ДИПЛОМНА РОБОТА

ДР.ПІс – 05.00.000 ПЗ

Група ПІс-2015

Данилюк І. В.

2019

ПВНЗ Університет Короля Данила

Кафедра Інформаційних технологій та програмної інженерії

УДК 004

ДИПЛОМНА РОБОТА

Тема ***Розробка інтернет-магазину з продажу книг***

Напрям підготовки ***6.050103 «Програмна інженерія»***

(код і назва спеціальності)

ПОЯСНЮВАЛЬНА ЗАПИСКА

ДР.ПІс – 05.00.000 ПЗ

(позначення)

Студент

Данилюк І.В.

(підпис) (дата) (розшифрування підпису)

Керівник проекту

к.ф.-м.н.,доц. Остафійчук П.Г.

(посада) (підпис) (дата) (розшифрування підпису)

Нормоконтроль

к.т.н. Мануляк І.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*
Напрям підготовки *6.050103 «Програмна інженерія»*

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

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Студенту *Данилюку Івану Вікторовичу*

1. Тема проекту (роботи) *Розробка інтернет-магазину з продажу книг*

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) *CMS WordPress, MySQL база даних, PHP, CSS.*

4. Зміст пояснювальної записки (перелік питань, що їх належить розробити) *1. Актуальність розробки програмного забезпечення, та аналіз існуючих аналогів. 2. Моделювання веб додатку та аналіз засобів рощробки 3. Програмна реалізація веб додатку.*

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) *Постанова задач, моделювання сайту, контекстна діаграма, діаграма класів та діаграма декомпозиції, структурна схема сайту, структура головної сторінки сайту, головна сторінка веб-сайту.*

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 р.____

Керівник _____Остафійчук П. Г.

Завдання прийняв до виконання ____Данилюк І.В.

КАЛЕНДАРНИЙ ПЛАН

Пор №	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1.	Актуальність розробки програмного забезпечення та існуючих аналогів веб додатків	30.11.2018	
2.	Моделювання веб ресурсу та аналіз засобів розробки	25.12.2018	
3.	Програмна реалізація веб додатку	21.02.2019	
4.	Оформлення пояснювальної записки	31.05.2019	
5.	Оформлення графічного матеріалу та підготовка до захисту роботи	07.06.2019	

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

Керівник проекту _____*Остафійчук П. Г.*
(підпис) (розшифровка підпису

АНОТАЦІЯ

Дипломна робота присвячена розробці інтернет магазину з продажу книг. В ній реалізовано каталог книг, сортування книг за жанрами, зворотній зв’язок з адміністрацією, можливість придбати книги.

У першому розділі проведено дослідження на актуальність розробки веб ресурсу та аналіз вітчизняних аналогів, проаналізовано їхні переваги та недоліки.

У другому розділі проведено моделювання веб ресурсу та обрано програмне середовище яке найкраще підходило для реалізації продукту.

У третьому розділі представлені описання створення інтерфейсу даного програмного продукту.

SUMMARY

Thesis is devoted to the development of an online store selling books. In it implemented a catalog of books, sorting books by genre, feedback with the administration, the ability to buy books.

In the first section, research was conducted on the relevance of web resource development and analysis of domestic analogues, their advantages and disadvantages were analyzed.

In the second section, web resource modeling was conducted and the software environment that best suits the product was chosen.

The third section describes the creation of the interface of this software product.

РЕФЕРАТ

Розрахунково-пояснювальна записка: 65 сторінок, 26 малюнків, 2 додатки.

Дипломний проект присвячується розробці інтернет магазину з продажу книг.

Об’єкт дослідження є інтернет-магазин з продажу книг.

Метою роботи є розробка каталогу книг та поділу їх за жанрами, реалізація інтерфейсу веб додатку.

Перший розділ складається з аналізу існуючих аналогів та постановки завдання.

В другому розділі описано моделювання веб ресурсу та засоби розробки, їхні переваги та недоліки.

Третій розділ містить опис етапів розробки, а також опис всього веб інтерфейсу інтернет магазину.

ІНТЕРФЕЙС, СТРУКТУРА, WORDPRESS, АДМІНПАНЕЛЬ, КОРИСТУВАЧІ, ІНТЕРНЕТ–МАГАЗИН, MYSQL.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ 8
ВСТУП 9
1 АКТУАЛЬНІСТЬ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА АНАЛІЗ ІСНУЮЧИХ АНАЛОГІВ ВЕБ ДОДАТКІВ 11
1.1 Актуальність розробки 11
1.2 Аналіз існуючих аналогів 14
1.2.1 Огляд сайту «Yakaboo» 14
1.2.2 Огляд сайту «Книгаренька» 15
1.2.3 Огляд сайту «Книгарня Нора-Друк» 17
1.3 Постановка завдання 18
2 МОДЕЛЮВАННЯ ВЕБ РЕСУРСУ ТА АНАЛІЗ ЗАСОБІВ РОЗРОБКИ 20
2.1 Діаграма прецедентів 20
2.2 Діаграма класів 22
2.3 Контекстна діаграма 24
2.4 Діаграма декомпозиції 26
2.5 Аналіз функціональних засобів для реалізації системи 28
2.5.1 Опис CMS Joomla 29
2.5.2 Огляд CMS DataLife Engine 31
2.5.3 Огдяд CMS WordPress 34
2.5.4 Особливості скриптової мови PHP 37
2.5.5 Система управління базою даних MySQL 38
3 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ ДОДАТКУ 42
3.1 Етапи розробки 42
3.2 Принципи розробки дизайну сайта 43
3.3 Структура сайту 44
3.4 Веб інтерфейс 48

ВИСНОВКИ..... 61
ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 63
Додаток А..... 66
Додаток Б 77

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

CSS – Cascading Style Sheets

PHP – Hypertext Preprocessor – скриптова мова програмування

Sql – Structured query language – декларативна мова програмування для взаємодії користувача з базами даних

CMS – Content Management Systems – це програмне забезпечення для організації веб-сайтів чи інших інформаційних ресурсів в Інтернеті чи окремих комп'ютерних мережах

СУБД – система управління базою даних.

WP – WordPress – система керування вмістом з відкритим кодом

DLE – DataLife Engine – багатофункціональна система управління сайтом

Joomla – безкоштовна система для створення веб-сайтів

ВСТУП

Актуальність теми. В наш час Інтернет став найбільш зручним та популярним джерелом інформації. Еволюція сфери інформаційних технологій є чинником, що впливає на зміни в економіці, політиці, культурі та освіті. У цих умовах інформація стає одним з основних стратегічних ресурсів, а інформаційні технології - інструментом, з допомогою якого цей ресурс використовується. Доступ до інтернету має майже кожна людина, тому більшість компаній уже мають web-ресурси, тим паче, що це є одним із найефективніших способів реклами та місце розміщення для сервісів, які дозволять клієнтам робити замовлення, чи зробити покупку з будь-якої точки світу. Серед 40 країн, що висвітлені у дослідженні активності читацької аудиторії 2018 року, Україна знаходиться на 12 місці. 74% українців купили хоча б одну книгу – так само як у Великій Британії або Швейцарії. Більшість українських читачів надають перевагу старому-доброму друку. В сегменті електронних книжок ситуація неоднозначна. Попри те, що майже половина українців читає електронні книги, піратський сегмент цього ринку – у два рази більший за легальний.

На даний час, переважна більшість читацької аудиторії знаходиться в великих містах, в селах та маленьких містечках ситуація гірша. Але з розвитком інформаційних технологій в Інтернеті, з'явилася велика кількість веб - ресурсів де люди можуть придбати той чи інший товар не виходячи з дому.

Так як останнім часом читацька аудиторія збільшується, а далеко не в кожному обласному та районному центрі України є велика кількість інтернет – магазинів з продажу книг, постає потреба у їх створенні. Таким чином, можна завоювати аудиторію віддалену від обласних центрів та великих міст. Щоб розробити простий для сприйняття та приємний для користування інтернет – магазин, треба провести певні дослідження і на основі зроблених висновків визначитися з найнеобхіднішим функціоналом для сайту.

Дана тема дипломної роботи є актуальна, так як із збільшенням інтернет-магазинів можна збільшити кількість потенційних читачів, і найголовніше, збільшити читацьку аудиторії в маленьких містечках та селах.

При розробці веб сайту, а в даному випадку інтернет-магазину, акцентувати увагу треба на функціоналі та зовнішньому вигляді. Інтернет магазин з продажу книг повинен мати не тільки сторінку з довільним каталогом книг, даний веб ресурс повинен містити куди ширший функціонал, такий як зручна навігація по сайті, сортування книг за категоріїями, зворотній зв'язок з адміністрацією сайту у разі виникнення певних питань у клієнта.

Об'єктом дослідження є інтернет магазин з продажу книг.

Задачі досліджень. До теми дипломної роботи були поставлені такі задачі:

1. Дослідження та аналіз існуючих аналогів;
2. Визначення основних характеристик і структури онлайн магазину;
3. Розробка дизайну сайту.

Методи досліджень. Були використані такі технології для реалізації веб ресурсу:

1. Система управлінні вмістом WordPress;
2. Мова програмування PHP;
3. Система управління базою даних MySQL.

Також, в ході розробки інтернет-магазину була використана наукова література для реалізації продукту.

Результати досліджень. Результатом дипломної роботи є готовий інтернет магазин з продажу книг.

1 АКТУАЛЬНІСТЬ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА АНАЛІЗ ІСНУЮЧИХ АНАЛОГІВ ВЕБ ДОДАТКІВ

1.1 Актуальність розробки

Відомо, що читання книг дозволяє позитивно вплинути на якість життя людини, а саме:

1. Розвивається рівень емпатії. Згідно із деякими дослідженнями, художня література допомагає людям розвивати почуття співпереживання. Більшість розповідей, телевізійних шоу і навіть деякі відеоігри підвищують рівень співчуття до інших людей, роблять їх більш чуйними.
2. Розвивається гнучкість мислення. Твір, який змушує читача думати глибше, може підвищити гнучкість мислення. Віршовані тексти містять величезну кількість метафор і алегорій. Це змушує читачів бути більш уважними до деталей і сенсів, які автор вкладає в певні слова.
3. Підвищення рівня креативності. Процес читання часто пов’язують з підвищенням рівня креативності. Вважається, що люди, які читають художню літературу регулярно, менш консервативні і не відчувають себе ніяково через двозначність. Вони більш творчі і вільні.
4. Покращення роботи мозку. Під час одного із досліджень вчені просканували мозок учасників експерименту до, під час та через 5 днів після прочитання роману, і помітили певні неврологічні зміни.
5. Читання допомагає боротись із упередженнями. Вивчаючи світ інших людей, ми здатні позбутись деяких упереджень.
6. Сприяє профілактиці старечої деменції. Читання стимулює роботу мозку, допомагаючи йому залишатися в гарній формі аж до похилого віку. Якщо читати досить регулярно протягом усього життя, то є

ймовірність знизити ризик втрати пам’яті і частково запобігти зниженню концентрації уваги.

7. Читання збільшує тривалість життя. Читання книг продовжує життя. Згідно з останніми дослідженнями, тривалість життя людей, які захоплюються читанням, в середньому на два-три роки більша, ніж у тих, хто нічого не читає. Тому, щоб прожити довше, потрібно виділяти на читання приблизно 30 хв в день [1].

Найновіше масштабне дослідження читацької аудиторії України представили в Києві. Його провели Центр Разумкова, культурно-видавничий проект «Читомо» та платформа Підручник.UA. Опитування проводили в усіх областях України і Києві наприкінці жовтня 2018 року. Дослідники проаналізували 2013 анкет від мешканців 65 міст та 57 сіл.

Найпопулярнішим заняттям на дозвіллі є перегляд телебачення. Друге і третє місця поділяють читання преси та активність у соцмережах. Друковані книжки щодня читають лише 8 %, кілька разів на тиждень – 16 %. Електронні чи аудіокнижки – відповідно 4 % і 11 %. Ніколи не читають друкованих книжок трохи більше третини українців.

Електронні книжки в нас найчастіше читають зі смартфона. Більшість завантажує їх в інтернеті безкоштовно, лише 8 % купують разово на спеціалізованих сайтах і 3 % користуюся онлайн-передплатою.

Кількість прочитаних українцями книг в рік:

1. 37 % не прочитали за останній рік жодної друкованої книжки;
2. 36 % прочитали від 1 до 5 книжок;
3. 14 % – від 6 до 10 книжок;
4. 6 % – від 11 до 25 книжок;
5. 4 % – від 26 до 50 книжок;
6. 2 % – більш ніж 50 книжок.

Головна причина, чому українці не читають книжок – немає потреби чи бажання. Лише десята частина опитаних такою причиною називає ціни, ще менше – брак хороших, цікавих, якісних і корисних книжок.

На рис 1.1 можна переглянути графік, наскільки активно читають українці.

Рисунок 1.1 – Графік опитування, про те як часто українці читають книги

Судячи по графіку можна сказати, що багато українців досі не прочитали жодної книги. Але порівняно з минулим опитування яке було проведене в 2013 році, активно збільшується кількість людей які починають читати.

Оскільки останнім часом українці починають активно цікавитися літературою постає потреба у створенні великої кількості сайтів каталогів з продажу та ознайомленням з книжками [2].

Сучасні технології стрімко входять в усі сфери діяльності людини. Не вийнятком також є сфера продажу. Ще декілька років назад, купити книги, підручники, журнали можна було тільки в спеціалізованих крамницях.Так як з кожним прийдешнім роком, бажання українців читати неспинно росте, виникає потреба збільшити продаж книг на різні тематики. Взявши до уваги те що, розробка web-сайту є справа не затратна, це дає нам та іншим книгарням змогу вийти на новий ринок де можна конкурувати. В інтернет магазині дуже просто можна пропонувати книги на любий смак.

Даний інтернет магазин призначений для:

1. Зацікавити людей купляти паперові книги;
2. Підняти рівень людей в Україні які читають книги;
3. Зробити людей більш креативними.

1.2 Аналіз існуючих аналогів

В даному підрозділі буде проведений аналіз існуючих інтернет магазинів на їхні переваги та недоліки, та зроблені відповідні висновки.

1.2.1 Огляд сайту «Yakaboo»

Сайт має досить яскравий і привабливий дизайн, який дає можливість користувачеві сконцентруватися на основному моменті – Каталозі книг.

Головна сторінка сайту дуже інформативна, на ній представлений блок, що закликає людей перейти до каталогу книг. На сайті представлені посилання на соціальні мережі, через які користувач може поділитися інформацією, яка його цікавить [3].

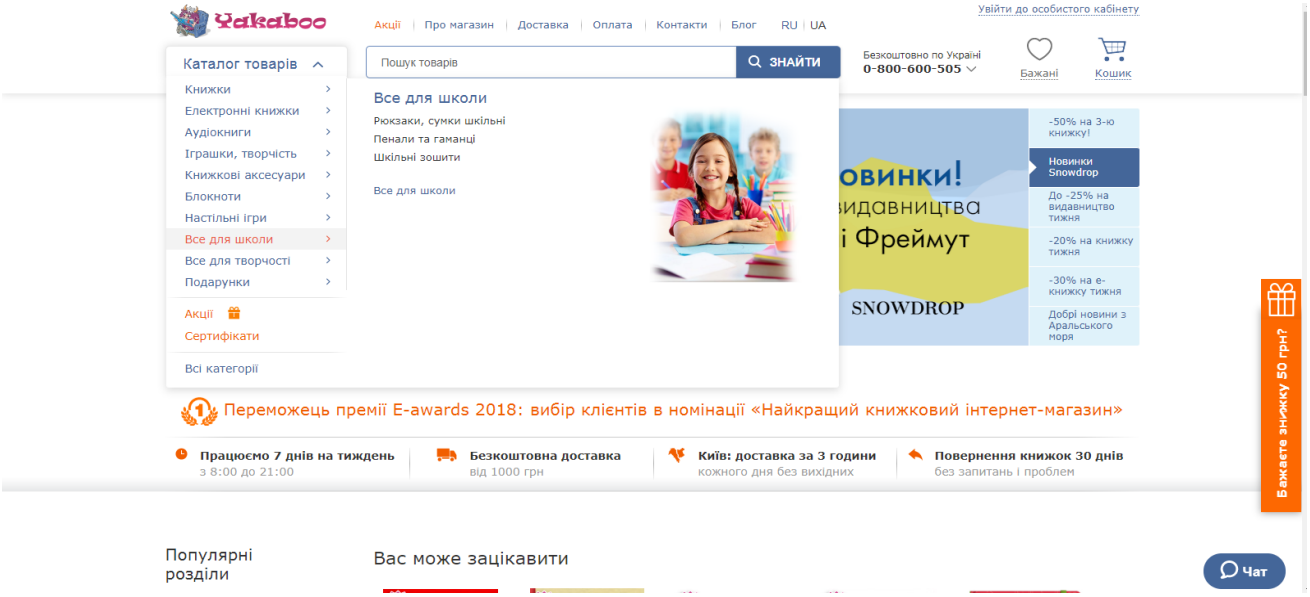

Рисунок 1.2 – Головна сторінка сайту «Yakaboo»

Структура даного сайту є така:

1. Акції – дана сторінка призначена для огляду та купівлі акційного товару;
2. Про магазин – інформація про даний інтернет магазин та його команду;
3. Доставка – умови доставки товару по території України;
4. Оплата – опис способів оплати в даному інтернет магазині;
5. Контакти – тут показані номери телефонів адміністраторів та місце розташування магазину книг;
6. Блог – на цій сторінці відображаються різні блоги про безліч жанрів книг;
7. Каталог товарів – випадаюче меню з великою кількістю товарів.
8. Пошук – пошук книг по сайті;
9. Корзина – сторінка де буде оформлюватися замовлення на обраний товар;
10. Бажані – сторінка для відображення вподобаного товару.

1.2.2 Огляд сайту «Книгаренька»

Сайт має не привабливий дизайн, що не закликає користувача перейти до пошуку та безпосереднього придбання книг. Структура даного порталу не дозволяє відвідувачеві швидко орієнтуватися в представленій інформації. Блоки сайту дуже великі і по цій причині дуже важко зорієнтуватися що вони означають. При натисканні на любий із блоків на головній сторінці, сайт просто оновлює головну сторінку та нікуди вас не перенаправляє [4].

Оцінити структуру сайту книжкового онлайн магазину можна за допомогою рис. 1.3, 1.4.

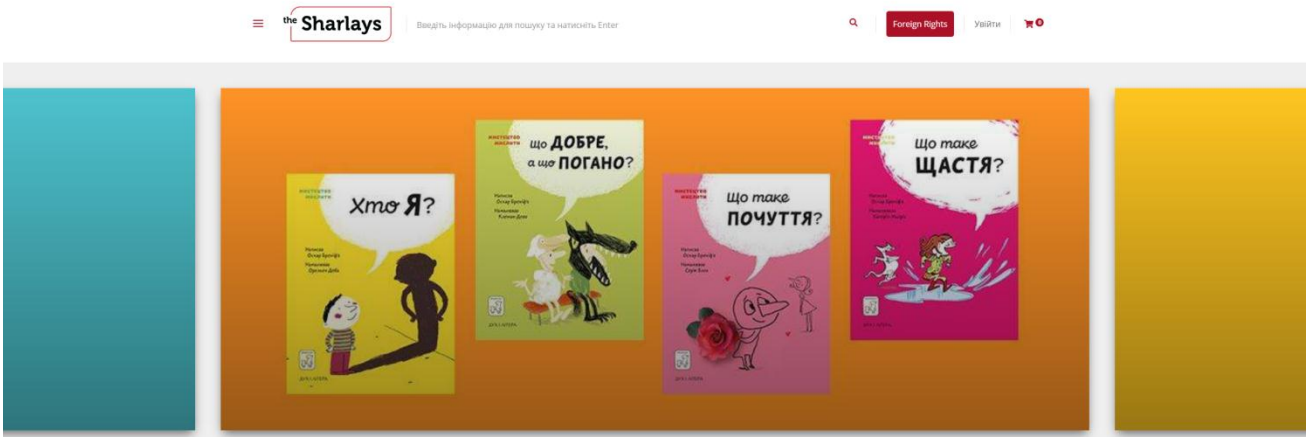

Рисунок 1.3 – Головна сторінка сайту «Книгаренька»

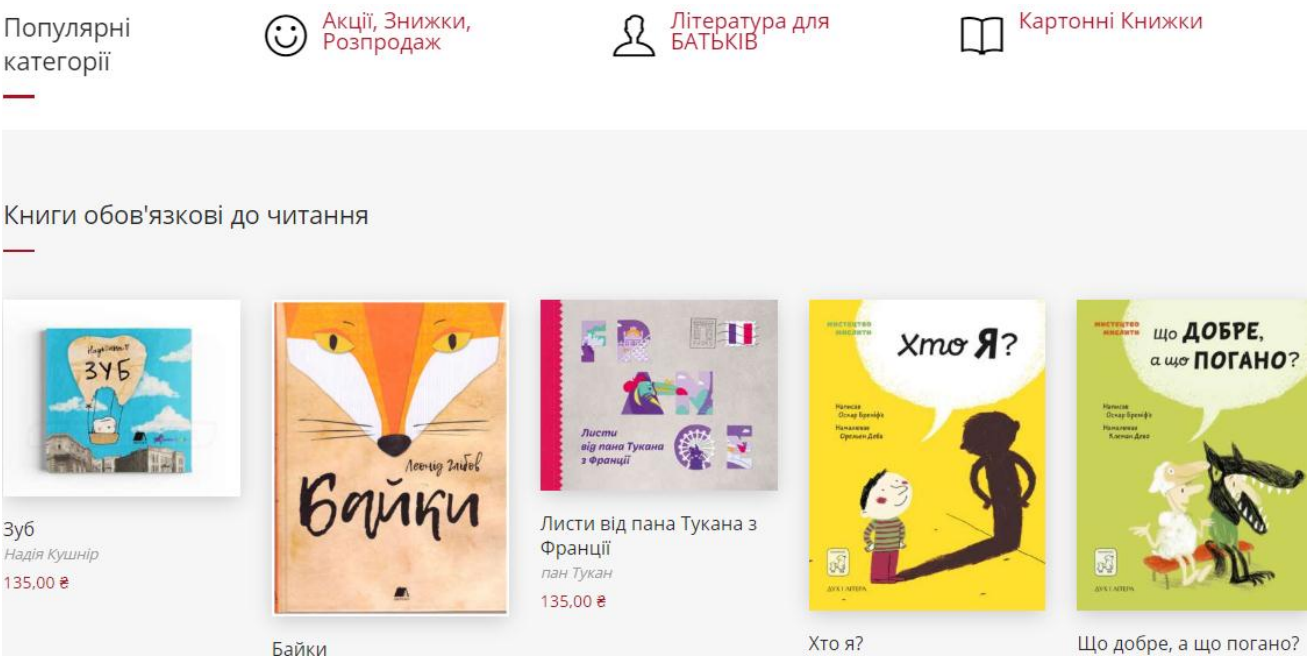

Рисунок 1.4 – Головна сторінка сайту «Книгаренька»

Структура сайту:

Головна сторінка – на головній сторінці представленавлена інформація про книги які обов’язкові до читання, інформація про новинки та найкращі книги.

Акції, Знижки, Розпродаж – даний розділ містить товари які зараз перебувають в знижках, та блок з текстом як придбати книги.

Література для батьків – цей розділ містить в собі каталог з книгами, які рекомендуються для прочитання батькам, та блок з інформацією чому саме ця література призначена для батьків.

Картонні книги – розділ містить в собі картонні книги з казками, оповіданнями та віршами. Також в даному розділі знаходяться книги для розвитку дітей з ребусами та лабіринтами.

1.2.3 Огляд сайту «Книгарня Нора-Друк»

Структура і дизайн сайту не дуже вдалі – через надмірну кількість червоного кольору сайту, користувачеві важко орієнтуватися в розділах, основні заголовки і блоки ніяк не виділені, що ускладнює роботу з пошуком інформації на даному сайті. Система пошуку та випадаюче меню категорій на англійській мові, що може бути не зрозумілим для більшості користувачів.

Кнопка «Кошик» також непереведена на українську мову. На сайті з права є бічне меню «Новини», це є стандартним блоком CMS WordPress [5].

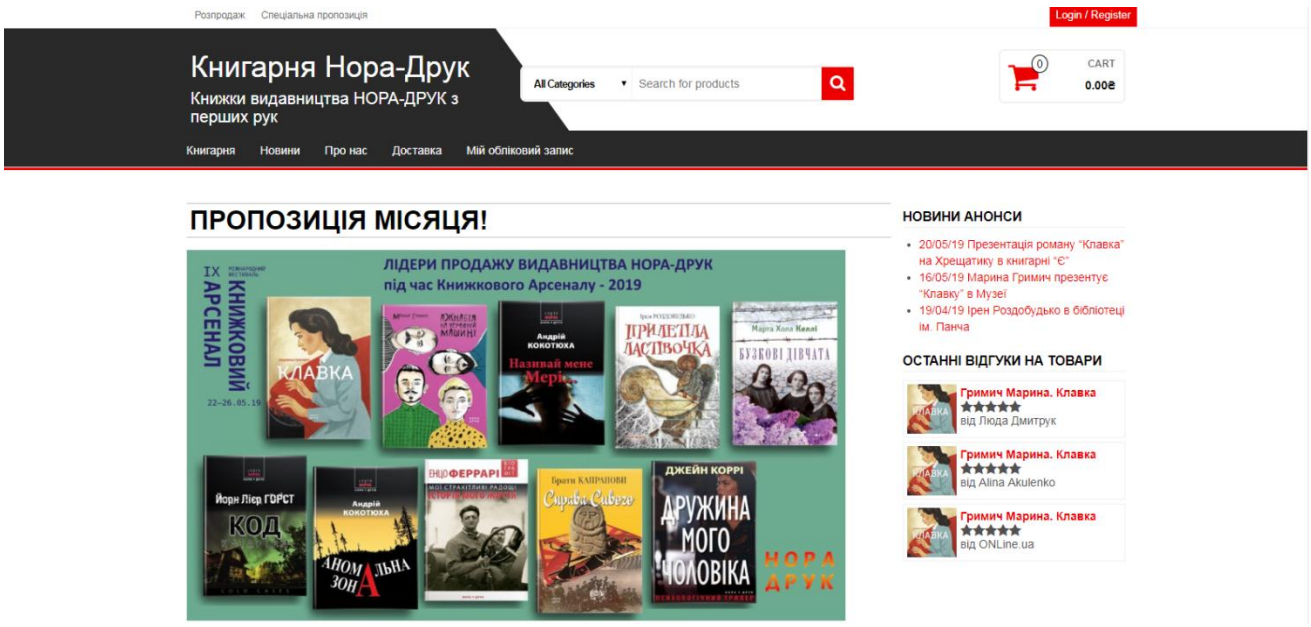

Рисунок 1.5 – Головна сторінка сайту «Книгарня Нора – Друк»

Структура даного сайту є така:

1. Головна – на головній сторінці представлений блок з відгуками на товар, блок новин, та статичну картинку з пропозиціями місяця.

2. Книгарня – на даній сторінці відображаються абсолютно всі книги які є в наявності інтернет магазину «Книгарня Нора – Друк».
3. Новини – на даній сторінці відображається блог з останніми новинами сайту.
4. Про нас – тут відображається короткий опис про сайт, та контактні дані адміністрації сайту.
5. Доставка – на цій сторінці вказані умови доставки книг по Україні.
6. Мій обліковий запис – тут можна зареєструватись на даному сайті або залогінитись при умові, що аккаунт уже існує.

1.3 Постановка завдання

Завданням дипломного проекту є створення інтернет магазину з продажу книг, за допомогою якого можна вибрати книгу потрібного жанру та придбати її [6].

За допомогою даного інтернет магазину люди зможуть:

1. Ознайомитись з особливостями сайту;
2. Обрати книгу;
3. Оформити замовлення.

Структура такої веб-орієнтованої системи повинна мати такий вигляд:

Головна сторінка. Повинна бути оформлена у вигляді Landing Page для залучення уваги аудиторії і призову перейти до каталогу. На ній будуть розташовуватися блоки із видавцями книг, найкращих користувачів магазину, блок із посиланням на «Twitter» та формою зворотнього зв’язку.

Про нас – на даній сторінці можна буде прочитати інформацію про те з якою метою був створений даний онлайн магазин.

Каталог книг – даний розділ буде мати випадаюче меню з категоріями книг. Обравши жанр, покупця перенаправляє на сторінку за обраною ним категорією. Там знаходяться такі категорії:

1. Жахи

2. Фантастика

3. Українська література

4. Містика

5. Фентезі

Корзина – тут можна буде оформити замовлення книги будь-якого жанру обраної покупцем в каталозі книг.

Для вирішення таких завдань потрібно:

1. Провести аналіз деяких існуючих сайтів книгарень;

2. Розглянути сучасні засоби розробки web-сайтів;

3. Виявити основні переваги та недоліки систем управління контентом;

4. Вибрати систему управління базами даних;

5. Розробити структуру майбутньої системи;

6. Зробити програмну реалізацію системи.

2 МОДЕЛЮВАННЯ ВЕБ РЕСУРСУ ТА АНАЛІЗ ЗАСОБІВ РОЗРОБКИ

2.1 Діаграма прецедентів

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів, обмежених границею системи, асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання застосовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором. При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

1. Асоціації;
2. Включення;
3. Розташування;
4. Узагальнення.

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності є таке відношення між двома елементами моделі, при якому зміна одного елемента приводить до зміни іншого елемента.

Відношення розширення визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється з базовим не завжди, а тільки при виконанні додаткових умов.

Діаграма прецедентів представлена на рис 2.4.

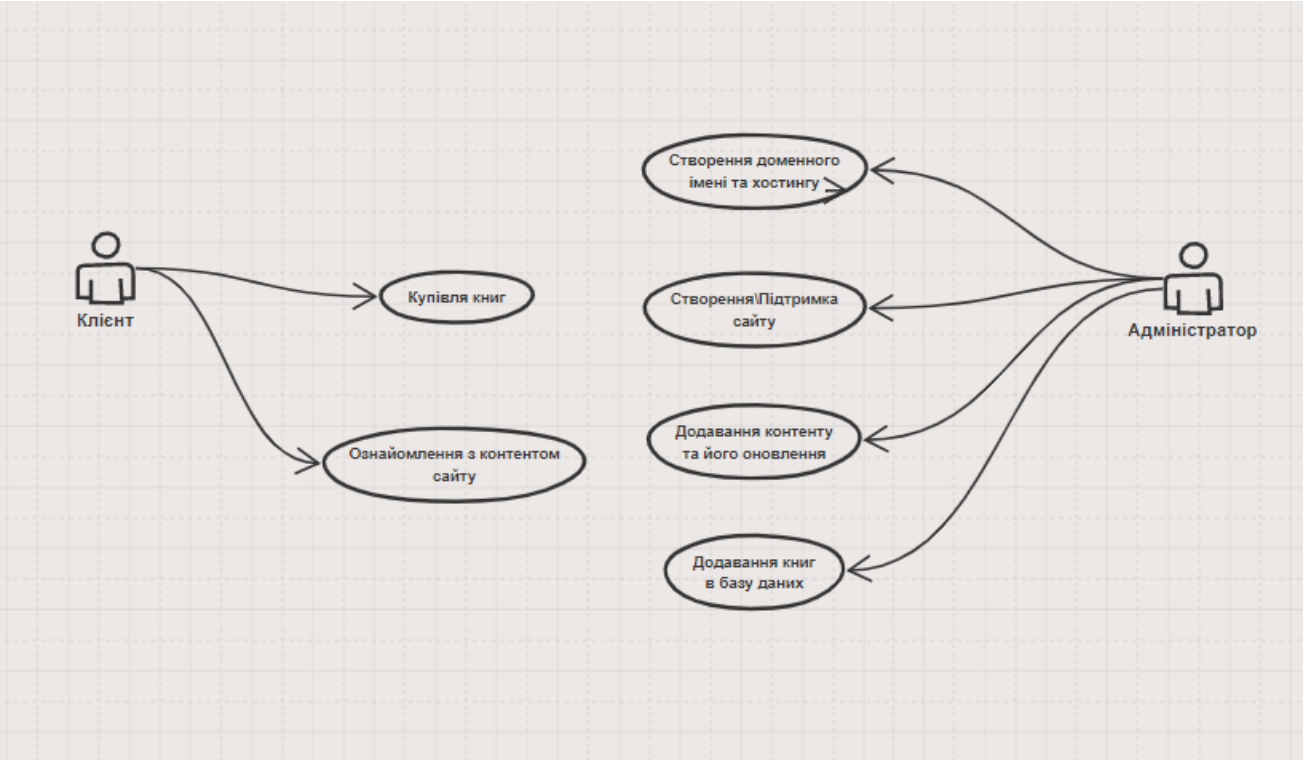

Рисунок 2.4 – Діаграма прецедентів

Частину інформації в моделі прецедентів можна усунути зазначенням зв'язків між прецедентами:

1. Узагальнення прецеденту – стрілка з не закрашеним трикутником (трикутник ставиться у більш загального прецеденту);
2. Включення прецеденту – пунктирна стрілка зі стереотипом «include»;
3. Розширення прецеденту – пунктирна стрілка зі стреотипом «extend» (стрілка входить в розширюваний прецедент, в додатковому розділі

якого може бути вказана точка розширення і, можливо у вигляді коментаря, умова розширення).

2.2 Діаграма класів

Моделювання сутностей програмної системи проводять за допомогою схеми класів UML. Схема класів, створювана на початкових етапах проектування програмного продукту, дозволяє попередньо проаналізувати сутності програмної системи та їх зв'язку на досить високому рівні абстракції.

Інакше кажучи, діаграма класу може відображати лише імена класів або імена та відповідні атрибути класів, або імена, атрибути та операції класів. Класи можуть перебувати у певних відношеннях або зв’язках [7].

Клас представляє множину об'єктів, яку об'єднує спільна структура та поведінка. Клас є абстракцією сутностей реального світу. Реальні екземпляри сутностей (instances) це так звані об'єкти.

Атрибути класу визначають склад і структуру даних, що зберігаються в об’єктах цього класу. Кожен атрибут має ім’я і тип, який визначає, які дані він представляє.

При реалізації об’єкта в програмному коді для атрибутів буде виділена пам’ять, необхідна для зберігання всіх атрибутів, і кожен атрибут матиме конкретне значення в будь-який момент часу роботи програми. Об’єктів одного класу в програмі може бути як завгодно багато, всі вони мають однаковий набір атрибутів описаний в класі, але значення атрибутів у кожного об’єкта свої і можуть змінюватися в ході виконання програми.

Діаграма класів відображає статичну структуру системи у термінах класів та відношень між цими класами.

Є такі відношення:

Асоціація – взаємна залежність між об’єктами різних класів, кожен з яких є рівноправним членом залежності. Для асоціації може позначатися кількість

екземплярів об’єктів кожного класу, які беруть участь у зв’язку (0 - якщо жодного, 1 – якщо один, * – якщо багато).

Можуть вказуватися мінімальна і максимальна кількість, наприклад, 0,1...* означає, що на відповідному кінці асоціації може не бути жодного екземпляра, бути один або багато.

Агрегація або відношення частин до цілого. Особливість цього відношення полягає в тому, що час існування об’єкта частини збігається з часом існування об’єкта цілого. Стрілка з ромбом на кінці, яка позначає відношення агрегації, спрямована від об’єкта частини до об’єкта-цілого (рис. 2.5).

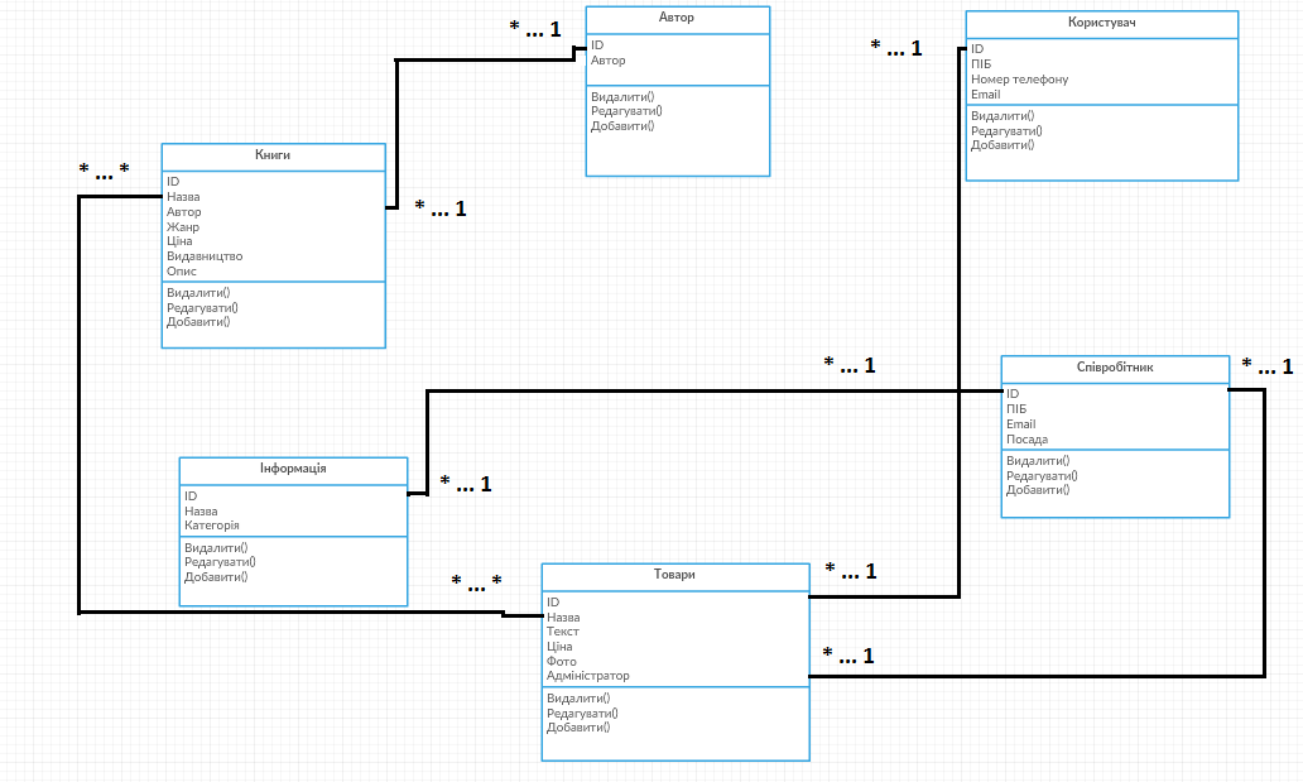

Рисунок 2.5 – Діаграма класів

На діаграмі класів показані основні сутності і зв'язку між ними. У нашому випадку, основними сутностями є:

1. Автор
2. Інформація
3. Книги
4. Товари

5. Користувач
6. Співробітник

2.3 Контекстна діаграма

Побудова контекстної діаграми починається з отримання всієї системи у вигляді одного блоку і граничних стрілок, що відображають зв’язки з функціями поза системою.

Діаграма, що складається з одного блоку і стрілок, визначає межі системи і називається контекстною діаграмою моделі. При цьому блок відображає кордони системи: все, що лежить всередині нього, є частиною системи, а все, що лежить поза ним, утворює зовнішнє середовище системи.

Стрілки на контекстній діаграмі служать для опису взаємодії системи з навколишнім світом. Такі стрілки називаються граничними. Вони можуть починатися біля кордону діаграми і закінчуватися біля роботи, або навпаки, починатися біля роботи і закінчуватися біля кордону діаграми.

Розрізняють п’ять типів граничних стрілок, що зв’'язують систему із зовнішнім світом:

1. Входи – матеріали або інформація, які використовуються або перетворюються роботою для отримання результату (стрілка, що входить в лівий кордон прямокутника). Допускається, що робота може не мати жодної стрілки входу.
2. Управління – правила, стратегії, стандарти, якими керується робота (стрілка, що входить в верхню межу прямокутника). На відміну від вхідної інформації управління не підлягає зміні в результаті виконання роботи. Кожна робота повинна мати хоча б одну стрілку управління.
3. Виходи – матеріали або інформація, які виробляються роботою (стрілка, яка виходить із правої межі прямокутника). Кожна робота

повинна мати хоча б один вихід, тому що робота без результату не має сенсу і не повинна моделюватися.

4. Механізми – ресурси, які дозволяють виконати роботу: персонал, верстати, аналітика стрілки механізму можуть не зображатися в моделі.
5. Виклик – стрілка, що вказує на іншу модель роботи (стрілка, яка виходить із нижньої межі прямокутника). Стрілка виклику використовується для вказівки того, що деяка робота виконується за межами модельованої системи.

В даному випадку використовується діаграма IDEF0 (рис. 2.6).

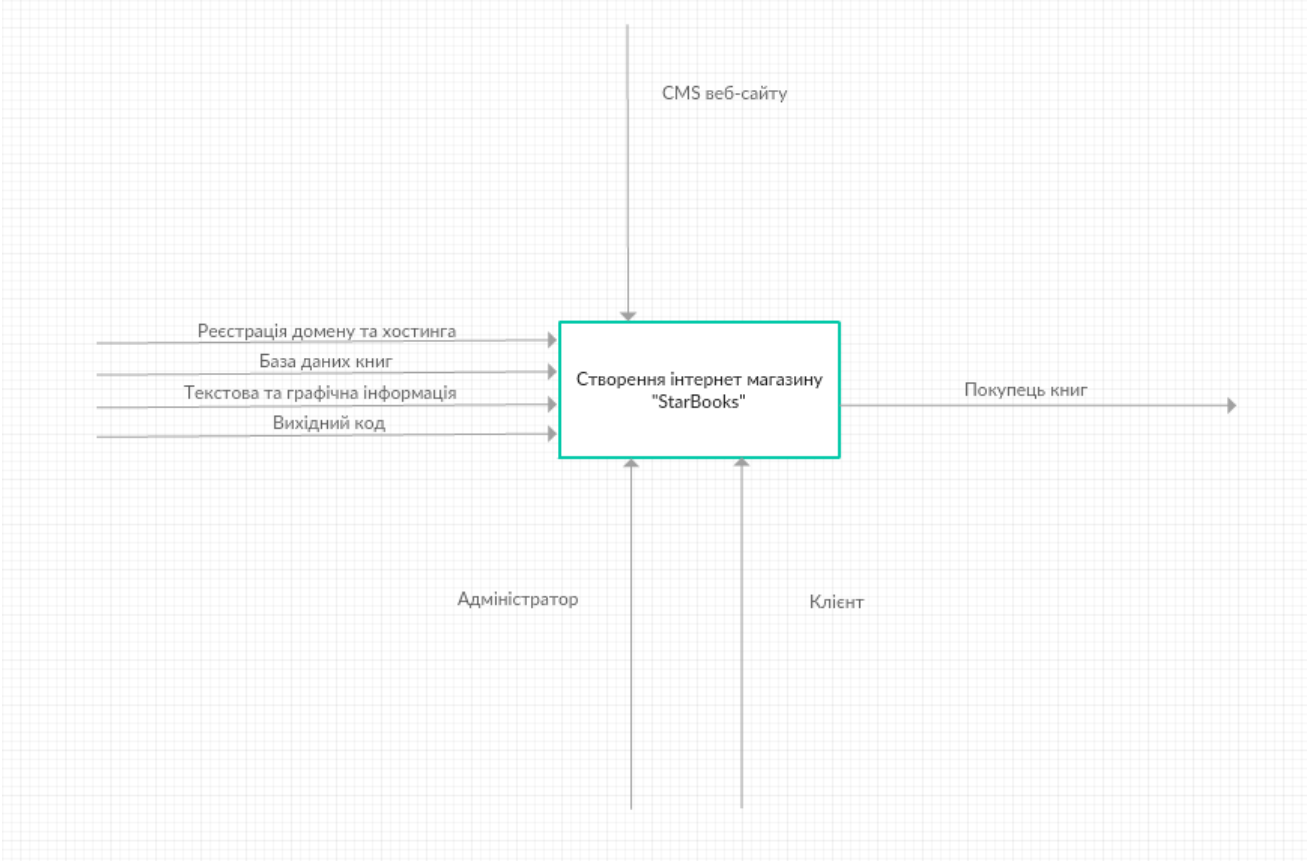

Рисунок 2.6 – Контекстна діаграма

Опис системи за допомогою IDEF0 називається функціональною моделлю.

Функціональна модель призначена для опису існуючих бізнес-процесів, в яких використовуються як природні, так і графічні мови. Для передачі інформації про конкретну систему джерелом графічної мови є сама методологія IDEF0. Функціональна модель являє з необхідним ступенем деталізації систему

функцій, які відображають свої взаємини через об'єкти системи. Моделі даних є докладним описом об'єктів системи, пов'язаних з системними функціями. Повна методологія SADT підтримує створення безлічі моделей для більш точного опису складної системи.

Кожна IDEF0 діаграма містить блоки і дуги. Блоки відображають функції модельованої системи. Дуги пов’язують блоки разом і відображають взаємодії і взаємозв’язки між ними [8].

На контекстної діаграмі показаний основний процес у вигляді створення інтернет магазину.

На вхід процесу надходять необхідні для створення сайту дані, тобто: текстова та графічна інформація, база даних, доменне ім’я і хостинг.

Виходами є реузульта, тобто досягнутий результат у вигляді покупки клієнтом книги.

2.4 Діаграма декомпозиції

Для головного процесу, присутнього на контекстній діаграмі, проводиться декомпозиція. На першому рівні ієрархії показуються основні внутрішні процеси системи і відповідні їм зовнішні сутності, сховища і потоки даних.

Для кожного процесу діаграми першого рівня може бути проведена декомпозиція, яка, в свою чергу, також може бути розкрита більш докладно. Декомпозиція процесів закінчується, коли досягнута необхідна ступінь деталізації або відображаються на черговому рівні діаграм процеси є елементарними і не можуть бути розбиті на більш дрібні.

При проведенні декомпозиції повинно виконуватися правило балансування – при декомпозиції процесу дочірня діаграма в якості зовнішніх сутностей може мати тільки ті об'єкти (процеси, зовнішні сутності, сховища даних), з якими має інформаційний зв'язок деталізований процес на батьківській діаграмі.

Роботи ніколи не розміщуються на діаграмі випадковим чином. Вони розміщуються за ступенем важливості, як її розуміє автор. У методології SADT цей відносний порядок називається домінуванням. Домінування розуміється як вплив, яке одна робота надає на інші роботи діаграми.

Самою домінуючою роботою діаграми може бути найважливіша з необхідної послідовності робіт, або яка планує або контролююча робота, що впливає на всі інші роботи.

Найбільш домінуюча робота, зазвичай, розміщується в верхньому лівому кутку діаграми, а найменш домінуюча – в правому нижньому кутку. В результаті виходить ступінчаста схема.

Діаграма декомпозиції призначена для деталізації роботи. Роботи нижнього рівня – це те ж саме, що робота верхнього рівня, але в більш детальному викладі. Як наслідок цього кордону роботи верхнього рівня – це те ж саме, що кордони діаграми декомпозиції [9].

При декомпозиції будь-якої роботи стрілки які входять в неї і виходять з неї повинні відображатися на діаграмі декомпозиції, тобто повинна відбутися міграція стрілок.

У попередньому підрозділі дипломної роботи була побудована контекстна діаграма, що містить тільки одну роботу, яка описує діяльність в цілому, без деталізації складових цієї роботи. В даному підрозділі буде побудована діаграми декомпозиції першого рівня в нотації IDEF0.

Декомпозиція – це поділ складного об'єкта, системи, завдання на складові частини, елементи.

За допомогою діаграми декомпозиції першого рівня покажемо, з яких робіт складається робота над сайтом «Книгарня StarBooks» (рис 2.7).

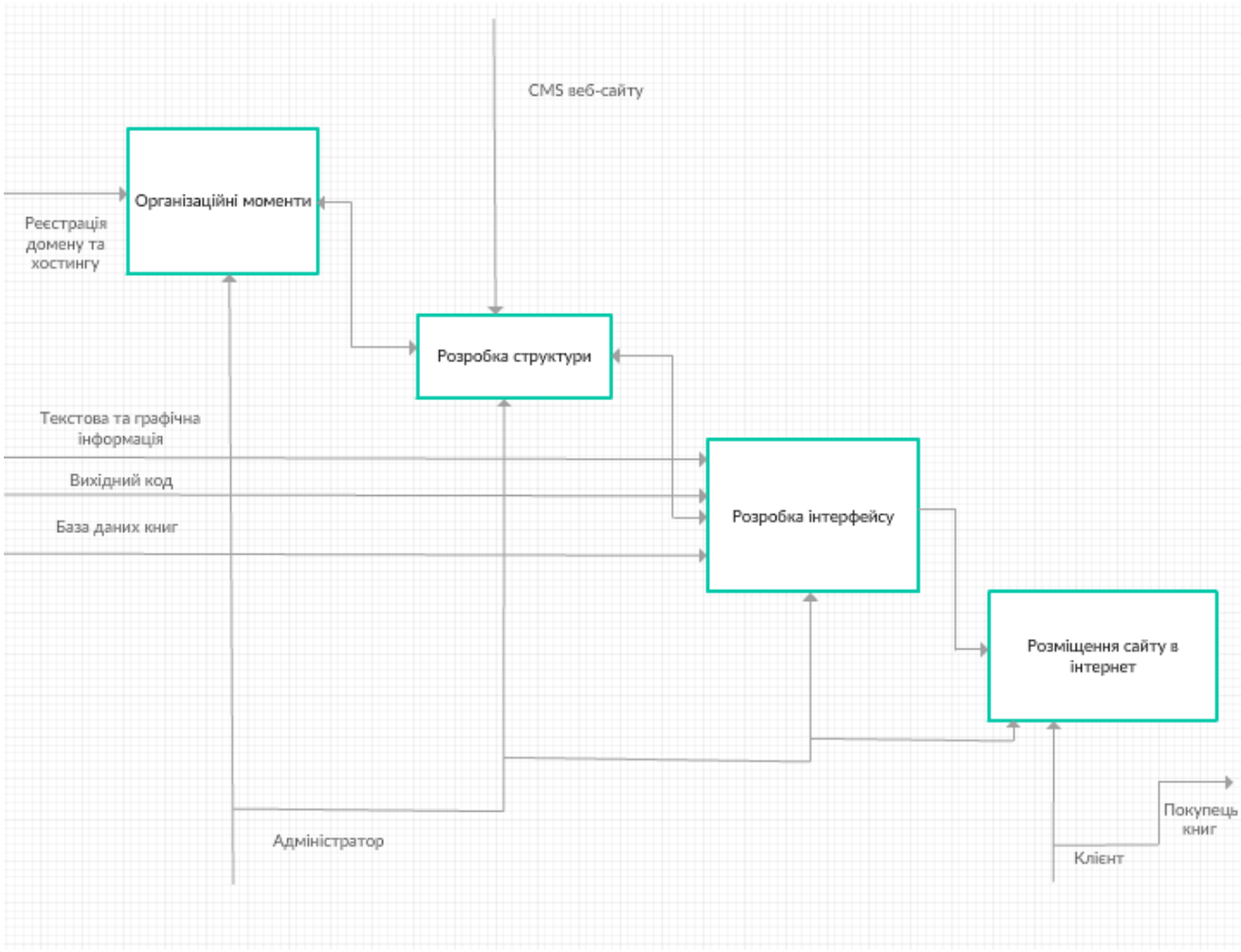

Рисунок 2.7 Діаграма декомпозиції

2.5 Аналіз функціональних засобів для реалізації системи

При виборі системи управління сайтом CMS потрібно враховувати кілька факторів, а саме:

1. Вартість даної системи;
2. Можливої функціональності має вистачити на реалізацію програмного забезпечення.

На сьогоднішній день існує безліч систем управління контентом як платних, так і безкоштовних, з досить великим функціоналом. Розглянемо основні з них: Joomla, DataLife Engine, WordPress.

2.5.1 Опис CMS Joomla

Joomla – безкоштовна система для створення веб-сайтів. Це проект з відкритим вихідним кодом, який, як і більшість подібних проектів, не стоїть на місці. Він дуже успішно розвивається, протягом уже семи років, і користується популярністю у мільйонів користувачів по всьому світу(рис 2.7).

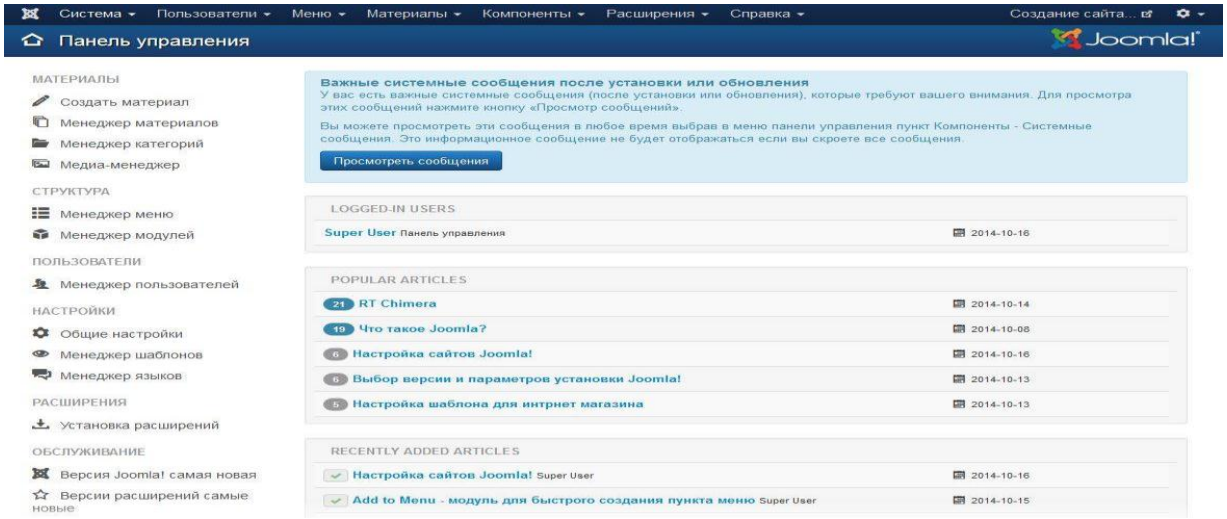

Рисунок 2.8 – Панель управління CMS Joomla

Переваги використання Joomla як CMS для сайту.

Безкоштовність і відкритий вихідний код. Мабуть, для багатьох користувачів саме ціна системи відіграє вирішальну роль при виборі того чи іншого продукту. Звичайно, як і у випадку з іншими системами, у вас може виникнути необхідність витратити деяку суму на додаткові компоненти, а також, послуги програміста і дизайнера при персоналізації сайту, але спочатку система поширюється вільно, і кожен може вносити в її код зміни за власним бажанням.

Орієнтованість на дизайнерів і розробників. Завдяки надзвичайно простої модифікації зовнішнього вигляду сайту, ви можете всього в кілька кліків змінити зовнішній вигляд сайту до невпізнання. Це ще одна чудова можливість максимально персоналізувати свій сайт, підлаштувати його зовнішній вигляд під ваші потреби [10, 11].

Наявність безлічі додаткових компонентів, розширень, плагінів, інструментів. Така ситуація стала майже стандартною для будь-якого популярного продукту з відкритим вихідним кодом. Завдяки тисячам додаткових компонентів ви зможете вбудувати в свій сайт практично що завгодно.

Є привабливість готового продукту. Навіть якщо у вас невеликий досвід роботи з сайтами, на виході ви отримаєте сайт з професійним виглядом. Це великий плюс для тих, хто вирішив створити комерційний продукт, який повинен виглядати солідно для користувача.

Постійні оновлення і підвищення якості та безпеки системи. Спільнота Joomla є однією з найбільших. Серед шанувальників цієї CMS величезна кількість розробників, які постійно створюють нові розширення, а також знаходять недоробки і способи поліпшення роботи системи. В результаті кінцевий користувач отримує все більш і більш якісну систему.

Універсальність. Як і у випадку з Drupal, можливості Joomla дозволяють створити сайт будь-якої складності і змісту з мінімальним використанням додаткових інструментів і розширень [12].

Недоліки використання Joomla як CMS для сайту:

Одним з недоліків Joomla можна назвати труднощі при переході на більш нову версію, коли можливе порушення деяких встановлених компонентів, причому можливо тільки послідовне оновлення через проміжні версії;

Для ранніх версій Joomla не є актуальна і складність самих оновлень. На щастя, в останній версії реалізований автоматичний пошук і установка оновлень;

Joomla лідирує в числі систем-улюбленців зломщиків. Сайти на цій системі досить часто піддаються атакам. Не останньою причиною тут є і недостатньо висока захищеність системи, але ситуація постійно покращується завдяки оперативній роботі спільноти Joomla;

Складність освоєння. Доводиться визнати, що функціональність і універсальність завжди вимагають деяких зусиль від користувача для їх освоєння;

Іншим загальним недоліком, властивим для всіх безкоштовних систем, є відсутність служби підтримки, але зате користувач отримує сотні навчальних матеріалів і посібників, а також десятки майданчиків (форуми і т.д.) для обміну досвідом та вирішення проблем які виникли.

2.5.2 Огляд CMS DataLife Engine

DataLife Engine – це багатофункціональна система управління сайтом. Завдяки дуже потужній системі управління новинами, публікаціями, статтями, користувачами, DataLife Engine призначений в першу чергу для організації власних засобів масової інформації та блогів в мережі інтернет. Велика розмаїтність базових можливостей скрипта, зразу інтегрованих в скрипт, дозволяють DataLife Engine на крок випереджати своїх найближчих конкурентів у даному напрямку створення сайтів. А завдяки точній і продуманій структурі ядра зводяться до мінімуму вимоги до ресурсів сервера, DataLife Engine здатний з легкістю працювати з високовідвідуваними проектами с мінімальним навантаженням на сервер.

Гнучкість і легкість в налаштуванні і інтеграції власного дизайну, дозволить вам в найкоротші терміни вивчити і зрозуміти структуру шаблонів і відповідно розгорнути власні сайти, без будь-яких додаткових матеріальних витрат.

Використання просунутої технології AJAX скоротить як трафік і ресурси вашого сервера, так і трафік ваших відвідувачів, не кажучи вже про зручність використання даної технології на сайті для відвідувачів. Завдяки підвищеній увазі до безпеки скрипта, до зручності роботи з ним, а також динамічному розвитку, на користь використання DataLife Engine вже висловилися більше 110 000 порталів по всьому світу і успішно використовують його в роботі.

Велика увага в скрипті приділено оптимізації під пошукові системи, що дозволяє привести на ваш сайт додаткових відвідувачів [13].

Також як і інший популярний двигун, DLE написаний на PHP і в якості СУБД використовує MySQL. DLE CMS має потужну систему публікацій, настройки та редагування новин, ставить мінімальні вимоги до ресурсів сервера, дозволяє легко працювати з високовідвідуваними проектами (рис. 2.9).

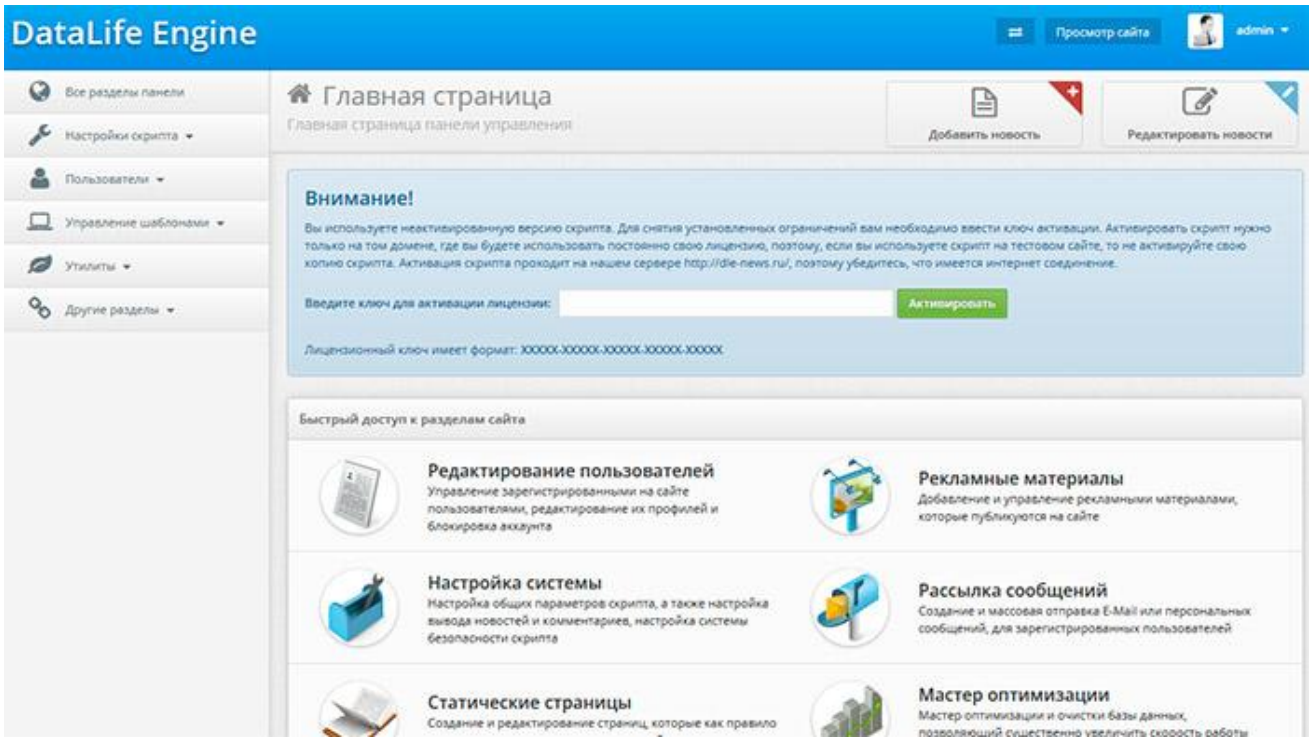

Рисунок 2.9 – Панель управління CMS DataLife Engine

Система управління контентом DLE для відвідувачів сайту надає наступні можливості:

1. Реєстрація на сайті;
2. Додавання коментарів;
3. Зміна та видалення користувачами своїх власних коментарів;
4. Можливість додавання новин з боку відвідувачів;
5. Модерація новин;
6. Можливість завантаження фотографії в профілі користувача;
7. Можливість відновлення забутого пароля;
8. Можливість редагування новини безпосередньо на сайті.

Переваги використання DLE як CMS для сайту:

1. За допомогою численних модулів вона модифікується в каталог або галерею. Відмінні риси двигуна: швидкість, простота і зручність використання [14];
2. Навчитися керувати цим скриптом легко. Адміністративна панель інтуїтивно зрозуміла і не вимагає особливих пояснень;
3. DLE може з легкістю працювати з відвідуваними проектами, не створюючи особливих навантажень на сервер, кількість запитів до бази даних є мінімальною.

Розробники передбачили ряд інструментів, які дозволяють налаштувати DLE під свої потреби, а саме:

1. Розділ «Аналіз продуктивності» в адмінці видає рекомендації щодо зміни і відключення тієї чи іншої можливості скрипта з зазначенням ступеня створюваного навантаження. Відмовившись від непотрібних можливостей можна спробувати зменшити навантаження, не переходячи на більш дорогий тариф на хостингу;
2. Розділ «Налагодження та оптимізація бази даних» - утиліта, здатна оптимізувати вашу базу даних, прискоривши її роботу і звільнивши місце на диску, а також зробити ремонт бази даних в разі пошкодження структури таблиць;
3. Розділ «Шаблони сайту» дозволить вам легко впроваджувати власний дизайн в найкоротші терміни. Для кожного запису і статичної сторінки можна задати свій шаблон. Висновок даних в шаблоні гнучко налаштовується (завдяки так званим тегам).

До недоліків скрипта часто відносять відсутність автоматичної установки модулів; сформовану порочну практику редагування файлів движка, чого в ідеалі не повинно бути через загрозу втрати змін при оновленні.

2.5.3 Огляд CMS WordPress

WordPress – це система створення та управління веб-сайтом.

WordPress – CMS з відкритим вихідним кодом, поширювана під GNU GPL. Написана на PHP, в якості бази даних використовує MySQL. Сфера застосування - від блогів до досить складних новинних ресурсів і навіть інтернет-магазинів. Вбудована система «тем» і «плагінів» разом з вдалою архітектурою дозволяє конструювати практично будь-які проекти.

WordPress є ідеальною платформою для публікацій, орієнтованих на естетику, підтримку стандартів і зручність використання. WordPress безкоштовний і вільний до розповсюдження (рис 2.10) [15, 16].

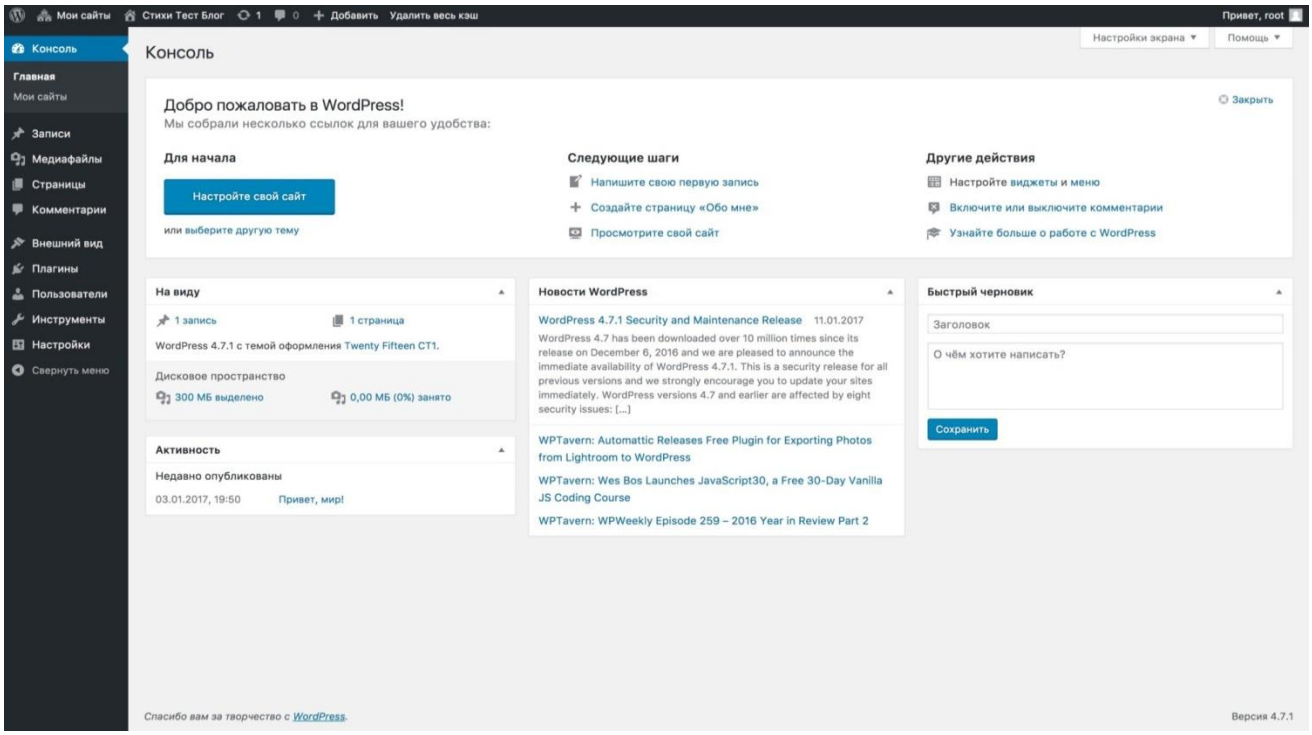

Рисунок 2.10 Панель управління CMS WordPress

Основні можливості WordPress:

1. Управління та адміністрування:
   1.1 WordPress розроблений для установки на персональний хостинг, що дає повний контроль над блогом, на відміну від безкоштовних блог-сервісів;

1.2 Управління зареєстрованими користувачами;

1.3 Особисті профілі користувачів;

1.4 Легке встановлення і оновлення. Встановлення займає всього 5 хвилин;

1.5 Динамічне створення сторінок, не має потреби перестворювати всі сторінки блогу при внесенні змін до нього;

1.6 Інтернаціоналізація та локалізація WordPress "говорить" вашою мовою.

2. Публікація:

2.1 RSS стрічки, публікація через протоколи RSS 1.0, RSS 2.0 і ATOM;

2.2 Постійні посилання.

3. Дизайн та розширення можливостей:

3.1 Заснований на шаблонах дизайн;

3.2 Шаблонові теги, можливість просто додати зміни в шаблони без особливих знань PHP;

3.3 Теми, які легко змінюють вид блогу, просто скачавши і встановивши нову тему до нього;

3.4 Модулі. Величезна кількість плагінів (додаткових підпрограм), що розширюють функціональність блогу, інтернет магазину і так далі.

4. Контент:

4.1 Захист паролем обраної інформації;

4.2 Запланована на певну дату публікація;

4.3 Багатосторінкові записи;

4.4 Завантаження файлів і зображень в запису.

5. Категорії:

5.1 Візуальний редактор;

5.2 Пошук по блогу;

5.3 Коментування.

Як CMS для розробки простого інтернет магазину, був обраний WordPress з кількох причин:

1. CMS WordPress абсолютно безкоштовний;
2. Велика кількість різних шаблонів;
3. Велика кількість плагінів;
4. Відкритий вихідний код;
5. Інтуїтивно зрозуміла адмінпанель;
6. Легка установка CMS на хостинг.

WordPress – це безкоштовна CMS, яка має багато переваг відносно інших систем. Наприклад, вона має дуже зручну адмінпанель, за допомогою якої можна зробити сайт будь-якої складності.

Крім того, для WordPress є величезна кількість безкоштовних шаблонів, які приємно дивують своєю якістю, різноманітність додаткових модулів під різні завдання. А велика спільнота завжди допоможе вирішити проблеми з налаштуваннями системи.

Для створення сайту вибрано CMS WordPress, так як є ряд плюсів які, а саме:

1. CMS WordPress абсолютно безкоштовний;
2. Велика кількість різних шаблонів;
3. Велика кількість плагінів;
4. Відкритий вихідний код;
5. Адмін панель легка в розумінні;
6. Легка установка;
7. Наявність офіційних програм для пристроїв Apple і Android.

Причини по яких не було вибрано Joomla і Datalife Engine:

1. Інтерфейси важчі у використанні;
2. Споживають більше ресурсів і потрібно купуляти більш дорогі і більш продуктивні хостинги і сервера.

2.5.4 Особливості скриптової мови PHP

PHP – скрипкова мова загального призначення, інтенсивно застосовується для розробки веб-додатків. В даний час підтримується переважною більшістю хостинг провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних веб-сайтів.

PHP має декілька недоліків, наприклад:

1. Не підходить для створення деяких додатків або системних компонентів;
2. Має слабкі засоби для роботи з вийнятками;
3. Глобальні параметри конфігурації впливають на базовий синтаксис мови, що ускладнює налаштування сервера і розгортання додатків;
4. Об’єкти передаються за значенням, що бентежить багатьох програмістів, які звикли до передачі об’єктів по посиланню, як це робиться в більшості інших мов;
5. Веб-додатки, написані на PHP, часто мають проблеми з безпекою.

Але незважаючи на ці недоліки, він має свої переваги, котрі відрізняють його від інших серверних мов програмування, і найбільш підходять під задачі саме web-додатків.

1. Є вільним програмним забезпеченням, поширюваним під особливою ліцензією (PHP ліцензії);
2. Легкий в освоєнні на всіх етапах;
3. Підтримується великою спільнотою користувачів і розробників;
4. Має розвинену підтримку баз даних;
5. Є величезна кількість бібліотек і розширень мови;
6. Може використовуватися в ізольованому середовищі;
7. Пропонує нативні кошти організації веб-сесії, програмний інтерфейс розширення;
8. Є досить повною заміною середовища ASP (Active Server Pages) від Microsoft;

9. Може бути розгорнутий майже на будь-якому сервері;
10. Адаптований під велику кількість апаратних платформ і операційних систем.

Поєднавши в собі переваги інших мов програмування та обравши для себе єдине основне спрямування, PHP став універсальним, пристосованим до веб-програмування, зрозумілим та простим для вивчення засобом роботи в Інтернеті.

Як відомо, WordPress написаний і працює виключно на PHP [17, 18, 19, 20].

2.5.5 Система управління базою даних MySQL

Для реалізації простого інтернет магазину «StarBooks» в якості бази даних була обрана MySQL. Для адміністрування базою даних буде використаний веб-додаток з відкритим кодом, написаний на мові PHP, який називається PhpMyAdmin. Він дозволяє швидко і просто створювати, редагувати і видаляти не тільки базу даних, але і окремі її елементи, такі як: таблиці, атрибути таблиць і безпосередньо самі записи. Безперечно, це значно спрощує роботу з базою даних і скорочує час на її реалізацію [21].

MySQL – вільна реляційна система управління базами даних. Розробка і підтримка MySQL здійснює корпорація Oracle. Продукт поширюється як під GNU General Public License, так і під власною комерційною ліцензією. Крім цього, розробники створюють функціональність за замовленням ліцензійних користувачів. Саме завдяки такому замовленню майже в найраніших версіях з’явився механізм реплікації.

MySQL є рішенням для малих і середніх додатків. Входить до складу серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP, VertrigoServ. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць. Користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі і GPL ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць. З часом MySQL все розширювалася і зараз вона одна з найпоширеніших систем керування базами даних.

Вона використовується, в першу чергу, для створення динамічних веб-сторінок, оскільки має впевнену підтримку з боку різноманітних мов програмування [22].

Переваги та недоліки MySQL.

MySQL являється компактним, багатопоточним сервером баз даних та характеризується великою швидкістю, стійкістю і простотою використання.

Ця система вважається гарним рішенням для малих і середніх додатків. Вихідний код сервера компілюється на безлічі платформ. Найбільш повно можливості сервера виявляються в UNIX-системах, де є підтримка багатопоточності, що підвищує продуктивність системи в цілому. Для некомерційного використання MySQL є безкоштовним.

Можливості сервера MySQL:

1. Простота у встановленні та використанні;
2. Підтримується необмежена кількість користувачів, що одночасно працюють із БД;
3. Кількість рядків у таблицях може досягати 50 млн;
4. Висока швидкість виконання команд;
5. Наявність простої і ефективної системи безпеки

Недоліками сервера MySQL залишається не реалізована підтримка транзакцій, замість якої пропонується використовувати LOCK/UNLOCK TABLE, а також відсутність підтримки для зовнішніх (foreign) ключів, тригерів,

збережених процедур та представлень (VIEW). Та для розробки малих та середніх інформаційних систем, призначених для використання в межах робочих груп ці недоліки є фактично невідчутними.

Оскільки в якості CMS була обрана WordPress, то вибір СУБД бази даних очевидний – це MySQL. Як було сказано раніше, WordPress працює на PHP, тому потрібно розглянути взаємодію MySQL та PHP [23, 24].

Реалізація взаємодії MySQL та PHP.

Проекти на основі безкоштовного ПЗ, які вимагають повнофункціональної системи керування базами даних часто використовують MySQL. До таких проектів належать, наприклад, WordPress, phpBB, Drupal та інше програмне забезпечення, побудоване на стеку продуктів LAMP (Linux, Apache, MySQL, PHP/Perl/Python).

Популярний візуальний інтерфейс PhpMyAdmin, написаний на PHP, дозволяє працювати з MySQL в графічному режимі. Цей інтерфейс дозволяє значно спростити роботу з базами даних в MySQL.

У текстовому режимі робота з базою даних виглядає просто як введення команд у командний рядок, а результати вибірок повертаються у вигляді своєрідних таблиць, поля в яких налазять один на одного, якщо дані не вміщаються на екрані.

PhpMyAdmin дозволяє використовувати всі переваги роботи в браузері, включаючи прокрутку зображення, у випадку, коли йому для відображення бракує розмірів екрану. Більшість базових SQL-функцій роботи з даними в PhpMyAdmin зведені до інтуїтивно зрозумілих, завдяки інтерфейсу, дій, що нагадують перехід за посиланнями в мережі Інтернет.

Проте однозначною перевагою роботи в текстовому режимі залишається необмеженість функцій реалізованим інтерфейсом.

В дистрибутив самої мови PHP входить розширення, що містить вбудовані функції для роботи з базою даних MySQL. При роботі з веб інтерфейсом для додавання інформації до бази даних куристувч повинен ввести ці дані в HTML-форму та відправити їх на сервер, а все інше виконується засобами скрипта. Для

перегляду вмісту таблиць достатньо перейти по посиланню і зайти на потрібну сторінку.

Процес одержання доступу до СУБД MySQL та роботи з нею можна розбити на такі чотири послідовних кроки:

1. З’єднання з БД:
   1.1 mysql_connect() – з’єднання з MySQL;
   1.2 mysql_select_db() – вибір БД.
2. Доступ до БД і отримання необхідних даних:
   2.1 mysql_query() – виконати SQL-запит;
   2.2 mysql_fetch_array(), mysql_fetch_assoc() – витягнути наступний запис.
3. Звільнення пам’яті:
   3.1 mysql_free_result() – звільнення пам’яті, відведеної для зберігання результату запиту.
4. Закриття з’єднання:
   4.1 mysql_close() – закриття з’єднання; виконується автоматично після завершення виконання скрипта.

Дуже часто помилки в скриптах із доступом до БД виникають у зв’язку з некоректно сформованим SQL-виразом. Тому першим кроком у налагодженні PHP-програми, що виконує запити до MySQL повинна бути перевірка правильності SQL-інструкцій. Для цього слід вивести SQL-запит на сторінку, і скопіювавши його, виконати безпосередньо в phpMyAdmin [25].

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ ДОДАТКУ

3.1 Етапи розробки

Процес розробки веб-сайту можна розділити на наступні етапи:

1. Визначити цілі створення веб-ресурсу – на цьому етапі необхідно визначити, для чого потрібен сайт, тобто які завдання він повинен вирішувати: надати загальне уявлення про функції сайту.
2. Створення технічного завдання для розробки веб-сайту – необхідно описати, які функції повинен виконувати сайт, побажання по дизайну сайту, які технології будуть використовуватися при його створенні.
3. Створення макета для майбутнього дизайну – на цьому етапі необхідно зробити прототип майбутнього сайту, показати, яка в нього буде структура, як будуть розташовані ті чи інші блоки. На основі цього прототипу буде зроблений майбутній дизайн сайту за допомогою таких засобів, як Adobe Photoshop або Adobe Illustrator.
4. Верстання макету – за допомогою гіпертекстової розмітки (HTML) та каскадних таблиць стилів (CSS), а також мови сценаріїв (JavaScript) проводиться верстка макету [26, 27].
5. «Натягування» макету на обрану CMS – з урахуванням специфіки обраної CMS, проводиться натягування розробленого макету під неї.
6. Програмування сайту – за допомогою PHP відбувається створення всіх сторінок сайту, визначається порядок роботи меню, розставляються гіперпосилання, створюється динаміка на сайті, програмуються основні складові потрібного функціоналу.
7. Наповнення сайту інформацією – додається контент сайту, тобто якась текстова та мультимедійна інформація.
8. Розташування сайту в мережі Інтернет – передбачається, що розробка сайту проводиться на локальному хості (Denwer, OpenServer), і тільки

після повної розробки сайт потрібно завантажити на хостинг в Інтернеті.

9. Тестування сайту – цей етап можна здійснити як до, так і після розміщення сайту за його «рідною» адресою. На цьому етапі виявляються всі помилки і недоліки в програмуванні і написанні текстів.

3.2 Принципи розробки дизайну сайта

На сьогоднішній час веб-дизайн базується на принципах акцентування, контраст, балансування, вирівнювання, повторення і зручність сприйняття.

Акцентування – це підкреслення особливої важливості або значимості якого-небудь елементу. Щоб дотримати в процесі створення сайту принцип акцентування, необхідно проаналізувати вміст сайту і з'ясувати, яка ієрархія елементів має місце в його вмісті, тобто треба уявити собі перелік елементів, які потрібні на веб-сторінці.

Контрастування – це візуальна диференціація двох і більше елементів. Елементи з високим ступенем контрастування виглядають чітко і відокремлено, в той час, як елементи з низьким ступенем контрастування здаються такими, що зливаються і мають тенденцію змішуватися в єдиній масі. Ця характеристика допомагає швидко привертати увагу до ключових елементів, наприклад до вмісту, елементів, що дозволяє утримувати певну дію, або до тексту, що дає зрозуміти, в чому полягає призначення сайту.

Принцип балансування крутиться навколо ідеї про те, як розподілені елементи в дизайні і як вони співвідносяться із загальним розподілом візуальної навантаження в рамках веб-сторінки.

Вирівнювання – це впорядкування елементів таким чином, щоб вони максимально близько підступали до природних ліній або кордонів, які вони утворюють. Тобто, це означає, що краще впорядковувати елементи по сітці.

Повторення передбачає неодноразове використання в дизайні одного і того ж елемента різними способами. Дизайн, який містить повторення, стає уніфікованим. Повторення може виражатися в безлічі форм, включаючи використання того ж кольору, обрисів, лінії, шрифти, зображення і загальний підхід до стилізації.

Зручність сприйняття – свого роду стежка, якою користувач дотримується при візуальному ознайомленні з елементами дизайну. Воно головним чином залежить від того, на чому дизайнер зробив акцент і в якій мірі він забезпечив контрастування елементів між собою. Фактично зручність сприйняття цілком залежатиме від того, як дизайнер скомпонував всі елементи сторінки.

При дотриманні всіх цих правил, можна зробити простий, зручний і привабливий дизайн для сайту.

3.3 Структура сайту

Структура сайту – це організація зв'язків між сторінками, розділами і різними елементами сайту.

Навігація забезпечує доступ до неї, тобто різні меню, перехресні посилання і карта сайту.

На етапі планування сайту і написання технічного завдання необхідно ретельно її продумати. Зручна і зрозуміла структура сайту допомагає користувачеві легко знаходити потрібну йому інформацію.

На сайтах із дуже великим обсягом інформації є сторінки третього рівня, а якщо необхідно – то й четвертого, п’ятого тощо. Загалом виділяють три типи структур веб-сайтів – лінійну, деревоподібну та довільну. Подорожуючи сайтом із лінійною структурою, з головної сторінки ви перейдете на другу сторінку, з неї – на третю тощо. На сайті з деревоподібною структурою з головної сторінки можна потрапити на одну з сторінок другого рівня, звідти – на одну з сторінок третього рівня тощо. Сайт із довільною структурою видається зовсім

неорганізованим, але саме в цьому й полягає принцип його створення. Подорожуючи таким сайтом, можна переходити з однієї його сторінки на інші в різні способи, і ваш шлях назад не обов'язково має бути таким самим.

Вибір структури визначається особливостями завдань, що розв'язуються за допомогою веб-сайту (рис 3.1).

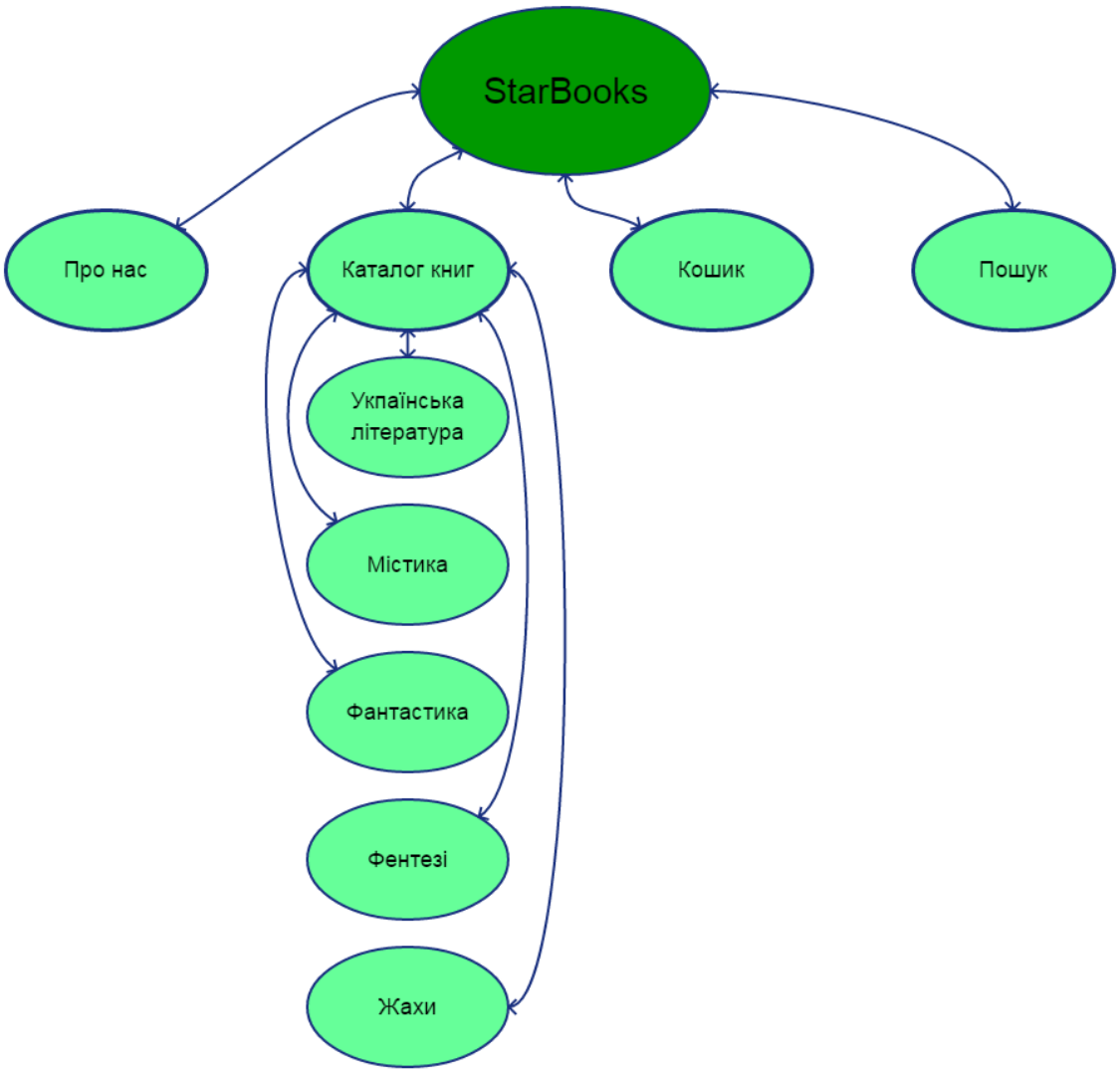

Рисунок 3.1 – Структура сайту «StarBooks»

Більшість веб-ресурсів побудовані за блоковим принципом. Все просто: нагорі шапка, зліва – меню, праворуч – блок реклами, посередині – інформація, а внизу (в підвалі сайту) написано – «Копірайт – всі права захищені».

Структура сайту може бути ускладнена – додано додаткове меню під шапкою і продубльовано над копірайтом. Також навігаційні посилання розміщують і справа замість реклами – це кому як подобається і залежить від призначення даного ресурсу. Така структура хороша для інформаційного порталу, оскільки залишено багато місця під посилання на різні розділи і лише під шапкою і над «підвалом» сайту відбувається дубляж. Однак, в такій структурі важче шукати потрібну інформацію, тому останнім часом багато сайтобудівники використовують техніку цільової сторінки, де інформація легко і швидко сприймається користувачами.

Landing Page (цільова сторінка) – це веб-сторінка, основне завдання якої є збір контактних даних цільової аудиторії. Використовується для посилення ефективності сприйняття інформації, збільшення аудиторії. Цільова сторінка зазвичай містить інформацію про послугу, на яку треба звернути увагу.

Існує чотири основних типи цільових сторінок:

1. Автономна цільова сторінка;
2. Мікросайту;
3. Головний сайт;
4. Сторінка сегментація.

Структура сайту «StarBooks» буде побудована по типу мікросайту.

Мікросайт – це окремий сайт з інформацією про рекламі послуги або товар, що складається з декількох сторінок. Зазвичай «мікро» містить мінімум текстової інформації, велика кількість привабливого товару з красивими зображеннями або послуги, відео-ролики. Подібний тип цільових сторінок часто використовується великими компаніями для реклами окремих проектів.

Landing Page має наступні переваги:

1. Зосередженість користувача;
2. Вищий коефіцієнт конверсії;
3. Стимуляція користувачів зробити конкретну дію;
4. Має більш привабливий вигляд, ніж звичайниі блокові сайти.

Результативність та ефективність сприйняття користувача інформації залежить від того, чи познайомитися користувач взагалі з приготовленими для нього пропозиціями. Тому однією з найважливіших особливостей цільової сторінки є її доступність для користувача. Враховуючи всі переваги Landing Page, головна сторінка, тобто точка входу на сайт, зроблена в даному стилі (рис 3.2).

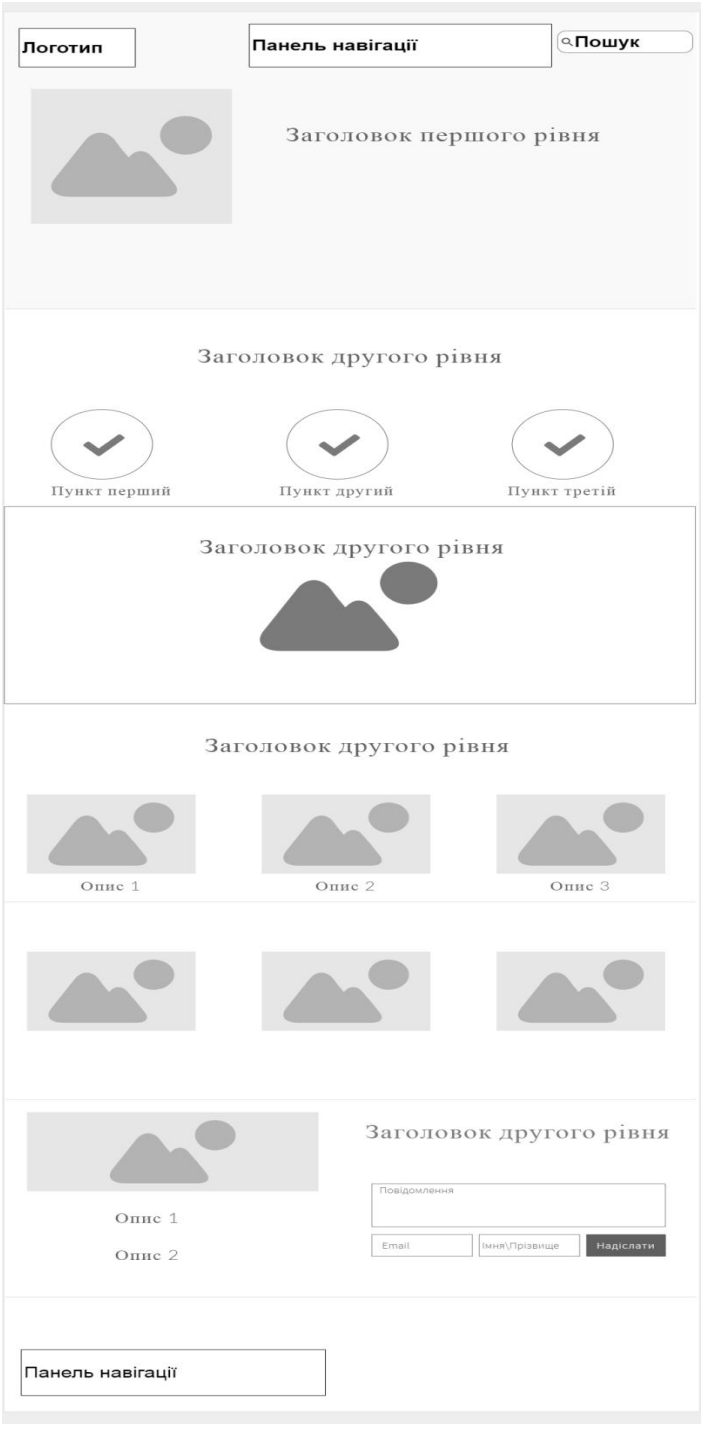

Рисунок 3.2 – Структура головної сторінки сайту «StarBooks»

3.4 Веб інтерфейс

Веб-інтерфейс – це сукупність засобів, за допомогою яких користувач взаємодіє з веб-сайтом або веб-застосунком через браузер. Веб-інтерфейси отримали широке поширення у зв'язку із зростанням популярності всесвітньої павутини і відповідно повсюдного розповсюдження веб браузерів.

Одним з основних вимог до веб-інтерфейсів є їхній однаковий зовнішній вигляд і однакова функціональність при роботі в різних браузерах.

Головна сторінка – це точка входу користувача на сайт, тому вона повинна бути максимум інформативна, містити навігацію, і зверстана таким чином, щоб користувачеві було легко сприймати інформацію[28, 29].

Для скорочення часу на створення головної сторінки сайту у вигляді Landing Page, була використана тема Hestia, яка останнім часом користується великою популярністю серед розробників на WordPress, через те, що включає в себе багато вбудованих модулів, використання яких допомагає скоротити час на рішення одноманітних завдань у вигляді розташування блоків в секціях.

В темі Hestia є безліч налаштувань за допомогою яких можна адаптивно розташувати секції на сторінці.

Тема Hestia використовує сітку Twitter Bootstrap, що робить створені секції адаптивними під будь-який дозвіл пристроїв, а це значить, що даний сайт буде правильно відображатися на різних пристроях.

Bootstrap – вільний набір інструментів для створення сайтів і веб додатків. Включає в себе HTML і CSS-шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації та інших компонентів веб-інтерфейсу, включаючи JavaScript-розширення.

Bootstrap використовує сучасні напрацювання в області CSS і HTML, тому необхідно бути уважним при підтримці старих браузерів.

До недоліків теми можна віднести бідну колірну гамму стандартного набору іконок. До переваг – хорошу реалізацію grid-сітки для масштабування веб-сторінки, створення адаптивного дизайну.

Основні переваги Bootstrap:

1. Сітка;
2. Адаптивність;
3. Форма і інші елементи;
4. Динаміка;
5. Повна інтеграція з препроцесором CSS, наприклад, LESS;
6. Універсальність.

Для реалізації сторінок каталогу книг був використаний пагін WooCommerce.

WooCommerce – це плагін для електронної комерції з відкритим кодом для WordPress. Він призначений для інтернет-магазинів малого та великого розміру за допомогою системи управління вмістом WordPress [30].

На сьогоднішній день, це найкраще рішення для тих, хто хоче створити інтернет-магазин на WordPress. Більше 30% інтернет-магазинів у всьому світі працюють на цьому плагіні.

До недоліків WooCommerce можна віднести вимогливість до ресурсів. Після встановлення плагіну на WordPress, навантаження, створювана на ваш сайт, серйозно зросте. При використанні слабкого хостингу або неякісної оптимізації сайту, це призведе до збільшення часу завантаження сторінок.

Основні переваги плагіну :

1. Безкоштовність;
2. Простота установки;
3. Зручний адміністраторський інтерфейс;
4. Докладна інформація про замовлення;
5. Вбудована зручна система аналітики.

Розглянемо головну сторінку сайту «StarBooks» на рис. 3.3.

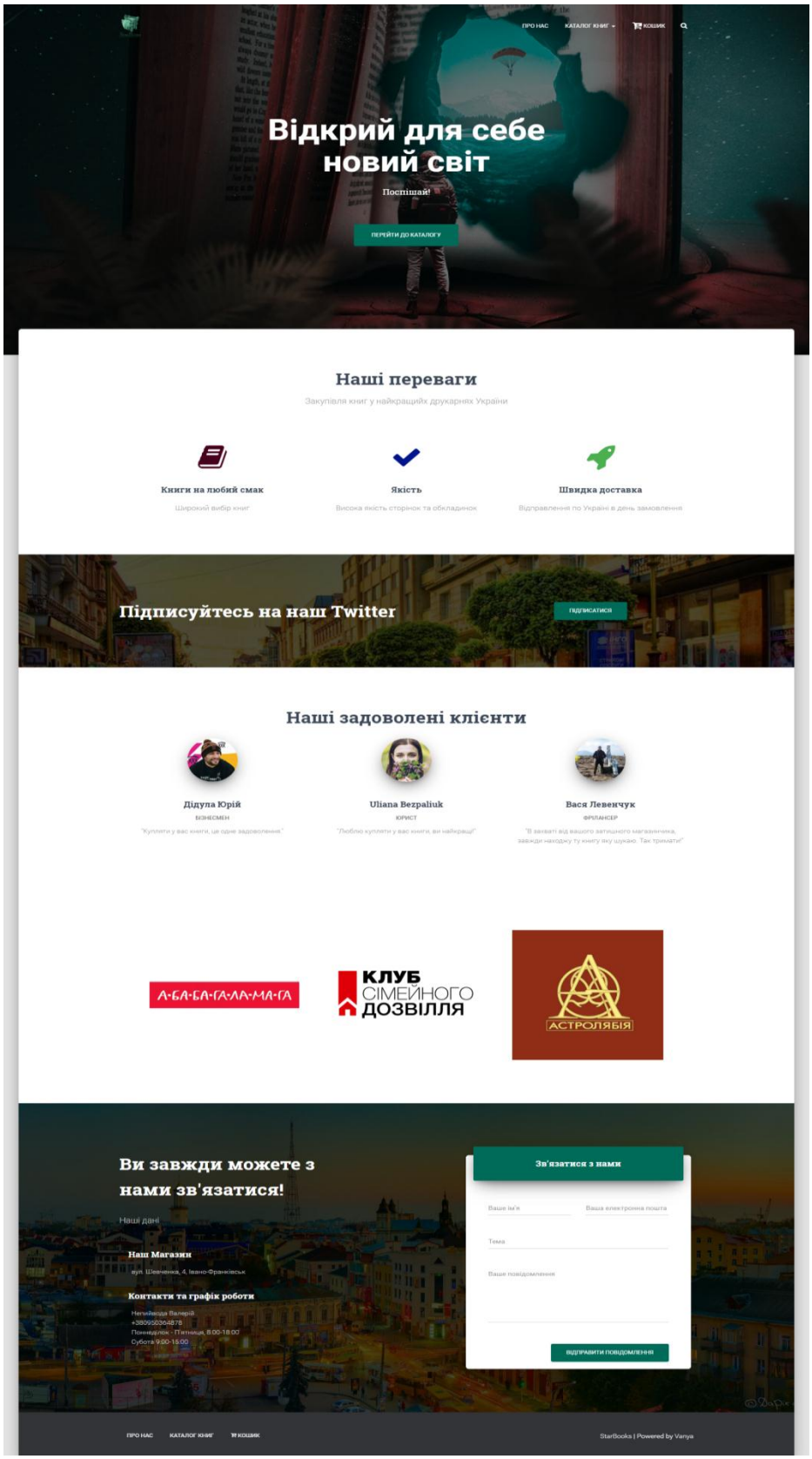

Рисунок 3.3 Головна сторінка сайту «StarBooks»

Розглянемо більш детально інші сторінки сайту.

На даній сторінці відображаються книги усіх жанрів які є на даному сайті (рис 3.4).

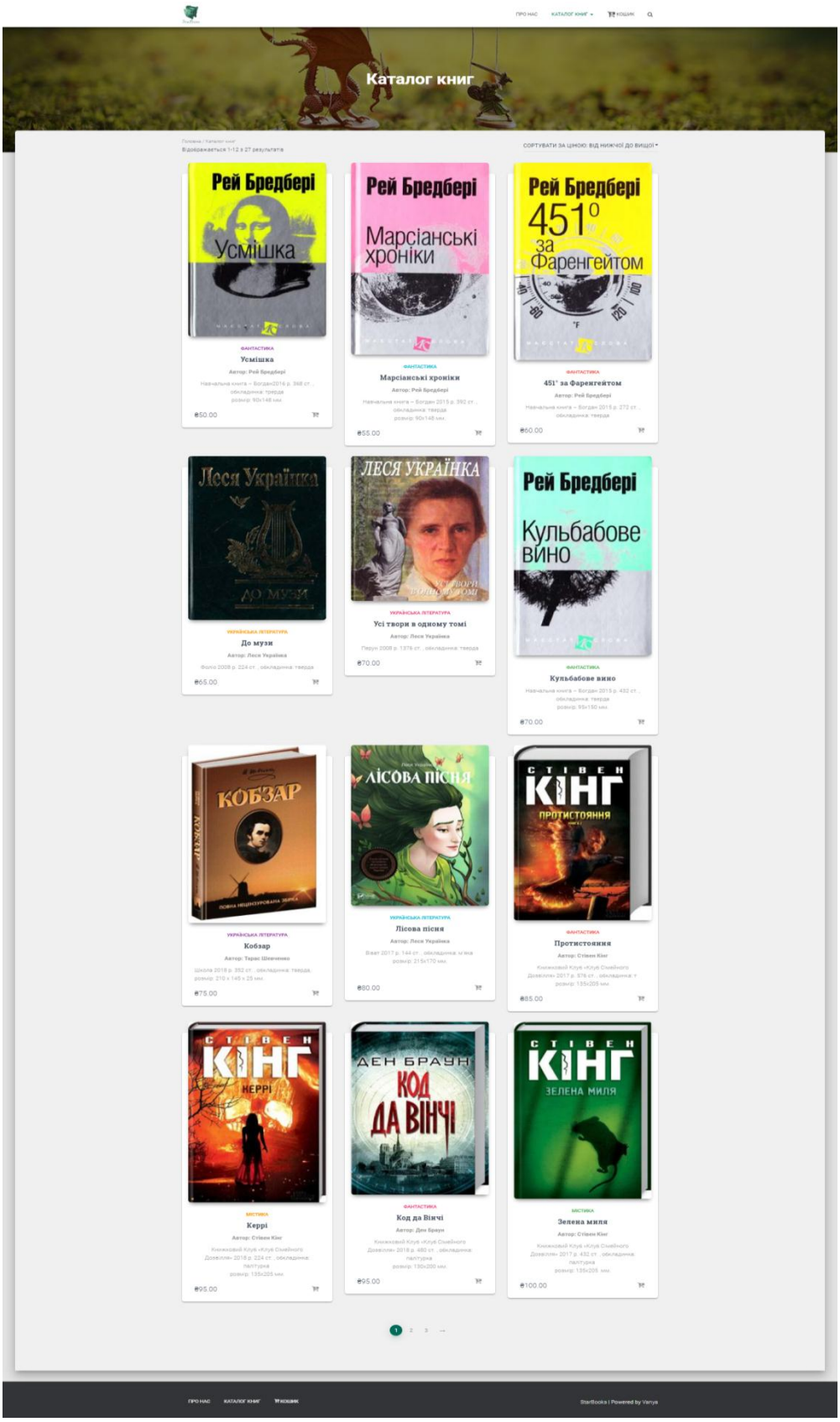

Рисунок. 3.4 Сторінка «Каталог книг»

На сторінці «Містика» відображаються книги даного жанру (рис 3.5).

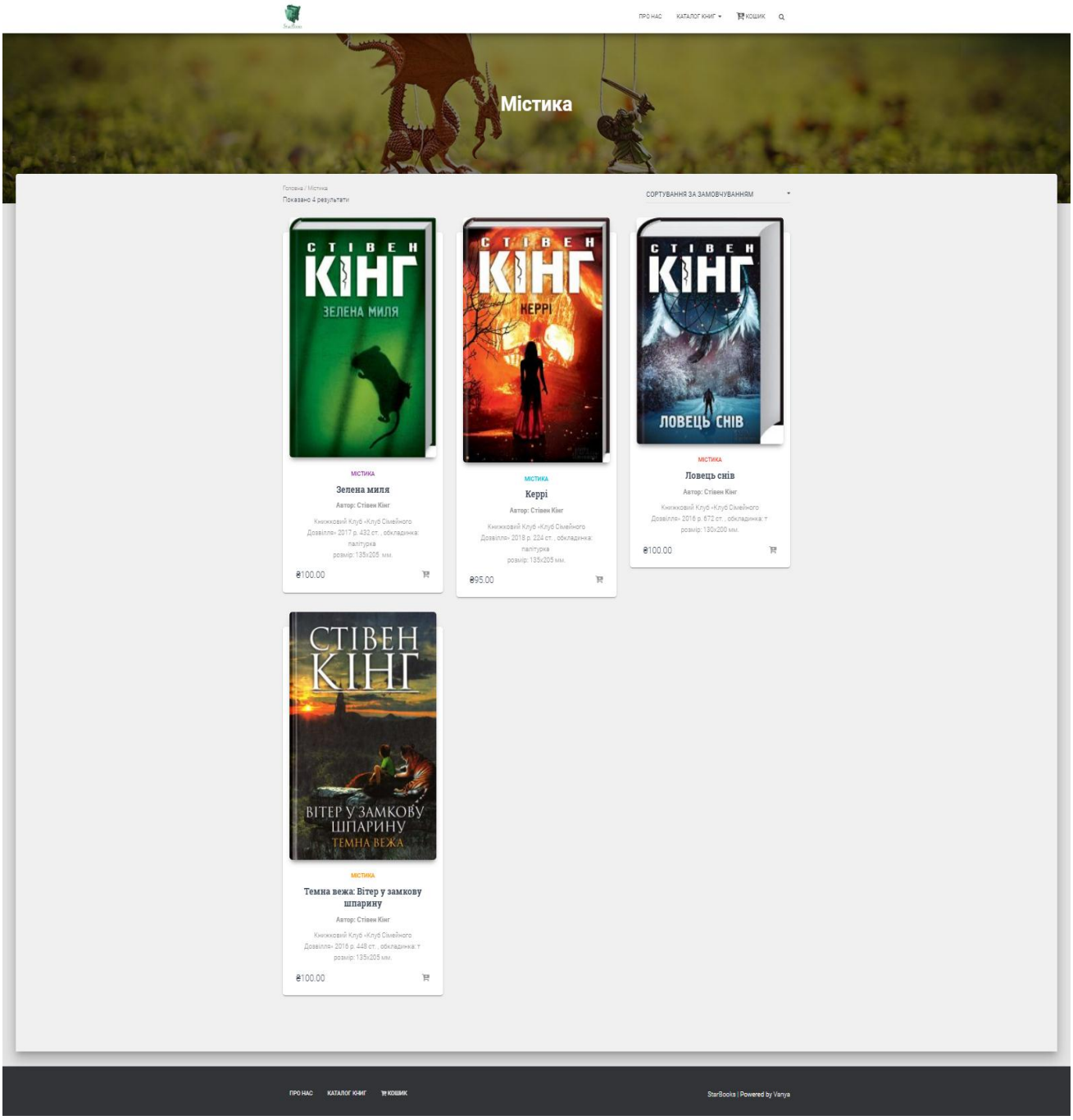

Рисунок. 3.5 Сторінка «Містика»

Тут можна вибрати любу книгу вітчизняних авторів (рис. 3.6).

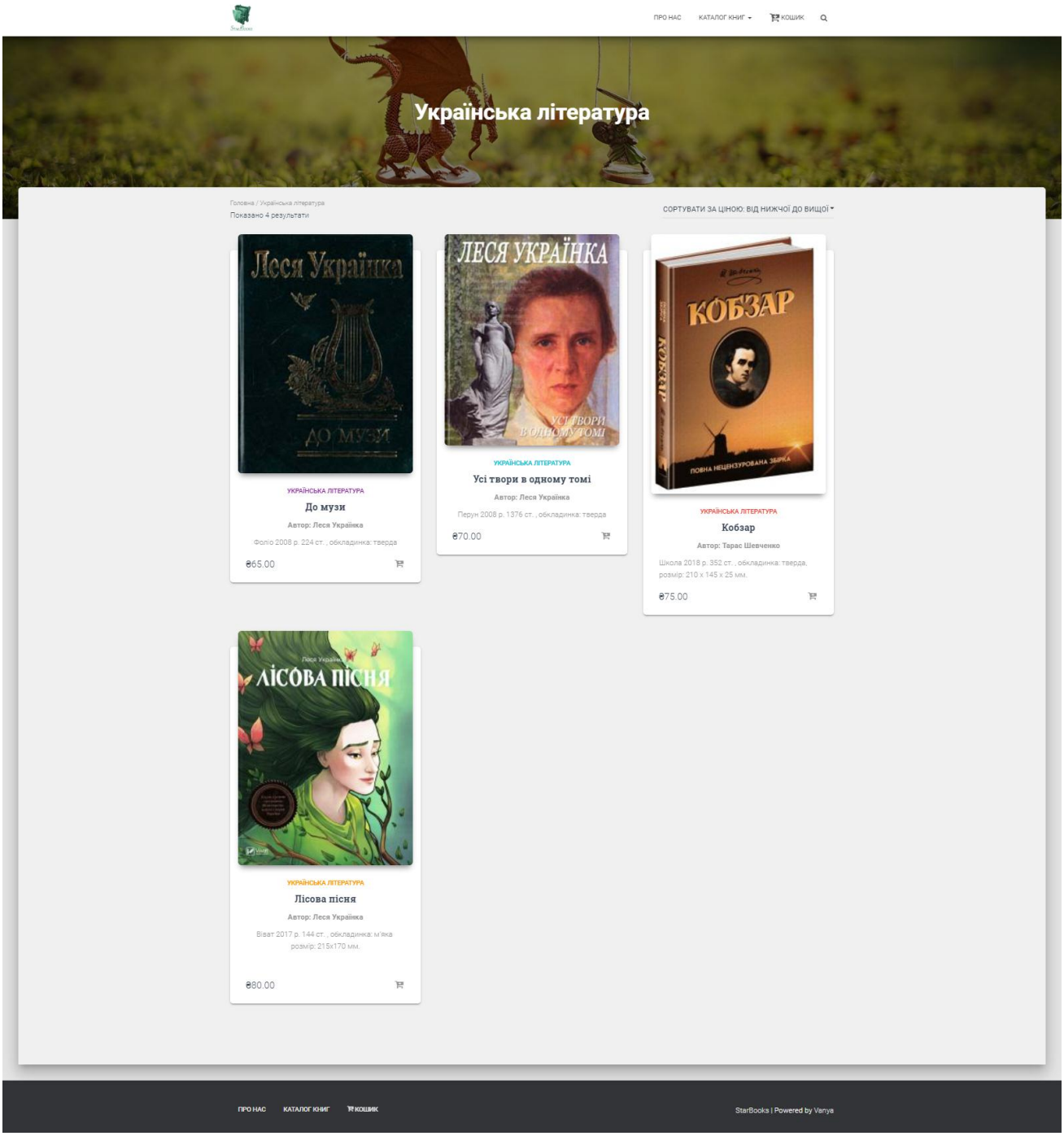

Рисунок 3.6 Сторінка «Українська література»

Перейшовши на дану сторінку користувач може переглянути книги жанру «Фантастика» (рис. 3.7).

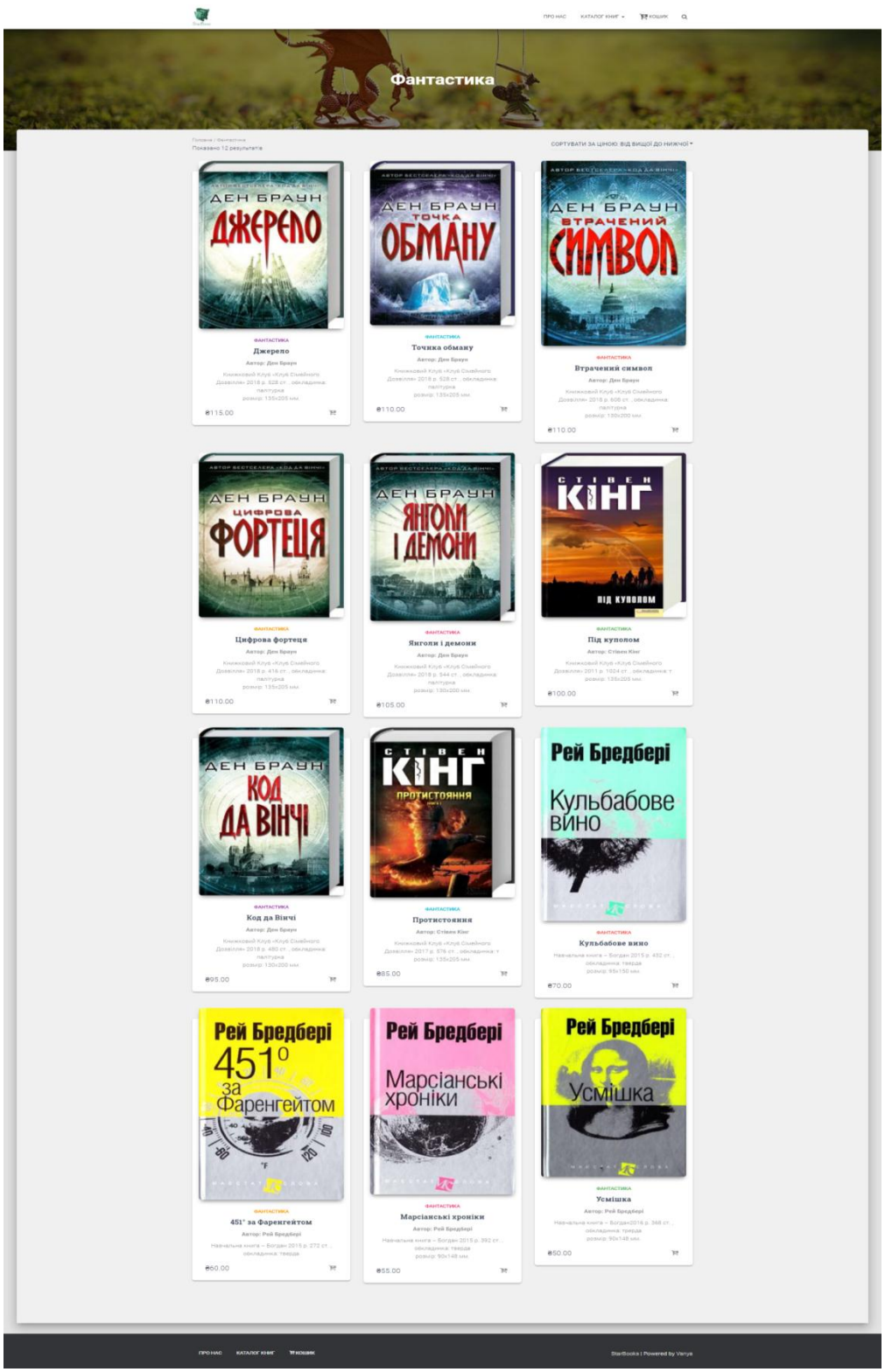

Рисунок 3.7 Сторінка «Фантастика»

Перейшовши на цю сторінку, ми можемо побачити та обрати для себе книги жанру «Жахи» . (рис. 3.8).

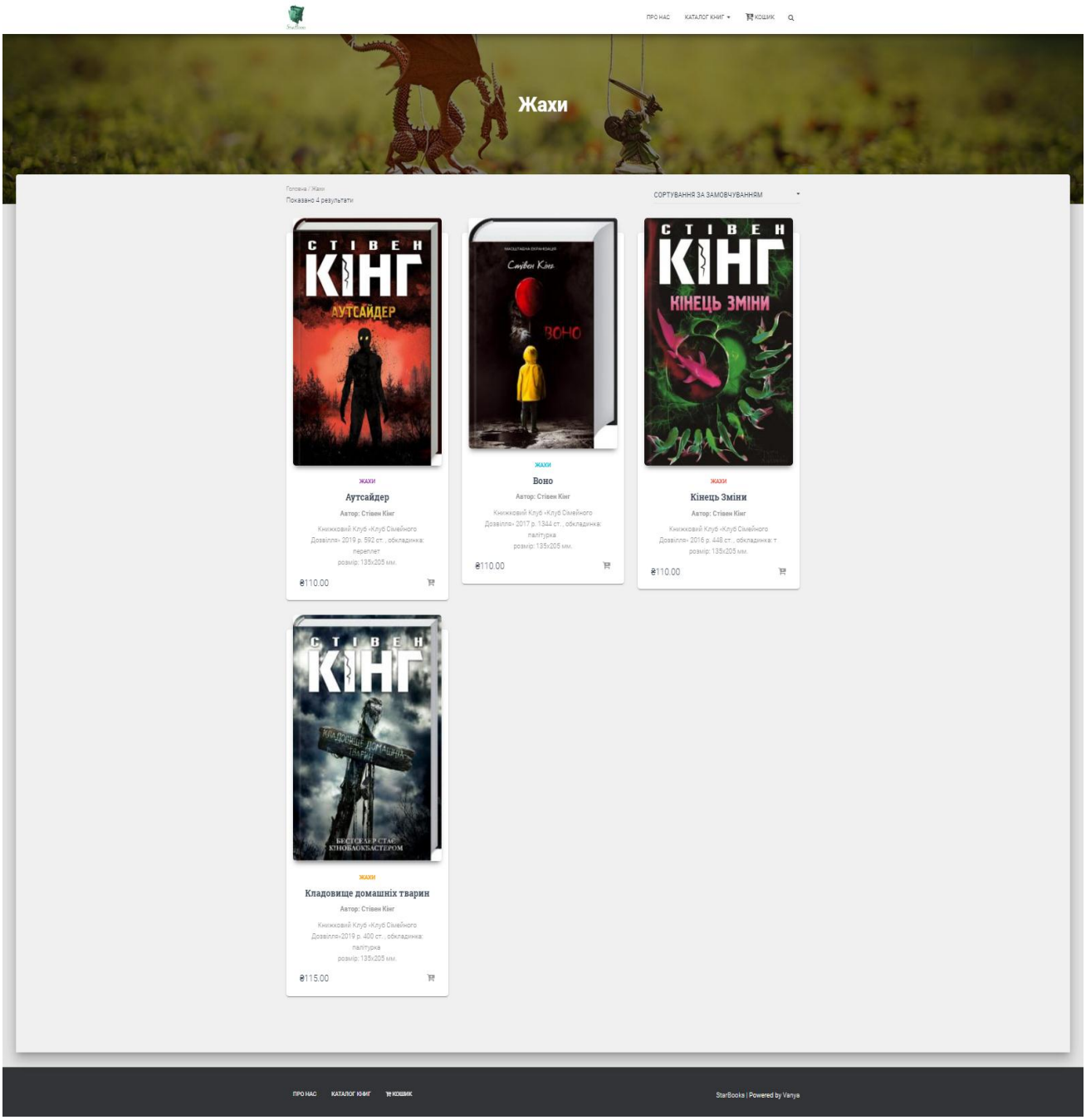

Рисунок. 3.8 Сторінка «Жахи»

Всі книги жанру «Фентезі» відображаються на цій сторінці (рис. 3.9).

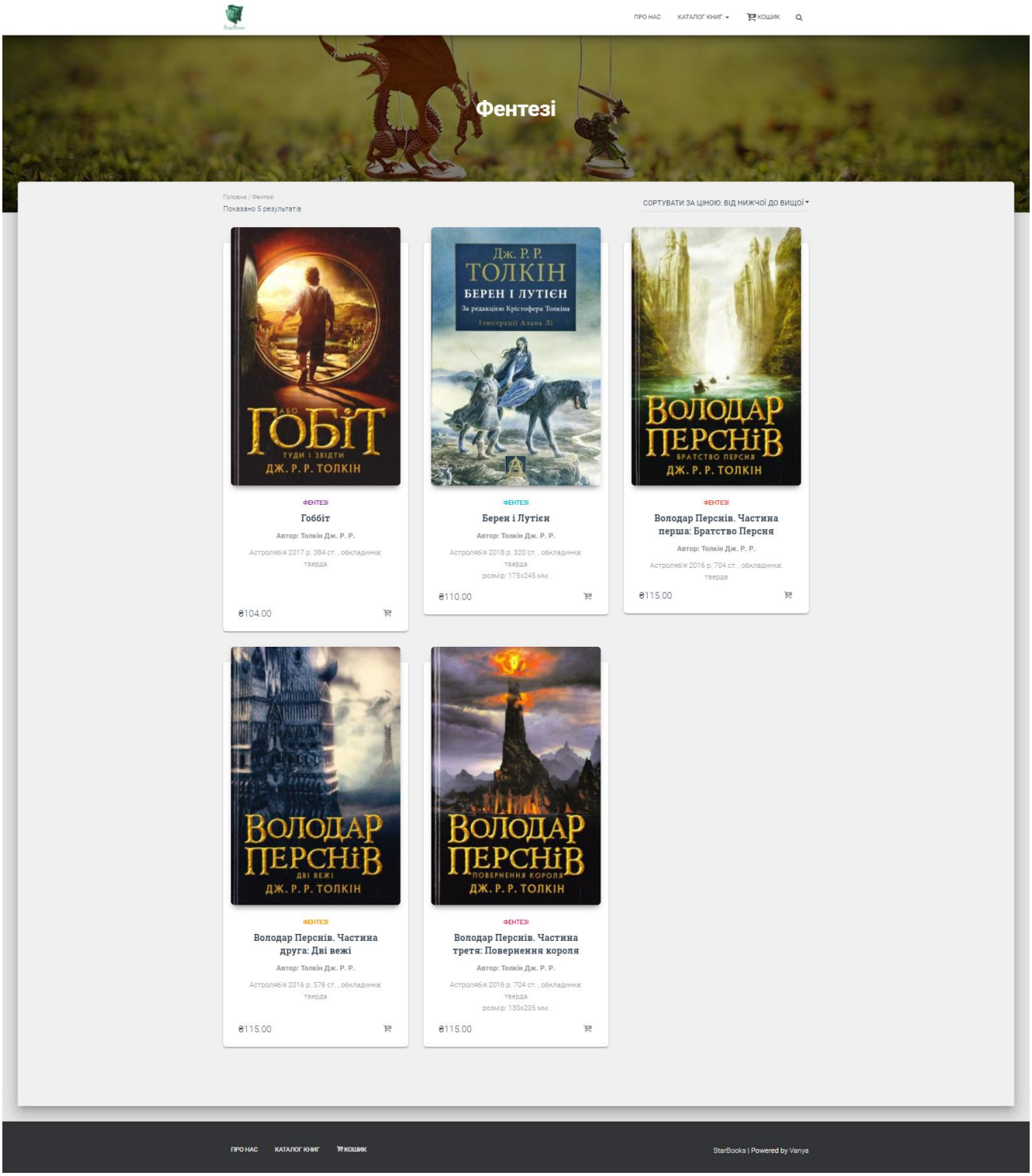

Рисунок 3.9 Сторінка «Фентезі»

Перейшовши по даній сторінці ми можемо дізнатися автора книги, короткий опис, видавництво, розмір, кількість сторінок, інформацію про палітурку книги, ціну, залишити відгук та додати дану книгу в кошик. Також при натисканні значка лупи, користувач може ознайомитися з уривком книги яку він обрав, та вирішити для себе сподобалась вона йому чи ні (рис. 3.10).

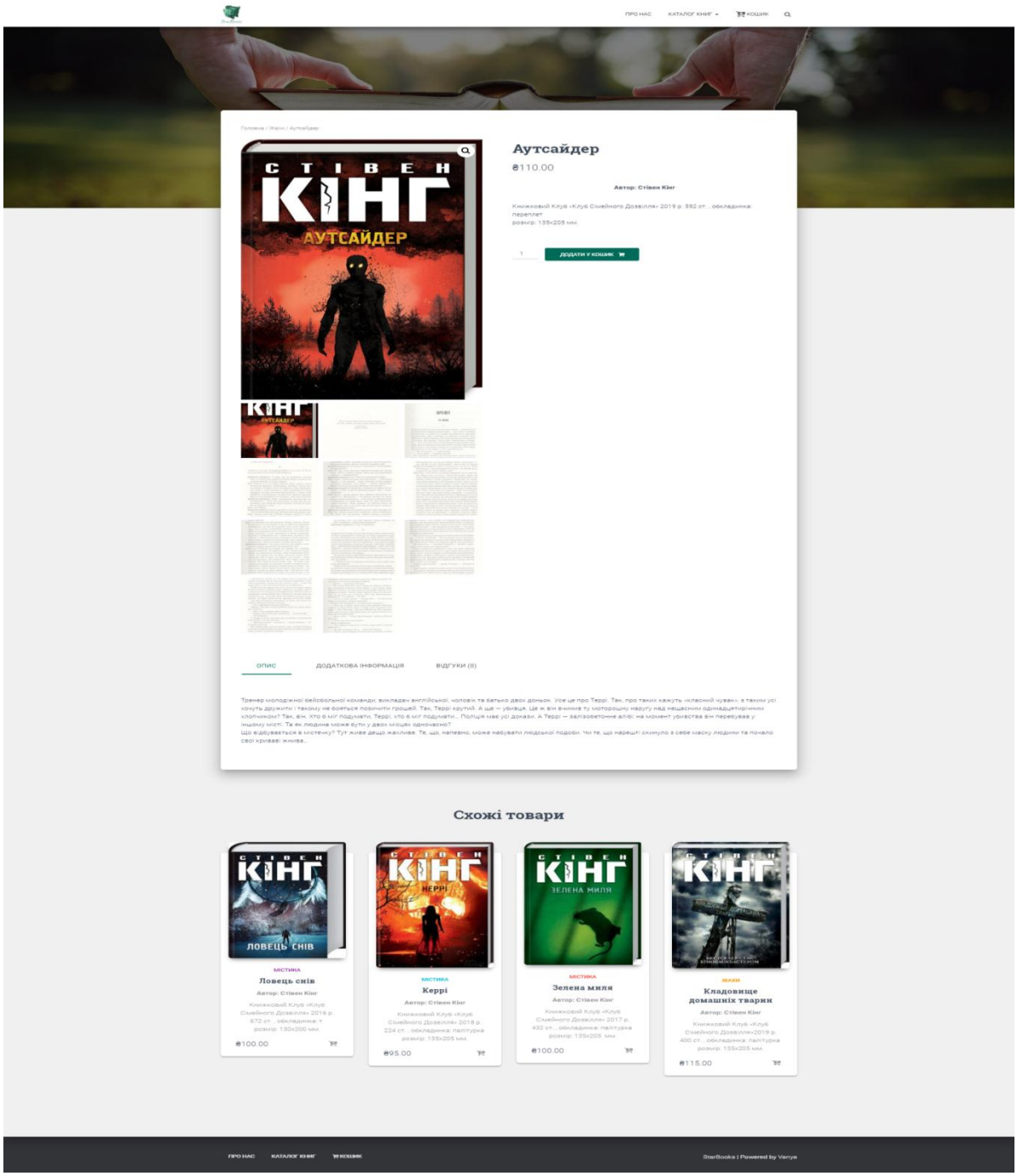

Рисунок 3.10 Сторінка книги «Аутсайдер»

В кошику можна переглянути кількість книг, ціну доданих книг, та ціну разом з доставкою. Після цього можна перейти до оформлення замовлення (рис. 3.11).

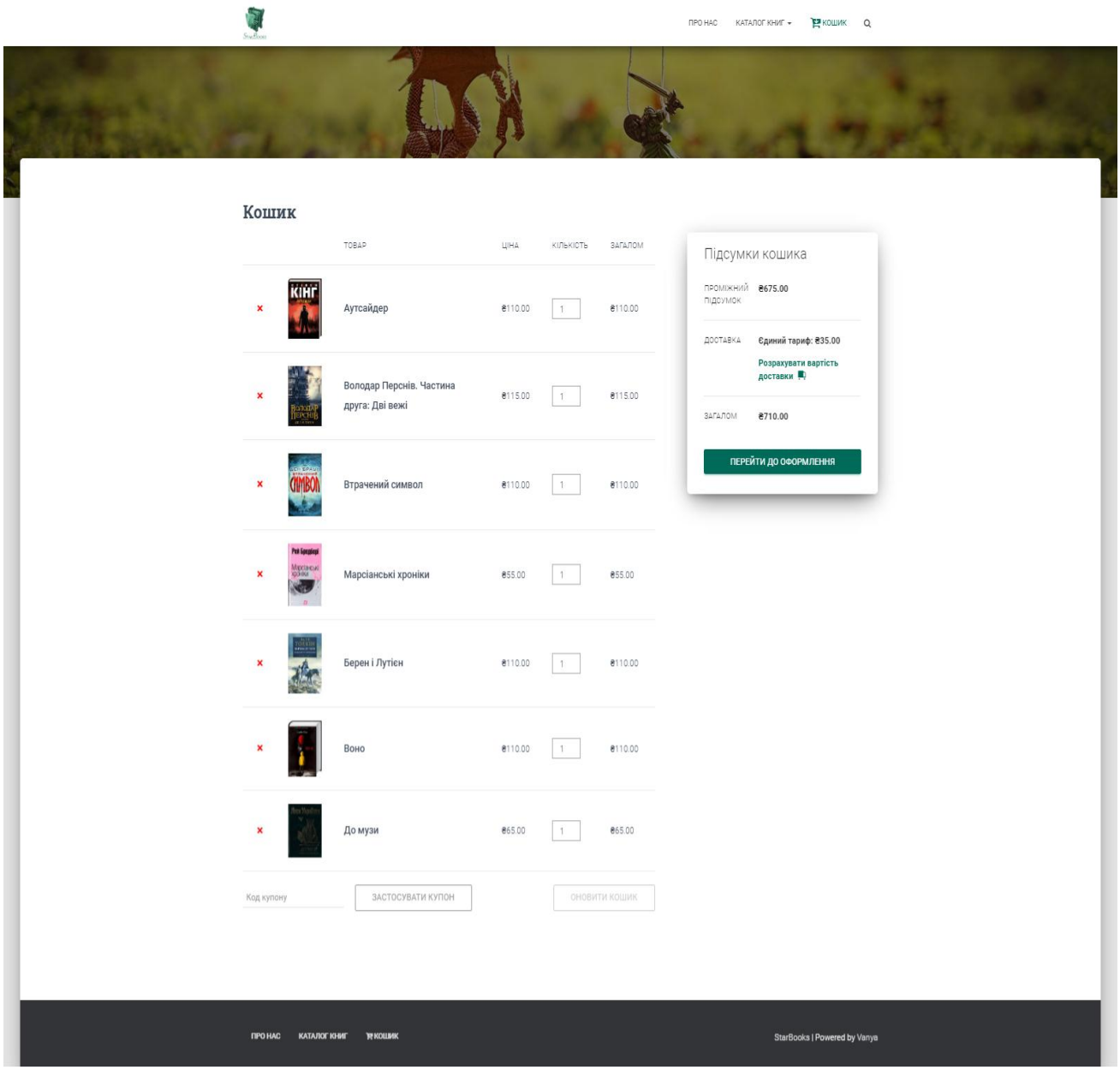

Рисунок 3.11 Сторінка «Кошик»

На даній сторінці розташовується форма, в якій користувач вводить персональні дані та дані про доставку товару. Також тут можна обрати спосіб оплати книг, користувач може заплатити при отриманні книги, або банківським переказом (рис 3.12).

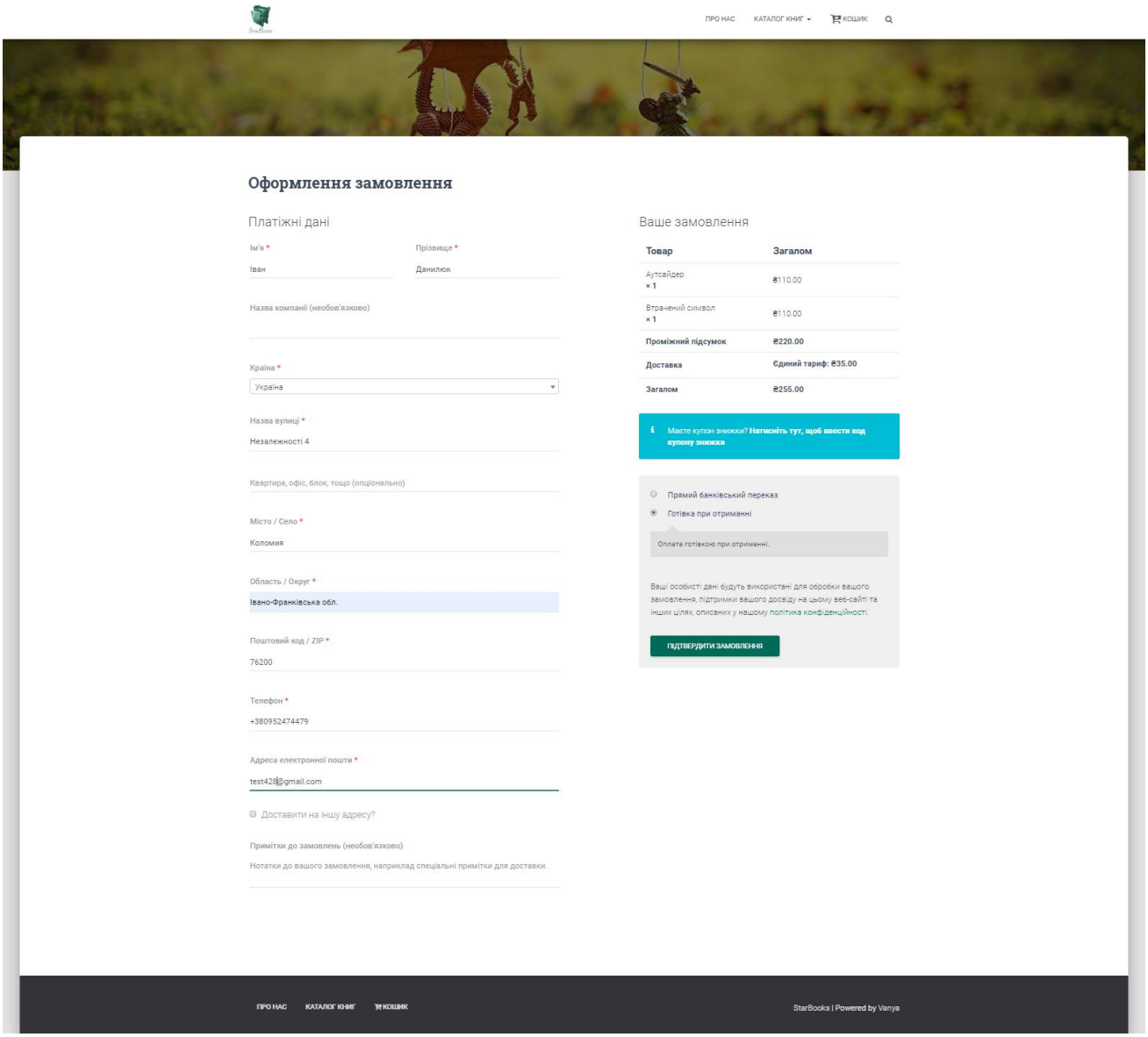

Рисунок 3.12 Сторінка «Оформлення замовлення»

Після натискання кнопки «Оформити замовлення» користувач буде перенаправлений на сторінку де зможе переглянути всі вибрані книги, кінцеву ціну та вибраний спосіб оплати (рис. 3.13).

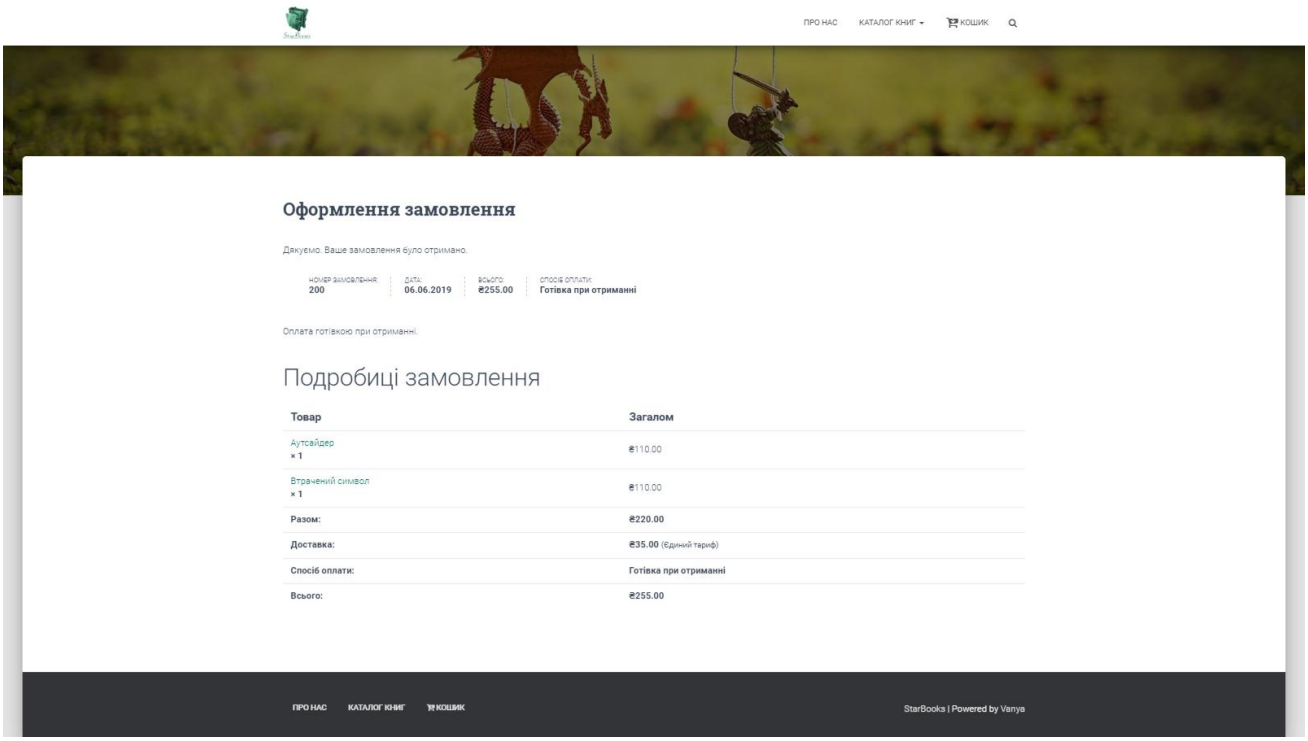

Рисунок 3.13 Сторінка «Подробиці замовлення»

В адмінпанелі при натисканні на вкладку «Товари» можна додати, редагувати або видалити книгу (рис 3.14).

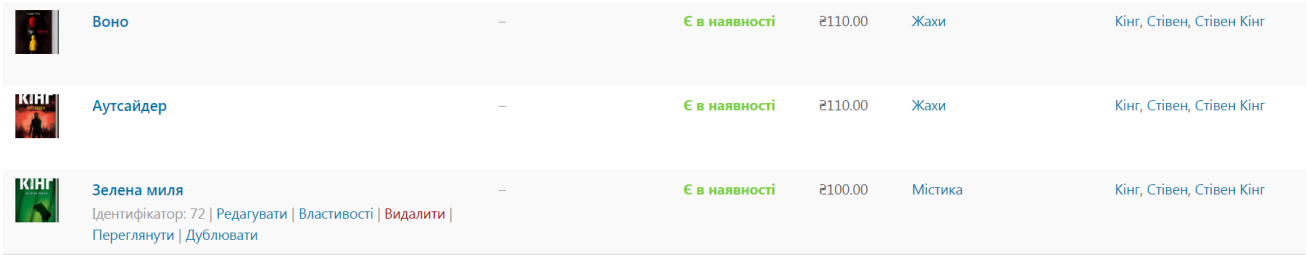

Рисунок 3.14 Вкладка «Товари» в адмінпанелі

ВИСНОВКИ

У дипломній роботі за результатами виконаних теоретичних та практичних досліджень розроблено програмне забезпечення, з допомогою якого можна придбати книги за різними категоріями. Реалізація веб-додатку здійснювалася за допомогою системи управління вмістом WordPress, теми Hestia та плагіну WooCommerce.

Завдяки використанню різних серверних і призначених для користувача елементів управління, застосування каскадних таблиць стилів і тем оформлення, вдалося створити привабливий, функціональний, зручний і інтуїтивно-зрозумілий інтерфейс, з можливістю швидкої модернізації в разі потреби.

У першому розділі дипломної роботи проведене дослідження, яке показало що є потреба у розробці програмного забезпечення, де кожна людина зможе здійснити замовлення книг з будь-якої точки України. Також, проведено аналіз вітчизняних аналогів програмного забезпечення та розглянуті їхні переваги та недоліки.

У другому розділі проведене моделювання веб-ресурсу з продажу книг та обране програмне забезпечення для реалізації продукту. Під час моделювання, розроблено діаграму прицедентів, суть даної діаграми полягає в тому що, проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Розроблено діаграму класів що відображає лише імена класів або імена та відповідні атрибути класів, або імена, атрибути та операції класів, класи можуть перебувати у певних відношеннях або зв’язках. Також, розроблена контекстна діаграма та діаграма декомпозиції. Побудова контекстної діаграми починається з отримання всієї системи у вигляді одного блоку і граничних стрілок, що зображують зв’язок з функціями поза системою, а для головного процесу присутнього на контекстній діаграмі, проведена декомпозиція, тобто поділ складного об'єкта, системи, завдання на складові частини.

Проведена порівняльна оцінка функціональних засобів для реалізації веб середовища показала, що серед систем управління вмістом найефективнішим буде CMS WordPress. В даній системі управління вмістом є ряд переваг, таких як велика кількість шаблонів, плагінів та відкритий вихідний код.

У третьому розділі проведені покрокові етапи розробки веб-додатку, принципи розробки дизайну сайту при дотриманні яких, можна зробити простий, зручний і привабливий дизайн для сайту. Проведена побудова структури веб додатку. В результаті обрано довільну структуру, так як подорожуючи таким сайтом, можна переходити з однієї його сторінки на інші в різні способи, і шлях назад не обов'язково має бути таким самим. При розробці веб інтерфейсу велику увагу надано структурі головної сторінки інтернет магазину. Головна сторінка – це точка входу користувача на сайт, вона максимум інформативна, містить навігацію, і зверстана таким чином, щоб користувачеві було легко сприймати інформацію. Для реалізації сторінок каталогу книг був використаний пагін WooCommerce. Основні переваги даного плагіну полягають в зручному адміністраторському інтерфейсі, вбудованій системі аналітики та докладній інформації про замовлення. На сьогоднішній день, це найкраще рішення для створення інтернет магазину на CMS WordPress.

Запропонований підхід розробки веб-ресурсу може бути рекомендований для широкого використання, оскільки реалізація продукту потребує мінімальних затрат, а інтернет-магазин має зручну та просту для розуміння адмінпанель та інтуїтивно зрозумілий інтерфейс.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Чому треба читати URL: https://mizky.com/article/150/benefits-reading (дата звернення: 27.01.2019).
2. Що читають українці URL: https://osvitoria.media/experience/shho-chytayut-ukrayintsi-prygolomshlyvi-tsyfry-ta-fakty/ (дата звернення: 29.01.2019).
3. Yakaboo URL: https://www.yakaboo.ua/ua/ (дата звернення: 29.01.2019).
4. Книгаренька. Маміно URL: https://www.knygarenka.com/ (дата звернення: 31.01.2019).
5. Книгарня Нора – Друк URL: https://books.nora-druk.com/ (дата звернення: 31.03.2019).
6. Дужий В. И. Волковая А.В. Программное обеспечение для компьютерных систем и сетей. Разработка технического. Харьков: Нац. аэрокосм. ун-т «Харьк. авиац. ин-т», 2007. 101 с.
7. И. К. Фролов, В. А. Перелыгин, Е. Э. Самойлов. Разработка, дизайн, программирование и раскрутка web-сайта. Москва: ТРИУМФ, 2009. 541 с.
8. Моделювання веб сайту. Матеріал з habr. URL: https://habr.com/ru/post/273795/ (дата звернення: 13.02.2019).
9. Основи декомпозиції. URL: https://studopedia.com.ua/1_162873_diagrami-dekompozitsii.html
10. Joomla! Матеріал з Вікіпедії – вільної енциклопедії: https://uk.wikipedia.org/wiki/Joomla (дата звернення: 13.02.2019).
11. Колисниченко, Д.Н. Выбираем лучший бесплатный движок для вашего сайта. CMS, Joomla! и Drupal. Санкт-Петербург.: БЧВ-Петербург, 2010. 288 с.
12. Ромашов В. CMS Drupal Система управления содержимым сайта. Санкт-Петербург: Питер, 2010. 254 с.

13. CMS DataLife Engine. Матеріал з Вікіпедії – вільної енциклопедії: https://dle-news.ru/release/1767-datalife-engine-v132-final-release.html (дата звернення: 09.03.2019).
14. Горнаков, С.Г. Осваиваем популярные системы управления сайтом. Москва: ДМК Пресс, 2009. 336 с.
15. Ульямс Б. WordPress для профессионалов. Разработка и дизайн сайтов – Санкт-Петербург: Питер, 2014. 464 с.
16. Грачев. А. Создаем свой сайт на WorldPress: работа с CMS WorldPress 3. Санкт-Петербург: Питер, 2011. 282 с.
17. Сергеев А. Н. Создание сайтов на основе WordPress. Санкт-Петербург: Питер, 2015. 275 с.
18. Tadlock Justin. Professional WordPress Plugin. Washington: John Wiley & Sons Limited, 2018. 536p.
19. PHP – скрипкова мова програмування. Матеріал з habr. URL: https://habr.com/ru/hub/php/ (дата звернення: 10.05.2019).
20. PHP. Матеріал з Вікіпедії – вільної енциклопедії: https://uk.wikipedia.org/wiki/PHP (дата звернення: 13.05.2019).
21. Диго С.М. Проектирование и эксплуатация баз даннях. Москва: Финансы и статистика, 1995. 280 с.
22. Воройский Ф. С. Основные технологические принципы разработки электронных библиотек гуманитарных вузов России. Москва: ТНБ, 2008. 324 с.
23. Спикльмайр С. Разработка Web-приложений и управление контентом для программистов . Москва: ДМК, 2003. 464 с.
24. Томсон Л. Разработка Web-приложений на PHP и MySQL. Санкт-Петербург: ООО «ДиаСофт», 2003. 672 с.
25. MySQL. Матеріал з Вікіпедії – вільної енциклопедії: https://uk.wikipedia.org/wiki/MySQL (дата звернення: 16.05.2019).
26. Ерл Каслдайн. Вивчаємо JQUERY. Новий стиль програмування на JavaScript. Санкт-Петербург: Пітер, 2011. 366 с.

27. Ерік А. Мейер CSS-каскадні таблиці стилів. Москва: Символ, 2006. 572 с.
28. Крістіан Д. Філіп Ч. Ajax і PHP. Москва: Символ, 2007. 332 с.
29. Люк Веллінг, Лора Томсон Розробка Web-додатків за допомогою PHP і MySQL. Москва: Вільямс, 2005. 876 с.
30. WooCommerce – найкраще рішення для інтернет-магазину на WordPress URL: http://linkomania.com.ua/woocommerce-najkrashhe-rishennya-dlya-internet-magazinu-na-wordpress/ (дата звернення: 21.05.2019).

Додаток А

Файл woocommerce.php

```
<?php
/**
 * Plugin Name: WooCommerce
 * Plugin URI: https://woocommerce.com/
 * Description: An eCommerce toolkit that helps you sell
anything. Beautifully.
 * Version: 3.6.3
 * Author: Automattic
 * Author URI: https://woocommerce.com
 * Text Domain: woocommerce
 * Domain Path: /i18n/languages/
 *
 * @package WooCommerce
 */

defined( 'ABSPATH' ) || exit;

// Define WC_PLUGIN_FILE.
if ( ! defined( 'WC_PLUGIN_FILE' ) ) {
     define( 'WC_PLUGIN_FILE', __FILE__ );
}

// Include the main WooCommerce class.
if ( ! class_exists( 'WooCommerce' ) ) {
     include_once dirname( __FILE__ ) . '/includes/class-
woocommerce.php';
}

/**
 * Returns the main instance of WC.
 *
 * @since  2.1
 * @return WooCommerce
 */
function WC() { // phpcs:ignore
WordPress.NamingConventions.ValidFunctionName.FunctionNameInvali
d
     return WooCommerce::instance();
}

// Global for backwards compatibility.
$GLOBALS['woocommerce'] = WC();
```

Файл class-wc-payment-gateways.php

```
<?php
/**
 * WooCommerce Payment Gateways
 *
 * Loads payment gateways via hooks for use in the store.
 *
 * @version 2.2.0
 * @package WooCommerce/Classes/Payment
 */

defined( 'ABSPATH' ) || exit;

/**
 * Payment gateways class.
 */
class WC_Payment_Gateways {

     /**
      * Payment gateway classes.
      *
      * @var array
      */
     public $payment_gateways = array();

     /**
      * The single instance of the class.
      *
      * @var WC_Payment_Gateways
      * @since 2.1.0
      */
     protected static $_instance = null;

     /**
      * Main WC_Payment_Gateways Instance.
      *
      * Ensures only one instance of WC_Payment_Gateways is
loaded or can be loaded.
      *
      * @since 2.1
      * @return WC_Payment_Gateways Main instance
      */
     public static function instance() {
          if ( is_null( self::$_instance ) ) {
               self::$_instance = new self();
          }
          return self::$_instance;
```

```
    }

    /**
     * Cloning is forbidden.
     *
     * @since 2.1
     */
    public function __clone() {
        wc_doing_it_wrong( __FUNCTION__, __( 'Cloning is
forbidden.', 'woocommerce' ), '2.1' );
    }

    /**
     * Unserializing instances of this class is forbidden.
     *
     * @since 2.1
     */
    public function __wakeup() {
        wc_doing_it_wrong( __FUNCTION__, __( 'Unserializing
instances of this class is forbidden.', 'woocommerce' ), '2.1'
);
    }

    /**
     * Initialize payment gateways.
     */
    public function __construct() {
        $this->init();
    }

    /**
     * Load gateways and hook in functions.
     */
    public function init() {
        $load_gateways = array(
            'WC_Gateway_BACS',
            'WC_Gateway_Cheque',
            'WC_Gateway_COD',
            'WC_Gateway_Paypal',
        );

        // Filter.
        $load_gateways = apply_filters(
'woocommerce_payment_gateways', $load_gateways );

        // Get sort order option.
        $ordering  = (array) get_option(
'woocommerce_gateway_order' );
        $order_end = 999;

        // Load gateways in order.
        foreach ( $load_gateways as $gateway ) {
```

```
                if ( is_string( $gateway ) && class_exists(
$gateway ) ) {
                    $gateway = new $gateway();
                }

                // Gateways need to be valid and extend
WC_Payment_Gateway.
                if ( ! is_a( $gateway, 'WC_Payment_Gateway' ) ) {
                    continue;
                }

                if ( isset( $ordering[ $gateway->id ] ) &&
is_numeric( $ordering[ $gateway->id ] ) ) {
                    // Add in position.
                    $this->payment_gateways[ $ordering[ $gateway-
>id ] ] = $gateway;
                } else {
                    // Add to end of the array.
                    $this->payment_gateways[ $order_end ] =
$gateway;
                    $order_end++;
                }
            }

            ksort( $this->payment_gateways );
        }

        /**
         * Get gateways.
         *
         * @return array
         */
        public function payment_gateways() {
            $_available_gateways = array();

            if ( count( $this->payment_gateways ) > 0 ) {
                foreach ( $this->payment_gateways as $gateway ) {
                    $_available_gateways[ $gateway->id ] =
$gateway;
                }
            }

            return $_available_gateways;
        }

        /**
         * Get array of registered gateway ids
         *
         * @since 2.6.0
         * @return array of strings
         */
        public function get_payment_gateway_ids() {
            return wp_list_pluck( $this->payment_gateways, 'id' );
```

```
     }

     /**
      * Get available gateways.
      *
      * @return array
      */
     public function get_available_payment_gateways() {
          $_available_gateways = array();

          foreach ( $this->payment_gateways as $gateway ) {
               if ( $gateway->is_available() ) {
                    if ( ! is_add_payment_method_page() ) {
                         $_available_gateways[ $gateway->id ] =
$gateway;
                    } elseif ( $gateway->supports(
'add_payment_method' ) || $gateway->supports( 'tokenization' ) )
{
                         $_available_gateways[ $gateway->id ] =
$gateway;
                    }
               }
          }

          return array_filter( (array) apply_filters(
'woocommerce_available_payment_gateways', $_available_gateways
), array( $this, 'filter_valid_gateway_class' ) );
     }

     /**
      * Callback for array filter. Returns true if gateway is of
correct type.
      *
      * @since 3.6.0
      * @param object $gateway Gateway to check.
      * @return bool
      */
     protected function filter_valid_gateway_class( $gateway ) {
          return $gateway && is_a( $gateway, 'WC_Payment_Gateway'
);
     }

     /**
      * Set the current, active gateway.
      *
      * @param array $gateways Available payment gateways.
      */
     public function set_current_gateway( $gateways ) {
          // Be on the defensive.
          if ( ! is_array( $gateways ) || empty( $gateways ) ) {
               return;
          }
```

```
          $current_gateway = false;
          $current         = WC()->session->get(
'chosen_payment_method' );

          if ( $current && isset( $gateways[ $current ] ) ) {
               $current_gateway = $gateways[ $current ];
          }

          if ( ! $current_gateway ) {
               $current_gateway = current( $gateways );
          }

          // Ensure we can make a call to set_current() without
triggering an error.
          if ( $current_gateway && is_callable( array(
$current_gateway, 'set_current' ) ) ) {
               $current_gateway->set_current();
          }
     }

     /**
      * Save options in admin.
      */
     public function process_admin_options() {
          $gateway_order = isset( $_POST['gateway_order'] ) ?
wc_clean( wp_unslash( $_POST['gateway_order'] ) ) : ''; // WPCS:
input var ok, CSRF ok.
          $order         = array();

          if ( is_array( $gateway_order ) && count(
$gateway_order ) > 0 ) {
               $loop = 0;
               foreach ( $gateway_order as $gateway_id ) {
                    $order[ esc_attr( $gateway_id ) ] = $loop;
                    $loop++;
               }
          }

          update_option( 'woocommerce_gateway_order', $order );
     }
}
```

Файл class-wc-data-store.php

```
<?php
/**
 * WC Data Store.
 *
 * @package WooCommerce\Classes
 * @since   3.0.0
```

```
 * @version 3.0.0
 */

defined( 'ABSPATH' ) || exit;

/**
 * Data store class.
 */
class WC_Data_Store {

     /**
      * Contains an instance of the data store class that we are
working with.
      *
      * @var WC_Data_Store
      */
     private $instance = null;

     /**
      * Contains an array of default WC supported data stores.
      * Format of object name => class name.
      * Example: 'product' => 'WC_Product_Data_Store_CPT'
      * You can also pass something like product_<type> for
product stores and
      * that type will be used first when available, if a store
is requested like
      * this and doesn't exist, then the store would fall back to
'product'.
      * Ran through `woocommerce_data_stores`.
      *
      * @var array
      */
     private $stores = array(
          'coupon'                => 'WC_Coupon_Data_Store_CPT',
          'customer'              => 'WC_Customer_Data_Store',
          'customer-download'     =>
'WC_Customer_Download_Data_Store',
          'customer-download-log' =>
'WC_Customer_Download_Log_Data_Store',
          'customer-session'      =>
'WC_Customer_Data_Store_Session',
          'order'                 => 'WC_Order_Data_Store_CPT',
          'order-refund'          =>
'WC_Order_Refund_Data_Store_CPT',
          'order-item'            => 'WC_Order_Item_Data_Store',
          'order-item-coupon'     =>
'WC_Order_Item_Coupon_Data_Store',
          'order-item-fee'        =>
'WC_Order_Item_Fee_Data_Store',
          'order-item-product'    =>
'WC_Order_Item_Product_Data_Store',
          'order-item-shipping'   =>
'WC_Order_Item_Shipping_Data_Store',
```

```
            'order-item-tax'        =>
'WC_Order_Item_Tax_Data_Store',
            'payment-token'         =>
'WC_Payment_Token_Data_Store',
            'product'               => 'WC_Product_Data_Store_CPT',
            'product-grouped'       =>
'WC_Product_Grouped_Data_Store_CPT',
            'product-variable'      =>
'WC_Product_Variable_Data_Store_CPT',
            'product-variation'     =>
'WC_Product_Variation_Data_Store_CPT',
            'shipping-zone'         =>
'WC_Shipping_Zone_Data_Store',
            'webhook'               => 'WC_Webhook_Data_Store',
        );

        /**
         * Contains the name of the current data store's class name.
         *
         * @var string
         */
        private $current_class_name = '';

        /**
         * The object type this store works with.
         *
         * @var string
         */
        private $object_type = '';


        /**
         * Tells WC_Data_Store which object (coupon, product, order,
etc)
         * store we want to work with.
         *
         * @throws Exception When validation fails.
         * @param string $object_type Name of object.
         */
        public function __construct( $object_type ) {
            $this->object_type = $object_type;
            $this->stores      = apply_filters(
'woocommerce_data_stores', $this->stores );

            // If this object type can't be found, check to see if
we can load one
            // level up (so if product-type isn't found, we try
product).
            if ( ! array_key_exists( $object_type, $this->stores )
) {
                $pieces      = explode( '-', $object_type );
                $object_type = $pieces[0];
            }
```

```
          if ( array_key_exists( $object_type, $this->stores ) )
{
               $store = apply_filters( 'woocommerce_' .
$object_type . '_data_store', $this->stores[ $object_type ] );
               if ( is_object( $store ) ) {
                    if ( ! $store instanceof
WC_Object_Data_Store_Interface ) {
                         throw new Exception( __( 'Invalid data
store.', 'woocommerce' ) );
                    }
                    $this->current_class_name = get_class( $store
);
                    $this->instance           = $store;
               } else {
                    if ( ! class_exists( $store ) ) {
                         throw new Exception( __( 'Invalid data
store.', 'woocommerce' ) );
                    }
                    $this->current_class_name = $store;
                    $this->instance           = new $store();
               }
          } else {
               throw new Exception( __( 'Invalid data store.',
'woocommerce' ) );
          }
     }

     /**
      * Only store the object type to avoid serializing the data
store instance.
      *
      * @return array
      */
     public function __sleep() {
          return array( 'object_type' );
     }

     /**
      * Re-run the constructor with the object type.
      *
      * @throws Exception When validation fails.
      */
     public function __wakeup() {
          $this->__construct( $this->object_type );
     }

     /**
      * Loads a data store.
      *
      * @param string $object_type Name of object.
      *
      * @since 3.0.0
```

```
 * @throws Exception When validation fails.
 * @return WC_Data_Store
 */
public static function load( $object_type ) {
     return new WC_Data_Store( $object_type );
}

/**
 * Returns the class name of the current data store.
 *
 * @since 3.0.0
 * @return string
 */
public function get_current_class_name() {
     return $this->current_class_name;
}

/**
 * Reads an object from the data store.
 *
 * @since 3.0.0
 * @param WC_Data $data WooCommerce data instance.
 */
public function read( &$data ) {
     $this->instance->read( $data );
}

/**
 * Create an object in the data store.
 *
 * @since 3.0.0
 * @param WC_Data $data WooCommerce data instance.
 */
public function create( &$data ) {
     $this->instance->create( $data );
}

/**
 * Update an object in the data store.
 *
 * @since 3.0.0
 * @param WC_Data $data WooCommerce data instance.
 */
public function update( &$data ) {
     $this->instance->update( $data );
}

/**
 * Delete an object from the data store.
 *
 * @since 3.0.0
 * @param WC_Data $data WooCommerce data instance.
```

```
     * @param array   $args Array of args to pass to the delete
method.
     */
    public function delete( &$data, $args = array() ) {
        $this->instance->delete( $data, $args );
    }

    /**
     * Data stores can define additional functions (for example,
coupons have
     * some helper methods for increasing or decreasing usage).
This passes
     * through to the instance if that function exists.
     *
     * @since 3.0.0
     * @param string $method     Method.
     * @param mixed  $parameters Parameters.
     * @return mixed
     */
    public function __call( $method, $parameters ) {
        if ( is_callable( array( $this->instance, $method ) ) )
{
            $object = array_shift( $parameters );
            return call_user_func_array( array( $this-
>instance, $method ), array_merge( array( &$object ),
$parameters ) );
        }
    }
}
```

Додаток Б

Файл customizer-preview.css

```
/*
 * Customizer stylesheet.
 */
.pirate-forms-placeholder .header-primary {
     z-index: 100;
     border: 2px dashed #8a8a8a;
     background: rgba(60, 60, 60, 0.9);
}

.pirate-forms-placeholder .card {
     position: relative;
     z-index: 50;
     border: 2px dashed #8a8a8a;
     background: rgba(255, 255, 255, 0.1);
}

.pirate-forms-placeholder .card:after {
     position: absolute;
     z-index: -1;
     top: 0;
     right: 0;
     bottom: 0;
     left: 0;
     border-radius: 6px;
     content: " ";
}

.pirate-forms-placeholder .pirate-forms-placeholder-overlay {
     display: inline-block;
     position: absolute;
     top: 0;
     height: 100%;
     padding: 0 30px;
     color: #fff;
     text-align: center;
}

.pirate-forms-placeholder .pirate-forms-placeholder-
overlay:before {
     display: inline-block;
     width: 0;
     height: 100%;
     vertical-align: middle;
     content: "";
```

```
}

.pirate-forms-placeholder .pirate-forms-placeholder-overlay
.placeholder-text {
    padding: 15px 15px;
    border-radius: 6px;
    background: rgba(0, 0, 0, 0.5);
    font-weight: 100;
}

.pirate-forms-placeholder .pirate-forms-placeholder-overlay
.placeholder-text .customize-partial-edit-shortcut button {
    top: -25px;
}

.pirate-forms-placeholder .pirate_forms_wrap form,
.pirate-forms-placeholder .pirate_forms_wrap div {
    z-index: -2;
    opacity: 0.3;
}

.pirate-forms-placeholder .pirate_forms_wrap form button,
.pirate-forms-placeholder .pirate_forms_wrap div button {
    border: 2px dashed #8a8a8a;
    background-color: #333;
    box-shadow: none;
}

.pirate-forms-placeholder .pirate-forms-placeholder-align {
    display: inline-block;
    width: 100%;
    margin: 0 -7px;
    vertical-align: middle;
}

.pirate-forms-placeholder .pirate-forms-placeholder-align:before
{
    display: inline-block;
    float: left;
    height: 100%;
    vertical-align: middle;
    content: "";
}

.customize-partial-edit-shortcut-custom_header {
    z-index: 9999;
}

.customize-partial-edit-shortcut-hestia_big_title_background {
    top: 80px;
    right: 80px;
}
```

```
.customize-partial-edit-shortcut-
hestia_placeholder_sidebar_woocommerce,
.customize-partial-edit-shortcut-hestia_placeholder_sidebar_1 {
    position: relative;
    top: -10px;
    left: 45px;
}

.customize-partial-edit-shortcut-hestia_feature_thumbnail,
.customize-partial-edit-shortcut-hestia_copyright_alignment,
.customize-partial-edit-shortcut-hestia_top_bar_alignment,
.customize-partial-edit-shortcut-hestia_ribbon_background,
.customize-partial-edit-shortcut-hestia_subscribe_background,
.customize-partial-edit-shortcut-hestia_contact_background {
    z-index: 1029;
    top: 10px;
    right: 10px;
}

.customize-partial-edit-shortcut-hestia_features_hide,
.customize-partial-edit-shortcut-hestia_about_hide,
.customize-partial-edit-shortcut-hestia_shop_hide,
.customize-partial-edit-shortcut-hestia_portfolio_hide,
.customize-partial-edit-shortcut-hestia_team_hide,
.customize-partial-edit-shortcut-hestia_pricing_hide,
.customize-partial-edit-shortcut-hestia_ribbon_hide,
.customize-partial-edit-shortcut-hestia_testimonials_hide,
.customize-partial-edit-shortcut-hestia_clients_bar_hide,
.customize-partial-edit-shortcut-hestia_subscribe_hide,
.customize-partial-edit-shortcut-hestia_blog_hide,
.customize-partial-edit-shortcut-hestia_contact_hide {
    z-index: 999;
    top: 0;
    left: 40px;
    opacity: 1;
}

.customize-partial-edit-shortcut-hestia_contact_form_shortcode,
.customize-partial-edit-shortcut-hestia_contact_content_new,
.customize-partial-edit-shortcut-hestia_shop_items,
.customize-partial-edit-shortcut-hestia_blog_items,
.customize-partial-edit-shortcut-hestia_shop_categories,
.customize-partial-edit-shortcut-hestia_shop_order,
.customize-partial-edit-shortcut-hestia_shop_shortcode,
.customize-partial-edit-shortcut-hestia_portfolio_items,
.customize-partial-edit-shortcut-hestia_portfolio_boxes_type,
.customize-partial-edit-shortcut-hestia_blog_categories,
body.blog .customize-partial-edit-shortcut-hestia_blog_hide,
.customize-partial-edit-shortcut-hestia_slider_alignment,
.customize-partial-edit-shortcut-hestia_big_title_background {
    display: none;
}
```

```
.customize-partial-edit-shortcut-primary-menu {
    float: right;
    position: relative;
    z-index: 10000;
    top: 0;
    right: -30px;
}

/**
 * Features section - selective refresh style
 */
.hestia-features:not(.is-shortcode),
.hestia-about:not(.is-shortcode),
.hestia-shop:not(.related):not(.is-shortcode),
.hestia-work:not(.is-shortcode),
.hestia-team:not(.is-shortcode),
.hestia-pricing:not(.is-shortcode),
.hestia-ribbon:not(.is-shortcode),
.hestia-testimonials:not(.is-shortcode),
.hestia-subscribe:not(#subscribe-on-blog):not(.is-shortcode),
body.page .hestia-blogs:not(.is-shortcode),
.hestia-clients-bar:not(.is-shortcode),
.hestia-contact:not(.is-shortcode) {
    position: relative;
}

.hestia-features:not(.is-shortcode):before,
.hestia-about:not(.is-shortcode):before,
.hestia-shop:not(.related):not(.is-shortcode):before,
.hestia-work:not(.is-shortcode):before,
.hestia-team:not(.is-shortcode):before,
.hestia-pricing:not(.is-shortcode):before,
.hestia-ribbon:not(.is-shortcode):before,
.hestia-testimonials:not(.is-shortcode):before,
.hestia-subscribe:not(#subscribe-on-blog):not(.is-
shortcode):before,
body.page .hestia-blogs:not(.is-shortcode):before,
.hestia-clients-bar:not(.is-shortcode):before,
.hestia-contact:not(.is-shortcode):before {
    position: absolute;
    z-index: 100;
    top: 20px;
    right: 20px;
    bottom: 20px;
    left: 20px;
    border: 2px dashed rgba(121, 121, 121, 0.22);
    content: " ";
    -webkit-transition: 0.3s ease;
    transition: 0.3s ease;
    pointer-events: none;
}

.hestia-features:not(.is-shortcode):hover:before,
```

```
.hestia-about:not(.is-shortcode):hover:before,
.hestia-shop:not(.related):not(.is-shortcode):hover:before,
.hestia-work:not(.is-shortcode):hover:before,
.hestia-team:not(.is-shortcode):hover:before,
.hestia-pricing:not(.is-shortcode):hover:before,
.hestia-ribbon:not(.is-shortcode):hover:before,
.hestia-testimonials:not(.is-shortcode):hover:before,
.hestia-subscribe:not(#subscribe-on-blog):not(.is-
shortcode):hover:before,
body.page .hestia-blogs:not(.is-shortcode):hover:before,
.hestia-clients-bar:not(.is-shortcode):hover:before,
.hestia-contact:not(.is-shortcode):hover:before {
    border-color: #00a0d0;
}

.hestia-features-title-area,
.hestia-shop-title-area,
.hestia-team-title-area,
.hestia-pricing-title-area,
.hestia-testimonials-title-area,
.hestia-subscribe-title-area,
.hestia-blogs-title-area,
.hestia-contact-title-area,
.hestia-portfolio-title-area {
    -webkit-transition: 0.3s ease;
    transition: 0.3s ease;
}

.hestia-features-title-area:hover .customize-partial-edit-
shortcut,
.hestia-shop-title-area:hover .customize-partial-edit-shortcut,
.hestia-team-title-area:hover .customize-partial-edit-shortcut,
.hestia-pricing-title-area:hover .customize-partial-edit-
shortcut,
.hestia-testimonials-title-area:hover .customize-partial-edit-
shortcut,
.hestia-subscribe-title-area:hover .customize-partial-edit-
shortcut,
.hestia-blogs-title-area:hover .customize-partial-edit-shortcut,
.hestia-contact-title-area:hover .customize-partial-edit-
shortcut,
.hestia-portfolio-title-area:hover .customize-partial-edit-
shortcut {
    opacity: 1;
}

.hestia-features-title-area .customize-partial-edit-shortcut,
.hestia-shop-title-area .customize-partial-edit-shortcut,
.hestia-team-title-area .customize-partial-edit-shortcut,
.hestia-pricing-title-area .customize-partial-edit-shortcut,
.hestia-testimonials-title-area .customize-partial-edit-
shortcut,
.hestia-subscribe-title-area .customize-partial-edit-shortcut,
```

```
.hestia-blogs-title-area .customize-partial-edit-shortcut,
.hestia-contact-title-area .customize-partial-edit-shortcut,
.hestia-portfolio-title-area .customize-partial-edit-shortcut {
    opacity: 0;
    -webkit-transition: 0.3s ease;
    transition: 0.3s ease;
}

.hestia-features-title-area .customize-partial-edit-shortcut
button,
.hestia-shop-title-area .customize-partial-edit-shortcut button,
.hestia-team-title-area .customize-partial-edit-shortcut button,
.hestia-pricing-title-area .customize-partial-edit-shortcut
button,
.hestia-testimonials-title-area .customize-partial-edit-shortcut
button,
.hestia-subscribe-title-area .customize-partial-edit-shortcut
button,
.hestia-blogs-title-area .customize-partial-edit-shortcut
button,
.hestia-contact-title-area .customize-partial-edit-shortcut
button,
.hestia-portfolio-title-area .customize-partial-edit-shortcut
button {
    margin: 0;
}

.hestia-features-content,
.hestia-about-content,
.hestia-team-content,
.hestia-table-one,
.hestia-table-two,
.hestia-ribbon-content,
.hestia-testimonials-content,
.hestia-blog-content,
.hestia-clients-bar,
.hestia-contact .hestia-description,
.hestia-contact-form-col {
    position: relative;
    -webkit-transition: 0.3s ease;
    transition: 0.3s ease;
}

.hestia-features-content .customize-partial-edit-shortcut,
.hestia-about-content .customize-partial-edit-shortcut,
.hestia-team-content .customize-partial-edit-shortcut,
.hestia-table-one .customize-partial-edit-shortcut,
.hestia-table-two .customize-partial-edit-shortcut,
.hestia-ribbon-content .customize-partial-edit-shortcut,
.hestia-testimonials-content .customize-partial-edit-shortcut,
.hestia-blog-content .customize-partial-edit-shortcut,
.hestia-clients-bar .customize-partial-edit-shortcut,
```

```
.hestia-contact .hestia-description .customize-partial-edit-
shortcut,
.hestia-contact-form-col .customize-partial-edit-shortcut {
    opacity: 0;
    -webkit-transition: 0.3s ease;
    transition: 0.3s ease;
}

.hestia-features-content .customize-partial-edit-shortcut
button,
.hestia-about-content .customize-partial-edit-shortcut button,
.hestia-team-content .customize-partial-edit-shortcut button,
.hestia-table-one .customize-partial-edit-shortcut button,
.hestia-table-two .customize-partial-edit-shortcut button,
.hestia-ribbon-content .customize-partial-edit-shortcut button,
.hestia-testimonials-content .customize-partial-edit-shortcut
button,
.hestia-blog-content .customize-partial-edit-shortcut button,
.hestia-clients-bar .customize-partial-edit-shortcut button,
.hestia-contact .hestia-description .customize-partial-edit-
shortcut button,
.hestia-contact-form-col .customize-partial-edit-shortcut button
{
    margin: 0;
}

.hestia-features-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
.hestia-features-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button,
.hestia-about-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
.hestia-about-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button,
.hestia-team-content .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_one_title button,
.hestia-team-content .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_two_title button,
.hestia-table-one .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_one_title button,
.hestia-table-one .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_two_title button,
.hestia-table-two .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_one_title button,
.hestia-table-two .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_two_title button,
.hestia-ribbon-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
```

```
.hestia-ribbon-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button,
.hestia-testimonials-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
.hestia-testimonials-content .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button,
.hestia-blog-content .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_one_title button,
.hestia-blog-content .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_two_title button,
.hestia-clients-bar .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_one_title button,
.hestia-clients-bar .customize-partial-edit-shortcut.customize-
partial-edit-shortcut-hestia_pricing_table_two_title button,
.hestia-contact .hestia-description .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
.hestia-contact .hestia-description .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button,
.hestia-contact-form-col .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_one_title button,
.hestia-contact-form-col .customize-partial-edit-
shortcut.customize-partial-edit-shortcut-
hestia_pricing_table_two_title button {
    left: 0;
}

.hestia-features-content:hover .customize-partial-edit-shortcut,
.hestia-about-content:hover .customize-partial-edit-shortcut,
.hestia-team-content:hover .customize-partial-edit-shortcut,
.hestia-table-one:hover .customize-partial-edit-shortcut,
.hestia-table-two:hover .customize-partial-edit-shortcut,
.hestia-ribbon-content:hover .customize-partial-edit-shortcut,
.hestia-testimonials-content:hover .customize-partial-edit-
shortcut,
.hestia-blog-content:hover .customize-partial-edit-shortcut,
.hestia-clients-bar:hover .customize-partial-edit-shortcut,
.hestia-contact .hestia-description:hover .customize-partial-
edit-shortcut,
.hestia-contact-form-col:hover .customize-partial-edit-shortcut
{
    opacity: 1;
}

.customize-partial-refreshing:before {
    position: absolute;
    z-index: 10000;
    top: 50% !important;
```

```
    right: auto !important;
    bottom: auto !important;
    left: 50% !important;
    margin: 0 auto;
    border: none !important;
    content: url(../img/ajax-loader.gif) !important;
    -webkit-transform: translate3d(-50%, 0, 0);
    transform: translate3d(-50%, 0, 0);
}
```

Файл font-sizes.css

```
/*
 * Header titles.
 */
.carousel .hestia-title {
    font-size: 67px;
}

.carousel span.sub-title {
    font-size: 18px;
}

@media (max-width: 768px) {
    .carousel .hestia-title,
    h1.hestia-title,
    .page-header.header-small .hestia-title,
    .page-header.header-small .title {
        font-size: 36px;
    }
}

/*
 * Section primary headings.
 */
h2.hestia-title,
h2.title {
    font-size: 37px;
}

@media (max-width: 768px) {
    h2.hestia-title,
    h2.title {
        font-size: 24px;
    }
}

.woocommerce div.product .product_title,
.woocommerce .related.products h2 {
    font-size: 32px;
}
```

```
.subscribe-line h3.hestia-title,
#comments .hestia-title,
.woocommerce .comment-reply-title,
.woocommerce-cart .blog-post h1.hestia-title,
.woocommerce-checkout .blog-post h1.hestia-title {
    font-size: 27px;
}

h5.description,
h5.subscribe-description {
    font-size: 18px;
}

.subscribe-line .description {
    font-size: 14px;
}

/*
 * Section secondary headings.
 */
.woocommerce-cart .blog-post h1.hestia-title,
.woocommerce-checkout .blog-post h1.hestia-title {
    font-size: 27px;
}

.hestia-work .card-title,
.woocommerce .cart-collaterals h2,
.woocommerce .cross-sells h2,
.woocommerce.single-product .summary .price,
.woocommerce-checkout .blog-post .section form.woocommerce-
checkout h3:not(#ship-to-different-address) {
    font-size: 23px;
}

h4.card-title,
.hestia-info h4.info-title,
.footer .widget h5,
section.contactus h4.hestia-title {
    font-size: 18px;
}

.blog .card-blog .card-title,
.archive .card-blog .card-title,
.blog .hestia-blog-featured-card .card-title {
    font-size: 25px;
}

/*
 * Section content.
 */
h6.category {
    font-size: 12px;
```

```
}

.woocommerce .product .card-product .card-description p,
.card-description,
section.pricing p.text-gray,
.woocommerce.single-product .woocommerce-product-details__short-
description {
     font-size: 14px;
}

.hestia-features .hestia-info p {
     font-size: 16px;
}

/*
 * Post/page headings.
 */
h1,
.single-post-wrap article h1,
.page-content-wrap h1,
.hestia-title.title-in-content {
     font-size: 42px;
}

@media (max-width: 768px) {
     .hestia-title.title-in-content {
          font-size: 36px;
     }
}

h2,
.single-post-wrap article h2,
.page-content-wrap h2 {
     font-size: 37px;
}

h3,
.single-post-wrap article h3,
.page-content-wrap h3 {
     font-size: 32px;
}

h4,
.single-post-wrap article h4,
.page-content-wrap h4 {
     font-size: 27px;
}

h5,
.single-post-wrap article h5,
.page-content-wrap h5 {
     font-size: 23px;
}
```

```
h6,
.single-post-wrap article h6,
.page-content-wrap h6 {
     font-size: 18px;
}

@media (max-width: 768px) {
     h1,
     .single-post-wrap article h1,
     .page-content-wrap h1,
     .page-template-template-fullwidth article h1 {
          font-size: 36px;
     }

     h2,
     .single-post-wrap article h2,
     .page-content-wrap h2,
     .page-template-template-fullwidth article h2 {
          font-size: 32px;
     }

     h3,
     .single-post-wrap article h3,
     .page-content-wrap h3,
     .page-template-template-fullwidth article h3 {
          font-size: 28px;
     }

     h4,
     .single-post-wrap article h4,
     .page-content-wrap h4,
     .page-template-template-fullwidth article h4 {
          font-size: 24px;
     }

     h5,
     .single-post-wrap article h5,
     .page-content-wrap h5,
     .page-template-template-fullwidth article h5 {
          font-size: 21px;
     }

     h6,
     .single-post-wrap article h6,
     .page-content-wrap h6,
     .page-template-template-fullwidth article h6 {
          font-size: 18px;
     }
}

/*
 * Post/page content.
```

```
 */
.single-post-wrap article p,
.page-content-wrap p,
.single-post-wrap article ul,
.page-content-wrap ul,
.single-post-wrap article ol,
.page-content-wrap ol,
.single-post-wrap article dl,
.page-content-wrap dl,
.single-post-wrap article table,
.page-content-wrap table,
.page-template-template-fullwidth article p,
.page-template-template-fullwidth article ul,
.page-template-template-fullwidth article ol,
.page-template-template-fullwidth article dl,
.page-template-template-fullwidth article table {
    font-size: 18px;
}

@media (max-width: 768px) {
    .single-post-wrap article p,
    .page-content-wrap p,
    .single-post-wrap article ul,
    .page-content-wrap ul,
    .single-post-wrap article ol,
    .page-content-wrap ol,
    .single-post-wrap article dl,
    .page-content-wrap dl,
    .single-post-wrap article table,
    .page-content-wrap table,
    .page-template-template-fullwidth article p,
    .page-template-template-fullwidth article ul,
    .page-template-template-fullwidth article ol,
    .page-template-template-fullwidth article dl,
    .page-template-template-fullwidth article table {
        font-size: 16px;
    }
}

/*
 * Body.
 */
body {
    font-size: 14px;
}

/*
 * Menu.
 */
.navbar a {
    font-size: 12px;
}
```