if (err) throw err
    process.stdout.write(stats.toString({
      colors: true,
      modules: false,
      children: false, // If you are using ts-loader, setting
this to true will make TypeScript errors show up during build.
      chunks: false,
      chunkModules: false
    }) + '\n\n')

    if (stats.hasErrors()) {
      console.log(chalk.red('  Build failed with errors.\n'))
```

```
            process.exit(1)
          }

          console.log(chalk.cyan('  Build complete.\n'))
          console.log(chalk.yellow(
            '  Tip: built files are meant to be served over an HTTP
server.\n' +
            '  Opening index.html over file:// won\'t work.\n'
          ))
        })
      })
```

Файл Check-versions.js

```
      'use strict'
      const chalk = require('chalk')
      const semver = require('semver')
      const packageConfig = require('../package.json')
      const shell = require('shelljs')

      function exec (cmd) {
        return require('child_process').execSync(cmd).toString().trim()
      }

      const versionRequirements = [
        {
          name: 'node',
          currentVersion: semver.clean(process.version),
          versionRequirement: packageConfig.engines.node
        }
      ]

      if (shell.which('npm')) {
        versionRequirements.push({
          name: 'npm',
          currentVersion: exec('npm --version'),
          versionRequirement: packageConfig.engines.npm
```

```
  })
}

module.exports = function () {
  const warnings = []

  for (let i = 0; i < versionRequirements.length; i++) {
    const mod = versionRequirements[i]

    if (!semver.satisfies(mod.currentVersion, mod.versionRequirement)) {
      warnings.push(mod.name + ': ' +
        chalk.red(mod.currentVersion) + ' should be ' +
        chalk.green(mod.versionRequirement)
      )
    }
  }

  if (warnings.length) {
    console.log('')
    console.log(chalk.yellow('To use this template, you must update following to modules:'))
    console.log()
    for (let i = 0; i < warnings.length; i++) {
      const warning = warnings[i]
      console.log('  ' + warning)
    }
    console.log()
    process.exit(1)
  }
}
```

Файл Utils.js

```
'use strict'
const path = require('path')
const config = require('../config')
```

```
const ExtractTextPlugin = require('extract-text-webpack-plugin')
const packageConfig = require('../package.json')

exports.assetsPath = function (_path) {
  const assetsSubDirectory = process.env.NODE_ENV === 'production'
    ? config.build.assetsSubDirectory
    : config.dev.assetsSubDirectory

  return path.posix.join(assetsSubDirectory, _path)
}

exports.cssLoaders = function (options) {
  options = options || {}

  const cssLoader = {
    loader: 'css-loader',
    options: {
      sourceMap: options.sourceMap
    }
  }

  const postcssLoader = {
    loader: 'postcss-loader',
    options: {
      sourceMap: options.sourceMap
    }
  }

  // generate loader string to be used with extract text plugin
  function generateLoaders (loader, loaderOptions) {
    const loaders = options.usePostCSS ? [cssLoader, postcssLoader] : [cssLoader]

    if (loader) {
      loaders.push({
        loader: loader + '-loader',
        options: Object.assign({}, loaderOptions, {
          sourceMap: options.sourceMap
```

```
          })
        })
      }

      // Extract CSS when that option is specified
      // (which is the case during production build)
      if (options.extract) {
        return ExtractTextPlugin.extract({
          use: loaders,
          fallback: 'vue-style-loader'
        })
      } else {
        return ['vue-style-loader'].concat(loaders)
      }
    }

    //      https://vue-loader.vuejs.org/en/configurations/extract-
css.html
    return {
      css: generateLoaders(),
      postcss: generateLoaders(),
      less: generateLoaders('less'),
      sass: generateLoaders('sass', { indentedSyntax: true }),
      scss: generateLoaders('sass'),
      stylus: generateLoaders('stylus'),
      styl: generateLoaders('stylus')
    }
  }

  // Generate loaders for standalone style files (outside of .vue)
  exports.styleLoaders = function (options) {
    const output = []
    const loaders = exports.cssLoaders(options)

    for (const extension in loaders) {
      const loader = loaders[extension]
      output.push({
        test: new RegExp('\\.' + extension + '$'),
        use: loader
```

```
    })
  }

  return output
}

exports.createNotifierCallback = () => {
  const notifier = require('node-notifier')

  return (severity, errors) => {
    if (severity !== 'error') return

    const error = errors[0]
    const filename = error.file && error.file.split('!').pop()

    notifier.notify({
      title: packageConfig.name,
      message: severity + ': ' + error.name,
      subtitle: filename || '',
      icon: path.join(__dirname, 'logo.png')
    })
  }
}
```

Файл Vue-loader.conf.js

```
'use strict'
const utils = require('./utils')
const config = require('../config')
const isProduction = process.env.NODE_ENV === 'production'
const sourceMapEnabled = isProduction
  ? config.build.productionSourceMap
  : config.dev.cssSourceMap

module.exports = {
  loaders: utils.cssLoaders({
    sourceMap: sourceMapEnabled,
```

```
    extract: isProduction
  }),
  cssSourceMap: sourceMapEnabled,
  cacheBusting: config.dev.cacheBusting,
  transformToRequire: {
    video: ['src', 'poster'],
    source: 'src',
    img: 'src',
    image: 'xlink:href'
  }
}
```

Файл Webpack.base.conf.js

```
'use strict'
const path = require('path')
const utils = require('./utils')
const config = require('../config')
const vueLoaderConfig = require('./vue-loader.conf')

function resolve (dir) {
  return path.join(__dirname, '..', dir)
}
module.exports = {
  context: path.resolve(__dirname, '../'),
  entry: {
    app: './src/main.js'
  },
  output: {
    path: config.build.assetsRoot,
    filename: '[name].js',
    publicPath: process.env.NODE_ENV === 'production'
      ? config.build.assetsPublicPath
      : config.dev.assetsPublicPath
  },
  resolve: {
    extensions: ['.js', '.vue', '.json'],
```

```
    alias: {
      'vue$': 'vue/dist/vue.esm.js',
      '@': resolve('src'),
    }
  },
  module: {
    rules: [
      {
        test: /\.vue$/,
        loader: 'vue-loader',
        options: vueLoaderConfig
      },
      {
        test: /\.js$/,
        loader: 'babel-loader',
        include:         [resolve('src'),         resolve('test'),
resolve('node_modules/webpack-dev-server/client')]
      },
      {
        test: /\.(png|jpe?g|gif|svg)(\?.*)?$/,
        loader: 'url-loader',
        options: {
          limit: 10000,
          name: utils.assetsPath('img/[name].[hash:7].[ext]')
        }
      },
      {
        test: /\.(mp4|webm|ogg|mp3|wav|flac|aac)(\?.*)?$/,
        loader: 'url-loader',
        options: {
          limit: 10000,
          name: utils.assetsPath('media/[name].[hash:7].[ext]')
        }
      },
      {
        test: /\.(woff2?|eot|ttf|otf)(\?.*)?$/,
        loader: 'url-loader',
        options: {
          limit: 10000,
```

```
            name: utils.assetsPath('fonts/[name].[hash:7].[ext]')
          }
        }
      ]
    },
    node: {
      dgram: 'empty',
      fs: 'empty',
      net: 'empty',
      tls: 'empty',
      child_process: 'empty'
    }
  }
```

Файл Webpack.dev.conf.js

```
  'use strict'
  const utils = require('./utils')
  const webpack = require('webpack')
  const config = require('../config')
  const merge = require('webpack-merge')
  const path = require('path')
  const baseWebpackConfig = require('./webpack.base.conf')
  const CopyWebpackPlugin = require('copy-webpack-plugin')
  const HtmlWebpackPlugin = require('html-webpack-plugin')
  const  FriendlyErrorsPlugin  =  require('friendly-errors-webpack-
plugin')
  const portfinder = require('portfinder')

  const HOST = process.env.HOST
  const PORT = process.env.PORT && Number(process.env.PORT)

  const devWebpackConfig = merge(baseWebpackConfig, {
    module: {
      rules:             utils.styleLoaders({             sourceMap:
config.dev.cssSourceMap, usePostCSS: true })
    },
```

```
    devServer: {
      clientLogLevel: 'warning',
      historyApiFallback: {
        rewrites: [
          {               from:               /.*/,               to:
path.posix.join(config.dev.assetsPublicPath, 'index.html') },
        ],
      },
      hot: true,
      contentBase: false, // since we use CopyWebpackPlugin.
      compress: true,
      host: HOST || config.dev.host,
      port: PORT || config.dev.port,
      open: config.dev.autoOpenBrowser,
      overlay: config.dev.errorOverlay
        ? { warnings: false, errors: true }
        : false,
      publicPath: config.dev.assetsPublicPath,
      proxy: config.dev.proxyTable,
      quiet: true,
      watchOptions: {
        poll: config.dev.poll,
      }
    },
    plugins: [
      new webpack.DefinePlugin({
        'process.env': require('../config/dev.env')
      }),
      new webpack.HotModuleReplacementPlugin(),
      new webpack.NamedModulesPlugin(), // HMR shows correct file
names in console on update.
      new webpack.NoEmitOnErrorsPlugin(),
      new HtmlWebpackPlugin({
        filename: 'index.html',
        template: 'index.html',
        inject: true
      }),
      new CopyWebpackPlugin([
        {
```

```
            from: path.resolve(__dirname, '../static'),
            to: config.dev.assetsSubDirectory,
            ignore: ['.*']
          }
        ])
      ]
    })

    module.exports = new Promise((resolve, reject) => {
      portfinder.basePort = process.env.PORT || config.dev.port
      portfinder.getPort((err, port) => {
        if (err) {
          reject(err)
        } else {

          devWebpackConfig.plugins.push(new FriendlyErrorsPlugin({
            compilationSuccessInfo: {
              messages: [`Your application is running here:
http://${devWebpackConfig.devServer.host}:${port}`],
            },
            onErrors: config.dev.notifyOnErrors
            ? utils.createNotifierCallback()
            : undefined
          }))

          resolve(devWebpackConfig)
        }
      })
    })
```

Файл Webpack.prod.conf.js

```
    'use strict'
    const path = require('path')
    const utils = require('./utils')
    const webpack = require('webpack')
    const config = require('../config')
```

```
const merge = require('webpack-merge')
const baseWebpackConfig = require('./webpack.base.conf')
const CopyWebpackPlugin = require('copy-webpack-plugin')
const HtmlWebpackPlugin = require('html-webpack-plugin')
const ExtractTextPlugin = require('extract-text-webpack-plugin')
const  OptimizeCSSPlugin  =  require('optimize-css-assets-webpack-
plugin')
const UglifyJsPlugin = require('uglifyjs-webpack-plugin')

const env = require('../config/prod.env')

const webpackConfig = merge(baseWebpackConfig, {
  module: {
    rules: utils.styleLoaders({
      sourceMap: config.build.productionSourceMap,
      extract: true,
      usePostCSS: true
    })
  },
  devtool:           config.build.productionSourceMap           ?
config.build.devtool : false,
  output: {
    path: config.build.assetsRoot,
    filename: utils.assetsPath('js/[name].[chunkhash].js'),
    chunkFilename: utils.assetsPath('js/[id].[chunkhash].js')
  },
  plugins: [
    new webpack.DefinePlugin({
      'process.env': env
    }),
    new UglifyJsPlugin({
      uglifyOptions: {
        compress: {
          warnings: false
        }
      },
      sourceMap: config.build.productionSourceMap,
      parallel: true
    }),
```

```
new ExtractTextPlugin({
  filename: utils.assetsPath('css/[name].[contenthash].css'),
  allChunks: true,
}),
new OptimizeCSSPlugin({
  cssProcessorOptions: config.build.productionSourceMap
    ? { safe: true, map: { inline: false } }
    : { safe: true }
}),
new HtmlWebpackPlugin({
  filename: config.build.index,
  template: 'index.html',
  inject: true,
  minify: {
    removeComments: true,
    collapseWhitespace: true,
    removeAttributeQuotes: true
  },
  chunksSortMode: 'dependency'
}),
new webpack.optimize.CommonsChunkPlugin({
  name: 'vendor',
  minChunks (module) {
    return (
      module.resource &&
      /\.js$/.test(module.resource) &&
      module.resource.indexOf(
        path.join(__dirname, '../node_modules')
      ) === 0
    )
  }
}),
new webpack.optimize.CommonsChunkPlugin({
  name: 'manifest',
  minChunks: Infinity
}),
new webpack.optimize.CommonsChunkPlugin({
  name: 'app',
  async: 'vendor-async',
```

```
            children: true,
            minChunks: 3
          }),
          new CopyWebpackPlugin([
            {
              from: path.resolve(__dirname, '../static'),
              to: config.build.assetsSubDirectory,
              ignore: ['.*']
            }
          ])
        ]
      })

      if (config.build.productionGzip) {
        const CompressionWebpackPlugin = require('compression-webpack-
plugin')

        webpackConfig.plugins.push(
          new CompressionWebpackPlugin({
            asset: '[path].gz[query]',
            algorithm: 'gzip',
            test: new RegExp(
              '\\.(' +
              config.build.productionGzipExtensions.join('|') +
              ')$'
            ),
            threshold: 10240,
            minRatio: 0.8
          })
        )
      }

      if (config.build.bundleAnalyzerReport) {
        const BundleAnalyzerPlugin = require('webpack-bundle-
analyzer').BundleAnalyzerPlugin
        webpackConfig.plugins.push(new BundleAnalyzerPlugin())
      }

      module.exports = webpackConfig
```

Файл Index.js

```
'use strict'

const path = require('path')

module.exports = {
  dev: {

    assetsSubDirectory: 'static',
    assetsPublicPath: '/',
    proxyTable: {},

    host: 'localhost', // can be overwritten by process.env.HOST
    port: 8080, // can be overwritten by process.env.PORT, if port is in use, a free one will be determined
    autoOpenBrowser: false,
    errorOverlay: true,
    notifyOnErrors: true,
    poll: false, // https://webpack.js.org/configuration/dev-server/#devserver-watchoptions-

    devtool: 'cheap-module-eval-source-map',

    cacheBusting: true,

    cssSourceMap: true
  },

  build: {
    index: path.resolve(__dirname, '../dist/index.html'),

    assetsRoot: path.resolve(__dirname, '../dist'),
    assetsSubDirectory: 'static',
    assetsPublicPath: './',
```

```
        productionSourceMap: true,
        devtool: '#source-map',
        productionGzip: false,
        productionGzipExtensions: ['js', 'css'],
        bundleAnalyzerReport: process.env.npm_config_report
      }
    }
```

Файл ProductValidator.js

```
    import {$http} from "../axiosConfig";
    import JwtHelper from "../helpers/jwtHelper";

    function getConfig() {
        return {
            headers: {
                ...JwtHelper.getAuthorizationHeader()
            }
        }
    }

    export function checkProduct(name, color) {
        return new Promise((resolve, reject) => {
            $http.post('/api/product/exist',      {name,      color},
getConfig())
                .then((response) => {
                    resolve(!response.data.isExist);
                })
                .catch((error) => {
                    reject();
                })
        })
    }
```

Додаток Б

Файл Package.json

```
{
  "name": "kdu_shop",
  "version": "1.0.0",
  "description": "",
  "author": "Marius <marius.shiman@gmail.com>",
  "private": true,
  "scripts": {
    "dev": "webpack-dev-server --inline --progress --config build/webpack.dev.conf.js",
    "start": "npm run dev",
    "build": "node build/build.js"
  },
  "dependencies": {
    "axios": "^0.18.0",
    "normalize.css": "^8.0.1",
    "vue": "^2.5.2",
    "vue-body-class": "^2.0.0-beta.2",
    "vue-router": "^3.0.1",
    "vue-sweetalert2": "^1.6.4",
    "vuelidate": "^0.7.4",
    "vuex": "^3.1.0"
  },
  "devDependencies": {
    "autoprefixer": "^7.1.2",
    "babel-core": "^6.22.1",
    "babel-helper-vue-jsx-merge-props": "^2.0.3",
    "babel-loader": "^7.1.1",
    "babel-plugin-syntax-jsx": "^6.18.0",
    "babel-plugin-transform-runtime": "^6.22.0",
    "babel-plugin-transform-vue-jsx": "^3.5.0",
    "babel-preset-env": "^1.3.2",
    "babel-preset-stage-2": "^6.22.0",
    "chalk": "^2.0.1",
    "copy-webpack-plugin": "^4.0.1",
```

```
    "css-loader": "^0.28.0",
    "extract-text-webpack-plugin": "^3.0.0",
    "file-loader": "^1.1.4",
    "friendly-errors-webpack-plugin": "^1.6.1",
    "html-webpack-plugin": "^2.30.1",
    "node-notifier": "^5.1.2",
    "optimize-css-assets-webpack-plugin": "^3.2.0",
    "ora": "^1.2.0",
    "portfinder": "^1.0.13",
    "postcss-import": "^11.0.0",
    "postcss-loader": "^2.0.8",
    "postcss-url": "^7.2.1",
    "rimraf": "^2.6.0",
    "semver": "^5.3.0",
    "shelljs": "^0.7.6",
    "uglifyjs-webpack-plugin": "^1.1.1",
    "url-loader": "^0.5.8",
    "vue-loader": "^13.3.0",
    "vue-style-loader": "^3.0.1",
    "vue-template-compiler": "^2.5.2",
    "webpack": "^3.6.0",
    "webpack-bundle-analyzer": "^2.9.0",
    "webpack-dev-server": "^2.9.1",
    "webpack-merge": "^4.1.0"
  },
  "engines": {
    "node": ">= 6.0.0",
    "npm": ">= 3.0.0"
  },
  "browserslist": [
    "> 1%",
    "last 2 versions",
    "not ie <= 8"
  ]
}
```

Файл AdminPanel.css

```
src/assets/css/adminPanel.css.admin-panel-header {
  color: white; }

.item {
  background: white;
  padding: 8px 15px;
  font-size: 1.1em;
  font-weight: bold;
  cursor: pointer;
  margin: 5px 0 0 0;
  border-radius: 10px 10px 0 0;
  box-shadow: 3px 0px 5px -1px rgba(0, 0, 0, 0.75); }

.panel-item {
  background: white;
  padding: 8px 15px;
  font-size: 1.1em;
  font-weight: bold;
  cursor: pointer;
  margin: 5px 0 0 0; }

.item-active {
  background: #179bd7;
  color: white; }

.list > .list-item:nth-child(even) {
  background: #e6e6e6; }

.list-item {
  padding: 8px;
  font-size: 1.3em;
  border: 1px solid lightgray; }
  .list-item, .list-item a {
    color: #606060; }

.panel {
```

```
  padding: 15px;
  border-radius: 0 6px 6px 6px;
  background: white;
  box-shadow: 0 2px 10px 0 rgba(0, 0, 0, 0.75); }

.panel-content {
  padding: 0 35px; }

.tab-panel {
  flex-wrap: nowrap;
  overflow-x: auto; }
```

Файл Cart.css

```
.cart-header {
  background: #179bd7;
  color: white;
  padding: 8px;
  font-size: 1.6em;
  position: sticky;
  top: 0;
  z-index: 99;
  box-shadow: 0px 2px 8px -4px rgba(0, 0, 0, 0.75); }

.order-bar {
  background: white;
  margin-top: 10px;
  padding: 25px 25px;
  position: sticky;
  bottom: 0;
  box-shadow: 0px -1px 8px -4px rgba(0, 0, 0, 0.75); }

.empty-cart-segment {
  margin-top: 120px;
  color: #606060;
  text-align: center; }
  .empty-cart-segment i {
```

```
    font-size: 100px;
    display: block; }
  .empty-cart-segment span {
    display: block;
    font-size: 1.1em;
    font-weight: bolder; }
  .empty-cart-segment button {
    padding: 10px 40px;
    font-size: .7em; }
```

Файл CartItem.css

```
.item {
  box-shadow: 0 1px 6px -4px rgba(0, 0, 0, 0.75);
  color: #606060;
  background: #f1f1f1;
  margin-top: 15px;
  border: 1px solid lightgrey;
  padding: 10px 15px; }
  .item img {
    max-width: 110px; }

.quantity-bar {
  font-size: 1.4em; }
  .quantity-bar i {
    border-radius: 6px;
    color: #606060; }
    .quantity-bar i:hover {
      padding: 5px;
      color: white;
      background: #179bd7; }

.delete-btn i {
  color: lightgrey;
  font-size: 1.8em; }

.item-header {
```

```
  font-size: 1.8em;
  font-weight: bolder; }

.details-link {
  font-size: 1.3em;
  color: #179bd7; }

.item-info {
  margin-top: 6px; }
  .item-info .row {
    padding: 7px 13px;
    align-items: center;
    height: 40px;
    line-height: 26px;
    background: lightgray;
    color: #606060;
    border-radius: 10px;
    font-weight: bold;
    font-size: 1em; }
    .item-info .row i {
      margin-right: 6px;
      font-size: 1.2em; }
```

Файл ErrorLabel.css

```
.error {
  width: 100%;
  text-align: center;
  color: white;
  background: #ff7b7d;
  overflow: hidden;
  transition: all .25s;
  line-height: 45px;
  height: 45px;
  padding: 0 5px; }
  .error i {
    margin-right: 5px; }
```

```
.slide-enter-active, .slide-leave-active {
  transition: all .3s;
  transition-delay: .3s; }

.slide-enter, .slide-leave-to {
  height: 0;
  margin: 0; }
```

Файл Form.css

```
.form {
  position: relative;
  transition: all .3s; }

.form-box {
  box-shadow: 0 2px 10px 0 rgba(0, 0, 0, 0.75);
  width: 400px;
  background: white;
  color: #606060;
  padding: 70px 30px 50px 30px; }

.form-box-2 {
  background: white;
  color: #606060; }

.form-2 {
  margin-top: 20px;
  transition: all .3s;
  color: #606060; }

.form-group-1 {
  margin: 0 0 15px 0; }
  .form-group-1 input, .form-group-1 select, .form-group-1
textarea {
    background: #e8e8e8;
    text-align: center;
```

```
          border: 1px solid lightgray;
          border-radius: 6px;
          padding: 8px 10px;
          margin: 8px 0;
          width: 100%;
          outline: none; }
          .form-group-1 input:focus, .form-group-1 select:focus, .form-
group-1 textarea:focus {
            border: 1px solid #606060; }
        .form-group-1 label {
          font-size: 1.3em; }
        .form-group-1 textarea {
          text-align: left; }

      .group-error {
        transition: all .3s; }
        .group-error input, .group-error select textarea {
          border: 1px solid #ff7b7d;
          background: #ffe1e6; }

      .form-header {
        margin-top: -1px;
        font-size: 1.5em;
        font-weight: bolder;
        color: #179bd7;
        background: #606060;
        padding: 1px 20px; }

      .form-group-header {
        margin-top: -1px;
        font-size: 1.3em;
        font-weight: bolder;
        color: #179bd7;
        background: #606060;
        padding: 10px 15px; }

      .error-label {
        overflow: hidden;
        font-weight: bold;
```

```
  font-size: 1.1em;
  color: white;
  background: #c70000;
  padding: 0 20px;
  height: 0;
  line-height: 40px;
  text-align: center;
  transition: height .25s; }

.error-label-show {
  height: 40px; }

.success-message {
  transition: all .3s;
  font-size: 1.2em;
  font-weight: bold;
  color: white;
  overflow: hidden;
  max-height: 0;
  padding: 0;
  background: #008028; }

.error-message {
  transition: all .3s;
  font-size: 1.2em;
  font-weight: bold;
  color: white;
  overflow: hidden;
  max-height: 0;
  padding: 0;
  background: #c70000; }

.message-show {
  max-height: 75px;
  padding: 10px; }

.form-icon {
  box-shadow: 0px 2px 10px 0px rgba(0, 0, 0, 0.75);
  font-size: 1.5em;
```

```
  position: absolute;
  top: -28px;
  width: 65px;
  height: 65px;
  border-radius: 100%;
  line-height: 65px;
  vertical-align: middle;
  text-align: center;
  color: white;
  background: #179bd7; }

.file-selected {
  background: green;
  width: 100%;
  color: white;
  padding: 3px 8px; }
```

Файл Main.css

```
body {
  background: #e6e6e6; }

.pagination .page-active {
  background: #179bd7;
  color: white; }
.pagination a {
  border-top: 1px solid lightgrey;
  border-right: 1px solid lightgrey;
  border-bottom: 1px solid lightgrey;
  min-width: 40px;
  padding: 5px;
  text-align: center;
  text-decoration: none;
  color: #606060; }
  .pagination a:first-child {
    border-left: 1px solid lightgrey; }
```

```
.submit-btn {
  border: 0;
  background: #179bd7;
  color: white;
  border-radius: 5px;
  margin: 15px 0 0 0;
  padding: 8px 15px;
  font-size: 1.3em;
  transition: background-color .3s; }
  .submit-btn:disabled {
    background: #d7d7d7;
    cursor: no-drop; }

.body-dark {
  background: #606060; }

.body-light {
  background: white; }
```

Файл Nav.css

```
.nav-bar {
  z-index: 99999;
  position: relative;
  background: #606060;
  margin: 0;
  padding: 0;
  width: 100%;
  box-shadow: 0px -9px 25px 12px rgba(0, 0, 0, 0.59); }

.logo {
  display: flex;
  height: 55px;
  background: #179bd7;
  align-items: center; }
  .logo img {
    width: 30px;
```

```
    height: auto; }

.nav-link {
  padding: 0; }
  .nav-link .link-content {
    height: 55px;
    padding-left: 13px;
    padding-right: 13px;
    display: flex;
    align-items: center;
    justify-content: center; }
    .nav-link .link-content i {
      color: white;
      font-size: 1.3em; }
    .nav-link .link-content img {
      display: flex;
      height: 25px;
      width: auto; }
    .nav-link .link-content span {
      display: flex;
      margin: 0;
      padding: 4px 0 0 8px;
      font-size: 1.3em;
      font-weight: bold;
      color: white; }
    .nav-link .link-content a {
      color: white; }
  .nav-link:hover {
    cursor: pointer;
    background: #179bd7; }
  .nav-link:hover .item-dropdown {
    display: block; }

.nav-link-active .link-content {
  background: #179bd7; }

.item-dropdown {
  background: #262626;
  width: 100%;
```

```
  display: none; }
  .item-dropdown a {
    text-decoration: none;
    display: block;
    padding: 5px 5px 5px 14px;
    color: white;
    font-size: 1.1em; }
    .item-dropdown a:hover {
      background: #179bd7; }

.search-field {
  height: 100%;
  padding: 10px; }
  .search-field input {
    border-radius: 6px;
    padding-right: 10px;
    padding-left: 5px;
    color: white;
    display: inline;
    height: 35px;
    width: 100%;
    background: #606060;
    border: 2px solid white; }
  .search-field img {
    position: absolute;
    top: 19px;
    bottom: 19px;
    right: 18px;
    width: 17px;
    height: 17px; }
  .search-field label {
    display: block;
    height: 100%;
    width: 100%; }

.menu {
  display: none; }

.show-menu {
```

```
  display: block; }

@media all and (min-width: 768px) {
  .nav-bar {
    height: 55px; } }
```

Файл OrderItem.css

```
.order-item {
  border-radius: 6px;
  overflow: hidden;
  background: white;
  margin-top: 10px;
  margin-bottom: 10px;
  color: #606060; }

.ordered-item a {
  color: #606060; }

.order-body {
  padding: 25px 10px 10px 10px; }

.details-segment {
  background: #e3e3e3;
  padding: 8px;
  transition: all .25s; }
  .details-segment i {
    margin-right: 5px; }
  .details-segment:hover {
    cursor: pointer;
    background: #606060;
    color: white; }

.details-active {
  background: #606060;
  color: white; }
  .details-active .arrow i {
```

Файл Product-page.css

```
.img-segment {
  background: #e5e5e5; }
  .img-segment img {
    max-width: 100%; }

.size-radio {
  padding-left: 7px;
  padding-right: 7px; }
  .size-radio .wrapper {
    transition: all 0.2s;
    color: black; }
  .size-radio label {
    padding: 5px 10px 5px 10px;
    border: 1px solid #606060;
    border-radius: 6px;
    text-align: center;
    display: block;
    width: 100%; }
  .size-radio input {
    opacity: 0;
    position: absolute; }
    .size-radio input:checked ~ .wrapper {
      background: #606060;
      color: white; }

.cart-btn-text {
  font-size: 1.6em;
  font-weight: bold; }

.cart-modal {
  background: #606060; }
```

БІОГРАФІЧНА ДОВІДКА

Тема дипломної роботи: Розробка інтернет-магазину брендованої продукції для Університету Короля Данила.

Обсяг пояснювальної записки:. 60 аркушів

Перелік графічних матеріалів:

- таблиць 1
- рисунків 26
- додатків 2 на 30 аркушів.

Дата закінчення дипломного проекту: “ 10 ” 06 2019 р.

нт – дипломник _____
(підпис) (розшифровка підпису)