ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ, ПРАВА ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

Пояснювальна записка

до кваліфікаційної роботи на здобуття ступеня вищої освіти бакалавр

на тему: **«Розроблення інформаційного сервісу підтримки продажу комп'ютерної техніки»**

Виконав: здобувач вищої освіти
за освітньо-професійною
програмою
Інформаційні управляючі системи
спеціальності 126 Інформаційні
системи та технології
ступеня вищої освіти бакалавр
групи 126ІСТ_бд_2020
Турбай Є. О.
Керівник: Одарущенко О. М.
Рецензент: Ковальчук С. Б.

Полтава – 2024 року

ВСТУП

Розвиток інформаційних технологій останнім часом зазнав значного прогресу, що значно вплинуло на способи ведення бізнесу. У зв'язку з цим, багато компаній, що спеціалізуються на продажі комп'ютерної техніки, шукають нові способи підтримки продажів за допомогою інформаційних сервісів. Наявність такого сервісу стає ключовою для підтримки клієнтів та забезпечення їхнього задоволення від придбаних товарів. Він може включати в себе широкий спектр функцій, починаючи від консультацій та рекомендацій щодо вибору техніки, і закінчуючи післяпродажним обслуговуванням та вирішенням проблем.

Основна мета даної дипломної роботи полягатиме у розробці та впровадженні інформаційного сервісу підтримки продажу комп'ютерної техніки. Дослідження буде спрямовано на визначення оптимальних методів та інструментів для створення такого сервісу, а також на розробку його функціональності та інтерфейсу з метою максимального задоволення потреб клієнтів. Методи системного аналізу та моделювання будуть використані для досягнення цієї мети.

Об'єктом дослідження виступатиме процес розробки інформаційного сервісу підтримки продажу комп'ютерної техніки, а предметом його функціональні можливості та архітектура. Важливість цієї теми полягає у тому, що від якості та ефективності такого сервісу залежить успішність бізнесу, його конкурентоспроможність та здатність задовольняти потреби сучасного ринку.

Актуальність теми: згідно з дослідженнями, ринок комп’ютерної техніки постійно зростає. Прогнозується, що до 2025 року його обсяг сягне 500 млрд. дол. США. Це створює великі можливості для компаній, які пропонують комп’ютерну техніку.

Одним із ключових факторів успіху в цьому конкурентному середовищі є ефективна підтримка продажу. Інформаційний сервіс, який надає повну та

актуальну інформацію про продукти, полегшує процес вибору та покупки для клієнтів, а також підвищує рівень лояльності.

Мета роботи: розробити інформаційний сервіс підтримки продажу комп'ютерної техніки, який буде відповідати потребам як клієнтів, так і продавців.

Об'єкт дослідження - процеси розроблення інформаційних сервісів підтримки продажу товарів.

Предмет дослідження: архітектура та функціонал інформаційного сервісу підтримки продажу комп'ютерної техніки.

Методи дослідження: системний аналіз (розділи 1, 2), моделювання (розділ 3).

Інформаційна база, що використовувалася: публікації в періодичних виданнях, вебресурси.

вебресурси, документація з розробки програмного забезпечення.

Практична значущість: Розробка та впровадження інформаційного сервісу підтримки продажу комп'ютерної техніки, підвищення ефективності продажу, збільшення рівня лояльності клієнтів

Робота складається зі вступу, трьох розділів, висновків та списку літератури. Обсяг роботи становить 55 сторінок, 2 таблиці, 25 рисунків, 2 додатки.

РОЗДІЛ 1
АНАЛІЗ ТЕХНОЛОГІЙ РОЗРОБКИ ІНФОРМАЦІЙНОГО СЕРВІСУ ПІДТРИМКИ ПРОДАЖУ

1.1. Історія появи та розвитку перших вебсайтів

Історія появи та розвитку перших вебсайтів є захоплюючим шляхом, який відображає еволюцію Інтернету та вебтехнологій. Від скромних стартів до сучасних динамічних та інтерактивних платформ, цей шлях відображає неабиякий прогрес у сфері технологій та комунікацій.

Перші кроки у світі вебсайтів були зроблені в 1990 р. Тімом Бернерс-Лі в Європейському організації для ядерних досліджень (CERN). Тут було створено перший вебсервер та веббраузер, що відкрило шлях до створення та перегляду перших вебсторінок (рис. 1.1).

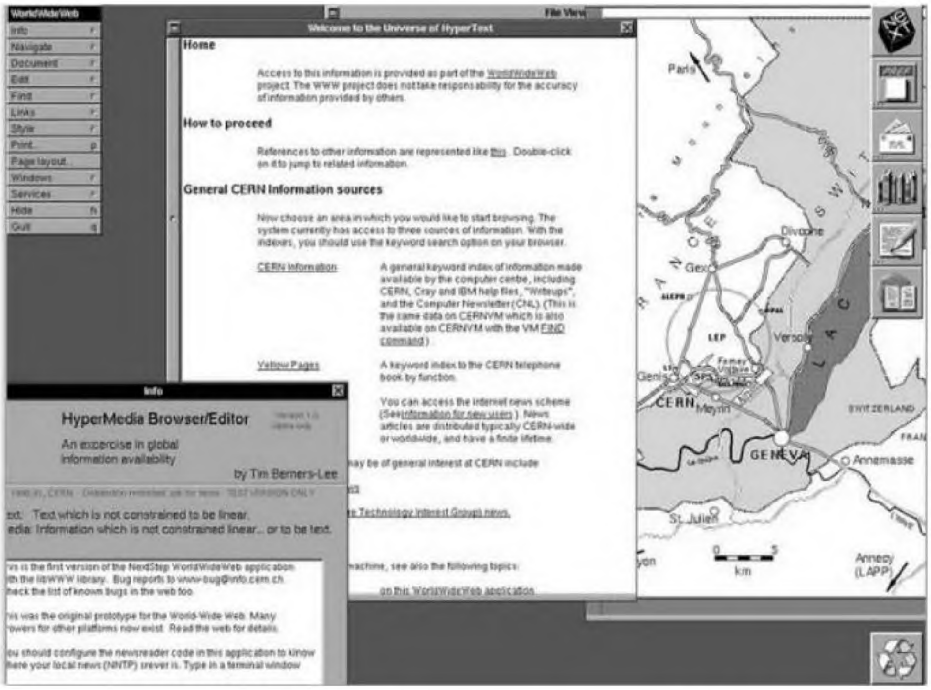

Рисунок 1.1 – Скріншот першого в світі сайту

На його думку, проєкт повинен був допомагати вченим шукати та ділитися інформацією. Ідею глобальної системи комунікації World Wide Web Тім Бернерс-Лі презентував ще у 1989 р. "Розпливчасто, але цікаво",

прокоментував концепцію його начальник Майк Сендалл, але дав добро на її втілення [1].

Історія українського сегменту інтернету почалася 19 грудня 1990 р., коли п'ятьом київським інженерам вдалося під'єднатися до мережі Інтернет. Навесні 1991 р. була створена компанія "Технософт", вона офіційно отримала статус інтернет-сервіс провайдера.

З 1991 р. Інтернет почав поширюватися серед наукових та академічних колективів, а в 1993 р. була запущена перша версія веббраузера Mosaic, що сприяло масовому розповсюдженню вебсайтів. Перші вебсторінки були переважно текстовими, з мінімальними графічними елементами (рис. 1.2).

Рисунок 1.2 – Перший веббраузер

Період з середини 90-х до початку 2000-х рр. відзначився вибухом популярності Інтернету. З'явилися перші пошукові системи, такі як Yahoo!, що значно полегшили пошук та навігацію в мережі. В цей період вебсайти стали більш багатофункціональними та включали в себе більше мультимедійного контенту (рис. 1.3).

У 1994 р. два аспіранти Стендфордського університету, Джері Янг та Девід Філо, розпочали роботу над вебкаталогом під назвою "Jerry's Guide to the World Wide Web". Цей каталог швидко набув популярності завдяки своїй всебічності та простоті використання [2].

Yahoo! домінував в Інтернеті протягом 1990-х та початку 2000-х рр. Однак у 2010-х рр. компанія зіткнулася з жорсткою конкуренцією з боку таких компаній, як Google та Microsoft, а також з низкою проблем, включаючи кібератаки та втрату користувачів.

У 2016 р. Yahoo! продала свій основний бізнес компанії Verizon, а у 2017 р. Verizon перепродала його Apollo Global Management.

Сьогодні Yahoo! продовжує працювати як дочірня компанія Apollo Global Management, зосереджуючись на таких продуктах, як Yahoo Mail, Yahoo Finance та Yahoo Sports.

Рисунок 1.3 – Перша пошукова система Yahoo!

Розвиток технологій значною мірою змінив повсякденне життя, і одним із прикладів цього є поява інтернет магазинів. Цей термін став невід'ємною

частиною нашого словникового запасу та буденного життя, оскільки онлайн-шопінг набуває все більшої популярності. Хоча інтернет магазини зараз є звичною справою, їх історія розпочалася ще у 90-х рр. минулого століття, коли технологічний прогрес дозволив першим підприємцям втілити ідею електронної торгівлі в життя. Один із найпоказовіших прикладів – американський вебсайт Amazon, який було засновано Джеффом Безосом 5 липня 1994 р. як інтернет магазин із продажу книг [3].

Перша книга на цьому сайті, підручник з програмування, була продана в 1995 рр. У той час Amazon мав величезну базу даних з мільйонами назв книг. У самого Джеффа книги не було: отримавши заявку, він відправив її в офлайн-магазин, з яким співпрацював, і прийняв їх товар. На складі Amazon у Сіетлі було лише 2000 книг, і це були єдині, які потрібно було надіслати одержувачам найближчим часом.

Потім, коли потік замовлень почав збільшуватися, співробітникам довелося вдень обробляти запити на комп'ютері, а ввечері відправлятися на склад, щоб прийняти товар і упакувати його в фірмові коробки.

Стрімке зростання Amazon почалося після того, як магазин показали на головній сторінці популярного пошукача Yahoo! За два місяці продажі зросли до 20 000 дол. США в тиждень.

З'явилися перші інвестори і виділили 1100 мільйонів на поліпшення сервера і оновлення сайту. Це дозволило привести Amazon в гідний вигляд. На ресурсі також є фото продукту. Стало зрозуміло, що стартап відбувся.

З того часу Amazon еволюціонував у глобального гіганта електронної комерції, пропонуючи мільйони різних товарів та хмарних сервісів у різних категоріях, встановлюючи нові стандарти у сфері онлайн торгівлі.

У 2019 бренд Amazon став найдорожчим у світі й оцінювався в 415 млрд. дол. США.У 2019 компанія стала найдорожчою у світі, обійшовши Microsoft, її ціна на 7 січня 2019 становила 797 млрд. дол. США.

У сучасній епохі вебсайти стали набагато складнішими та інтерактивними. Вони використовують різноманітні технології, такі як

JavaScript, CSS та HTML, для створення динамічного контенту. Сучасні вебсайти можуть включати анімацію, відео, інтерактивні форми та багато інших функцій.

Історія перших вебсайтів свідчить про неймовірний прогрес у розвитку Інтернету та вебтехнологій. Від простих статичних сторінок до складних та інтерактивних платформ, вебсайти продовжують еволюцію та надихати на нові ідеї та можливості в галузі онлайн комунікацій та бізнесу.

1.2. Огляд сучасних технологій розробки інформаційних сервісів підтримки продажів

Розробка інформаційних сервісів підтримки (ІСП) вимагає використання сучасних технологій, котрі забезпечують швидкість, ефективність та надійність роботи системи. HTML, CSS і JavaScript є основними технологіями для розробки вебсайтів. Кожна з цих технологій відповідає за різні аспекти створення: HTML – за структуру та зміст, CSS – за оформлення, а JavaScript – за динаміку та взаємодію.

Це мова гіпертекстової розмітки документів, яка працює через систему тегів і допомагає структурувати та компонувати елементи вебсторінок. HTML каже браузеру, де має бути заголовок, текст, картинки, посилання тощо. Це каркас, на основі якого будуються всі сайти [4].

Початковою мотивацією для створення HTML була необхідність уніфікувати та структурувати документи, щоб зробити їх більш читабельними та зрозумілими. Спочатку HTML використовувався для створення простих сторінок із текстовим вмістом та гіперпосиланнями, які можна було переміщувати з одного документа в інший.

Наразі останньою версією є HTML5. Завдяки новим можливостям HTML5, розробники можуть використовувати нові елементи, які полегшують створення складних інтерфейсів. Крім того, HTML5 підтримує багато нових

API для реалізації різних функцій, таких як мультимедіа, геолокація та робота офлайн – це оновлення дозволило розробникам спростити розробку додатків за допомогою нових функцій (рис. 1.4).

Рисунок 1.4 – HTML5

В HTML5 були додані спеціальні семантичні елементи і теги, в таблиці нижче представлено ті, які використовують найчастіше:

Таблиця 1.1 – Семантичні теги в HTML5

Назва	Функції
header	Заголовок сайту або розділу. Включає в себе навігацію або логотип.
main	Основний контент сторінки, унікальний для документа.
footer	Нижня частина сайту або розділу, містить інформацію про авторські права, контакти тощо.
nav	Контейнер для навігаційних посилань (меню).
aside	Контент, який не є основною частиною сторінки, але пов'язаний з нею, зазвичай – бокова панель.
section	Дозволяє розділяти сторінку на певні секції (розділи).
summary	Клацнувши на даний заголовок, можна розгорнути та побачити додаткову інформацію.

Нова версія HTML відкриває нові можливості, дозволяє краще та ефективніше взаємодіяти з клієнтами, створювати web-додатки.

Наступним важливим інструментом, навіть необхідним, є так звана "мова стилів" CSS.

CSS – це спеціальна мова, за допомогою якої описують зовнішній вигляд сторінок сайту (як і де відображати елементи вебсторінки), написаних мовами розмітки даних [5].

CSS був створений для того, щоб відокремити стилі від структури вебдокументів, написаних на HTML. Перша версія CSS була випущена в

1996 р. і була розроблена для того, щоб зробити ваш вебсайт більш гнучким і керованим.

CSS також дозволяє використовувати стилі для різних медіа-просторів, таких як екрани, друк та мобільні пристрої. Це робить його незамінним інструментом для створення адаптивних та доступних вебсайтів, які добре виглядають на будь-якому пристрої та в будь-якому середовищі використанн [6].

Однією з головних переваг використання CSS є те, що вміст сторінки можна відокремити від дизайну. Таке розділення дозволило покращити сприйняття та доступність контенту, зробити відображення контенту більш гнучким та котрольованим у різних умовах, зробити контент більш структурованим та простим, а також усунути повтори. Власне, це і було основною метою створення даної технології (рис. 1.5).

Рисунок 1.5 – CSS

Отже, HTML використовується для структурування вмісту сторінки, а CSS для форматування цього структурованого вмісту. А поєднання HTML і CSS дозволяє створювати в документі неймовірні речі і поступово вивчаючи CSS можна в цьому переконатися. Можливо, створити деякі функції за допомогою CSS набагато простіше, ніж за допомогою JavaScript.

JavaScript – це мова програмування котра була створена з метою додання інтерактивності на сайт (рис. 1.6).

Програми цією мовою називаються скриптами. Їх можна писати прямо в коді HTML-сторінок і вони автоматично виконуватимуться при їх завантаженні. Скрипти записуються та виконуються як простий текст. Для запуску їм не потрібна спеціальна підготовка чи компілятор [7].

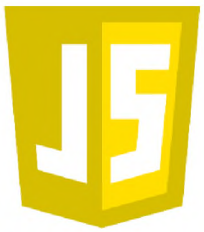

Рисунок 1.6 – JavaScript

При використанні в рамках технології HTML код JavaScript включається в HTML-код сторінки і виконується вбудованим в браузер інтерпретатором. Код JavaScript вставляється в тег script, але JavaScript є мовою сценаріїв за замовчуванням у більшості браузерів (рис. 1.7).

```
<!DOCTYPE HTML>
<html>

<body>

  <p>Текст перед скриптом...</p>

  <script>
    alert( 'Привіт, світ!' );
  </script>

  <p>...Після скрипту.</p>

</body>

</html>
```

Рисунок 1.7 – Приклад використання JavaScript

JavaScript є скриптовою мовою програмування, котра використовується для надання динаміки та інтерактивності вебсторінкам. Вона дозволяє виконувати різноманітні дії, такі як анімація, валідація даних, маніпуляція DOM (Document Object Model) та взаємодія з користувачем.

Отже, HTML5, CSS, та JavaScript є невід'ємними складовими розвитку сучасних вебтехнологій. Вони дозволяють розробникам створювати естетичні, інтерактивні та функціональні вебсайти та додатки. Правильне використання цих технологій дозволяє покращити користувацький досвід та забезпечити високу якість вебпродуктів.

Серед інших важливих компонентів ІСП є вебфреймворки, бібліотеки а також хмарні платформи.

Сучасні вебфреймворки значно спрощують розробку ІСП, забезпечуючи високий рівень абстракції та готові рішення для типових задач. Вони дозволяють розробникам зосередитися на бізнес-логіці додатка, а не на рутинних завданнях. Одним із найпопулярніших вебфреймворків є Ruby (рис. 1.8).

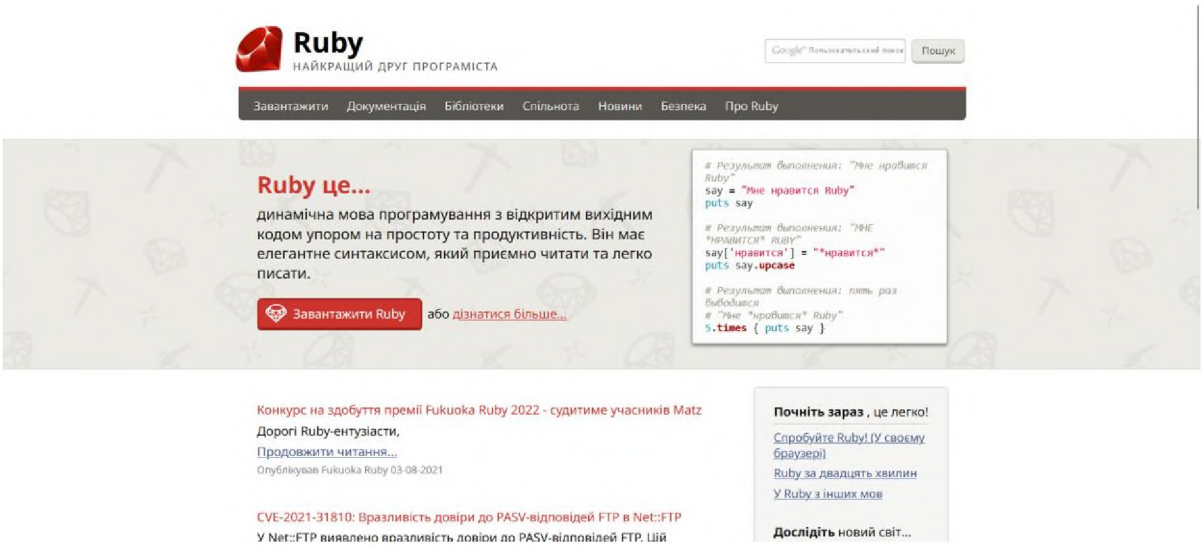

Рисунок 1.8 – Офіційний сайт вебфреймворка Ruby

Ruby це високорівнева, динамічна, об'єктно-орієнтована мова програмування, яку було розроблено в Японії наприкінці 1980-х рр. Вона відома своєю простотою синтаксису, що робить її доступною для новачків у галузі програмування. Основою Ruby є філософія "програмування має бути задоволенням", що позначилося в її доброзичливій і експресивній природі [8].

Проаналізувавши його, можна виділити декілька переваг, а саме:

1) швидка розробка завдяки конвенціям та стандартам. Ruby базується на філософії "Convention over Configuration" (Конвенція над конфігурацією), що означає, що розробники можуть швидко створювати функціональність, використовуючи вбудовані конвенції та стандарти. Це спрощує процес розробки та дозволяє швидше виводити продукт на ринок;

2) велика кількість готових бібліотек та плагінів. Екосистема Ruby має велику кількість готових бібліотек та плагінів, які дозволяють розробникам ефективно використовувати готові рішення для широкого спектру завдань, таких як аутентифікація, авторизація, робота з базою даних тощо. Це дозволяє значно прискорити процес розробки та зосередитися на вирішенні різних бізнес-задач, що необхідні для правильної та безпомилкової роботи;

3) активна спільнота розробників. Ruby має велику та активну спільноту розробників, яка активно підтримує розвиток та підтримку фреймворку. Це означає, що розробники можуть легко знаходити відповіді на свої питання, отримувати допомогу та ради від інших учасників спільноти, а також брати участь у відкритому обміні досвідом та рішеннями проблем.

Якщо виділяти його основні недоліки, то до них можна віднести, те що хоча Ruby і пропонує багато вбудованих стандартних рішень, налаштування специфічних конфігурацій може бути складним. Це особливо стосується складних або нетипових вимог до системи, які вимагають змін у стандартному поведінці фреймворку.

Це особливо актуально для проектів з нетиповими вимогами або великими командами, де потрібно встановлювати і налаштовувати різноманітні компоненти системи відповідно до конкретних потреб. В таких випадках важливо мати гнучкість і зручні інструменти для швидкого та ефективного внесення змін у конфігурацію, що може бути викликом для розробників, використовуючи Ruby.

Також, в порівнянні з деякими іншими вебфреймворками, такими як Java Spring або ASP.NET, Ruby менше поширений у корпоративному середовищі. Це може створювати деякі труднощі у впровадженні або підтримці проєктів, особливо в компаніях, де існують вже встановлені стандарти та інфраструктура для інших технологій.

Загалом, Ruby є потужним інструментом для швидкої розробки вебдодатків, забезпечуючи велику кількість вбудованих функцій та підтримку конвенцій над конфігурацією. Завдяки активній спільноті розробників та

розмаїттю готових бібліотек та плагінів, Ruby пропонує швидкий шлях до розробки функціональних та надійних вебдодатків. Однак, варто враховувати складність налаштування специфічних конфігурацій та меншу поширеність у корпоративному середовищі, що може створювати деякі виклики для проєктів, які вимагають підтримки великих компаній або інтеграції з існуючими інфраструктурами.

Важливо зазначити, що незважаючи на деякі недоліки, Ruby залишається однією з найпопулярніших мов програмування для веб-розробки.

Наступним із вебфреймворків можна розглянути фреймворк, що часто використовується для налаштування та роботи із базами даних та , на базі мови програмування Java – Spring (рис. 1.9).

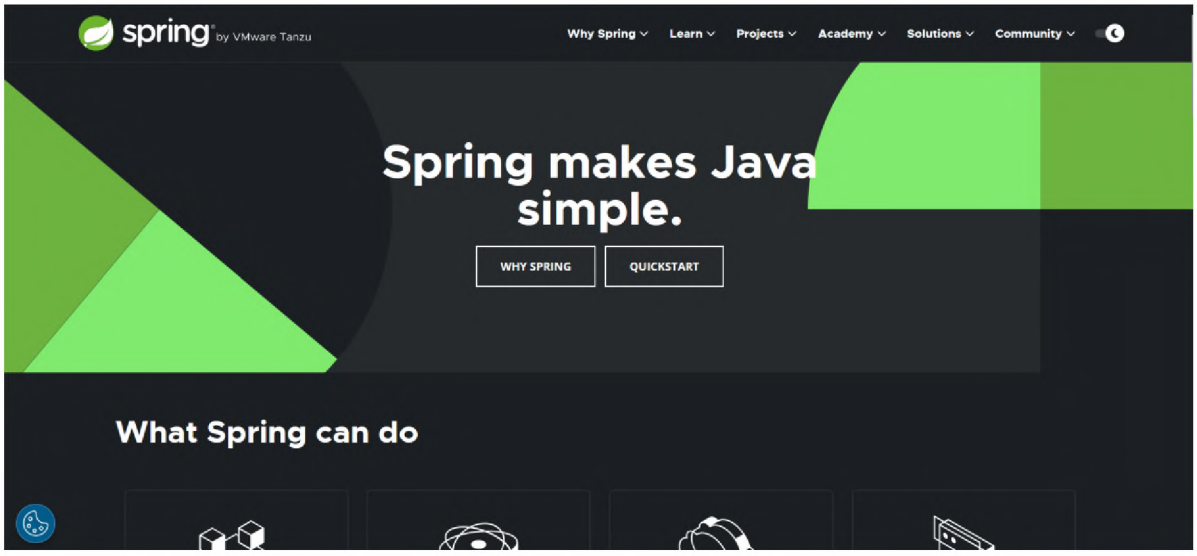

Рисунок 1.9 – Офіційний сайт вебфреймворка Spring

Spring – це фреймворк із відкритим вихідним кодом. Його написано Родом Джонсоном та випущено у 2003 р. за ліцензією Apache 2.0. Він є відомим і широко використовуваним фреймворком для розробки масштабованих та ефективних корпоративних додатків на мові програмування Java. Він включає в себе різноманітні модулі для обробки даних, забезпечення безпеки та інтеграції з іншими сервісами, що робить його вельми гнучким і потужним інструментом для розробки інформаційних сервісів підтримки [9].

Але Spring – це не один якийсь конкретний фреймворк. Це скоріше загальна назви для цілого ряду невеликих фреймворків, кожен із яких виконує якусь свою роботу. Він має модульну структуру, що дозволяє включати лише необхідні для додатка модулі і не включати ті, які не використовуватимуться. Цей підхід допоміг Spring перегнати свого конкурента на той час, EJB, оскільки програми, що використовували EJB, мали багато залежностей і були менш гнучкими та повільними [10].

Фреймворк складається з різних модулів, таких як Data Access, Web, Core Container та інші (рис. 1.10). Кожен із цих модулів надає специфічні функції, які допомагають розробникам створювати надійні та масштабовані веб-додатки.

Модуль Data Access забезпечує інтеграцію з різними базами даних, а модуль Core Container містить основні компоненти фреймворку.

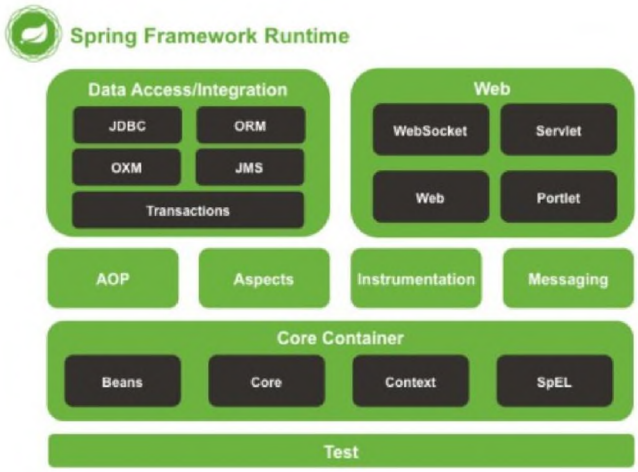

Рисунок 1.10 – Модулі вебфреймворка Spring

Завдяки модульній архітектурі, Spring дозволяє розробникам використовувати тільки ті частини фреймворку, які необхідні для конкретного проекту, що робить його надзвичайно гнучким та ефективним інструментом для розробки сучасних корпоративних додатків.

Spring має певні переваги, які виділяють його серед інших фреймворків, а саме:

1) висока продуктивність та масштабованість. Spring дозволяє створювати ефективні та високопродуктивні додатки, які можуть легко масштабуватися з ростом обсягів даних та навантаження;

2) широкий набір модулів та інтеграційних можливостей. Фреймворк має великий екосистему модулів та інтеграцій, які дозволяють розробникам легко інтегрувати різноманітні технології та сервіси у свої додатки;

3) підтримка корпоративних стандартів безпеки. Spring надає розробникам широкі можливості для реалізації безпеки в їх додатках, включаючи аутентифікацію, авторизацію та захист від різних видів атак.

До недоліків можна віднести те, що він може виявитися складним для новачків у програмуванні на Java через велику кількість концепцій та інструментів, що потрібно засвоїти.

Наступним компонентом ІСП – є фронтенд-бібліотеки, вони є ключовими інструментами у розробці сучасних вебдодатків. Вони надають розробникам широкий набір інструментів та можливостей для створення інтерактивних та ефективних інтерфейсів для користувачів.

До переваг використання фронтенд-бібліотек можна віднести:

1) швидкість розробки. Фронтенд-бібліотеки та фреймворки дозволяють розробникам швидко створювати компоненти інтерфейсу та взаємодіяти з ними, що сприяє швидкій розробці та впровадженню нових функціональних можливостей;

2) масштабованість. Багато фронтенд-інструментів надають можливості для масштабування додатків, що дозволяє їм працювати ефективно навіть у випадку збільшення обсягу даних або користувацького трафіку;

3) спрощення роботи з DOM. Фронтенд-бібліотеки та фреймворки дозволяють розробникам працювати з DOM більш ефективно, забезпечуючи спрощення взаємодії з елементами сторінки та їх оновлення;

4) спільнота та екосистема. Багато фронтенд-інструментів мають великі та активні спільноти розробників, а також розширені екосистеми бібліотек та

плагінів, що дозволяє розробникам швидко знаходити рішення на випадок проблем або використовувати готові компоненти для своїх проєктів.

Якщо виділяти основні недоліки використання фронтенд-бібліотек можна виділити такі, як:

1) великий обсяг коду. Зазвичай фронтенд-інструменти мають великий обсяг коду, що може призводити до збільшення обсягу завантаження сторінки та сповільнення її роботи, особливо на мобільних пристроях з обмеженими ресурсами;

2) навчання та оновлення. Використання нових фронтенд-бібліотек та фреймворків може вимагати часу на їх освоєння та навчання, а також постійного вивчення нових версій та оновлень;

3) залежність від зовнішніх сервісів. Деякі фронтенд-інструменти можуть вимагати підключення до зовнішніх сервісів або серверів для роботи, що може створювати додаткові точки невдачі та залежності.

Досить часто використовують бібліотеку "React.js" це бібліотека JavaScript, розроблена компанією Facebook, яка використовується для побудови інтеракивних та динамічних користувацьких інтерфейсів у вебдодатках. Вона використовується для швидкого та ефективного створення інтерактивних користувацьких інтерфейсів і вебдодатків із застосуванням значно меншої кількості коду, ніж під час використання звичайного JavaScript [11].

З кожним роком React стає дедалі надійнішим і тепер його можна використовувати для створення власних мобільних додатків з використанням React Native та настільних застосунків з використанням "Electron.js". Тож React підходить для широкого спектру вебпроєктів, від простих сайтів, до складних вебдодатків.

Ключові концепції React включають у себе компоненти, котрі є основними будівельними блоками додатка. Кожен компонент може мати власний стан і методи життєвого циклу, які визначають його поведінку та відображення. Компоненти можуть бути вкладені один в одного, створюючи

ієрархію компонентів, яка представляє собою структуру інтерфейсу додатка [12].

React також пропонує JSX (JavaScript XML) розширення JavaScript, яке дозволяє писати HTML-подібний код прямо в JavaScript файлі, що полегшує створення компонентів та розуміння структури інтерфейсу [13].

Однією з важливих переваг React є його зосередженість на односторінкових додатках (SPA), які взаємодіють з користувачем без перезавантаження сторінки. Це дозволяє створювати швидкі та реактивні додатки, які надають користувачам приємний досвід взаємодії.

Загалом, React є потужним інструментом для розробки вебдодатків, який надає зручність, ефективність та гнучкість у розробці інтерактивних інтерфейсів.

Підсумовуючи можна дійти висновку, що фреймворки та бібліотеки відіграють важливу роль у розробці вебсайтів та вебдодатків, надаючи інструменти та структури, які спрощують процес розробки, підвищують ефективність та забезпечують надійність.

1.3. Аналіз сучасних тенденцій у розробці інформаційних сервісів підтримки

Швидкий розвиток інформаційних технологій вимагає постійного вдосконалення ІСП. Сучасні тенденції у розробці таких сервісів зумовлені необхідністю підвищення ефективності, масштабованості та безпеки систем. Цей розділ присвячений аналізу основних тенденцій, які впливають на розвиток ІСП, та їхньому впливу на архітектуру та функціонал систем.

Однією з провідних тенденцій у розробці ІСП є використання хмарних технологій. Хмарні платформи, такі як Amazon Web Services (AWS), Google Cloud Platform (GCP) та Microsoft Azure, пропонують різноманітні інструменти та сервіси для розгортання, масштабування та управління ІСП.

Використання хмарних технологій дозволяє організаціям знизити витрати на інфраструктуру, забезпечити високу доступність та масштабованість сервісів, а також швидко адаптуватися до змін у вимогах бізнесу. Крім того, хмарні технології забезпечують надійний захист даних та відповідність стандартам безпеки.

Хмарні технології надають компаніям доступ до обчислювальних ресурсів та послуг через Інтернет, що дозволяє зосередитися на розробці продуктів та послуг, замість витрат на обслуговування інфраструктури.

Основні переваги використання хмарних технологій включають:

1) масштабованість в контексті хмарних технологій відноситься до можливості динамічно змінювати обсяги обчислювальних ресурсів у залежності від потреб користувача. Це означає, що компанії можуть легко збільшувати або зменшувати кількість обчислювальних потужностей в межах хмарної інфраструктури відповідно до обсягу роботи або обсягу трафіку;

2) гнучкість в контексті хмарних технологій означає можливість швидко адаптувати та змінювати інфраструктуру відповідно до потреб бізнесу або змінних умов ринку. Наприклад, якщо компанія вирішує розширити свої послуги або запустити новий продукт, вона може легко розгорнути необхідні обчислювальні ресурси та сервіси в хмарному середовищі без значних витрат часу на придбання та налаштування фізичного обладнання;

3) ефективність витрат в хмарних технологіях забезпечується моделлю плати за фактом використання, де користувачі платять лише за реальне використання обчислювальних ресурсів та послуг. Це дозволяє компаніям оптимізувати свої витрати, оскільки вони не зобов'язані інвестувати у придбання та підтримку великого обсягу обладнання, як у випадку власної інфраструктури.

Хоча хмарні технології мають багато переваг, вони також мають свої недоліки, які варто враховувати:

1) залежність від Інтернет-з'єднання. Використання хмарних послуг передбачає постійне підключення до Інтернету. Якщо з'єднання буде

недоступним або недостатньо стабільним, це може призвести до перебоїв у доступі до даних та сервісів;

2) проблеми з безпекою даних. Збереження конфіденційної інформації у хмарних сервісах може викликати стурбованість щодо безпеки даних. Хоча провайдери хмарних послуг зазвичай мають рівень безпеки вище, ніж більшість організацій можуть собі дозволити, існує ризик доступу до даних третіми сторонами через порушення безпеки з боку провайдера;

3) проблеми з приватністю даних. Використання хмарних послуг може вимагати передачі конфіденційних даних третім сторонам, що може викликати занепокоєння з приводу приватності;

4) загрози для безпеки мережі. Хмарні технології можуть стати об'єктом атак з боку зловмисників, які намагаються отримати доступ до даних або витратити ресурси. Спільна інфраструктура в хмарних сервісах може зробити їх вразливими до таких атак;

5) можливість відмови сервісу. Існує ризик того, що провайдер хмарних послуг може припинити надання послуг або змінити їхні умови, що може призвести до проблем для користувачів, які покладаються на ці послуги.

Ці недоліки важливо уважно розглядати і враховувати під час прийняття рішення щодо використання хмарних технологій для інформаційного сервісу.

Аналіз вартості та масштабованості різних хмарних рішень є критичним етапом у виборі оптимальної інфраструктури для проєкту. Для цього проводиться докладне порівняння різних хмарних платформ з урахуванням таких факторів, як: вартість, масштабованість, продуктивність та надійність.

До прикладу можна навести одну із провідних платформ Microsoft Azure (рис. 1.11).

Azure надає широкі можливості для масштабування, включаючи вертикальне та горизонтальне масштабування за допомогою сервісу Azure Virtual Machines та Azure Kubernetes Service.

Azure пропонує гнучку модель ціноутворення, яка може бути особливо вигідною для підприємств, які вже використовують інші продукти Microsoft.

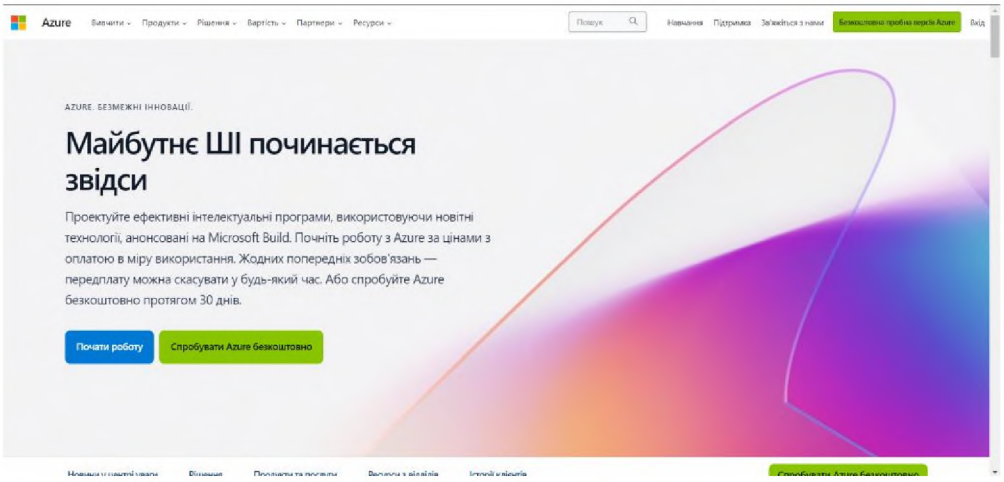

Рисунок 1.11 – Офіційний сайт хмарного сервісу Microsoft Azure

У висновку варто підкреслити, що хмарні технології та сервіси є важливим інструментом для розробки та впровадження інформаційних сервісів підтримки продажу комп'ютерної техніки. Розглянуті популярні платформи хмарних обчислень, такі як AWS, Microsoft Azure, Google Cloud Platform та IBM Cloud, надають широкий спектр можливостей для впровадження проєктів різної складності та масштабів.

Однак, важливо враховувати як переваги, так і недоліки використання хмарних сервісів. З одного боку, вони забезпечують гнучкість, масштабованість та ефективність витрат, що робить їх привабливими для бізнесу. З іншого боку, можуть виникати проблеми з безпекою даних, залежність від інфраструктури постачальника та можливість виникнення додаткових витрат.

Мікросервісна архітектура є ще однією важливою тенденцією у розробці ІСП. Вона передбачає розбиття системи на невеликі, незалежні сервіси, кожен з яких виконує окрему функцію та взаємодіє з іншими сервісами через стандартизовані API. Це дозволяє підвищити гнучкість системи, полегшує процес оновлення та масштабування, а також покращує надійність, оскільки відмова одного сервісу не впливає на роботу всієї системи. Проте мікросервісна архітектура потребує складного управління та моніторингу, а також наявності надійної мережевої інфраструктури.

Важливу роль у сучасній розробці ІСП відіграють методології DevOps та CI/CD (Continuous Integration/Continuous Deployment). Вони передбачають інтеграцію процесів розробки та операційного управління, що дозволяє прискорити випуск нових версій програмного забезпечення, покращити якість коду та забезпечити швидке реагування на зміни у вимогах користувачів. DevOps методологія включає автоматизацію процесів розгортання, тестування та моніторингу, що знижує ризики помилок та підвищує ефективність розробки.

Іншою значущою тенденцією є впровадження інтелектуальних систем. Використання штучного інтелекту (ШІ) та машинного навчання (МН) у ІСП дозволяє покращити обробку даних, забезпечити аналітику в реальному часі, автоматизувати процеси прийняття рішень та надавати користувачам більш точні та релевантні рекомендації. Інтелектуальні системи можуть також допомагати в обробці запитів користувачів, забезпечуючи швидку та точну підтримку.

Кібербезпека є критично важливим аспектом у розробці сучасних ІСП. Збільшення кількості кіберзагроз та вимог до захисту даних змушує організації впроваджувати передові технології та практики для забезпечення безпеки своїх систем. Це включає використання шифрування, багатофакторної автентифікації, виявлення та реагування на загрози в реальному часі, а також відповідність вимогам регуляторів та стандартів безпеки.

Аналіз сучасних тенденцій у розробці інформаційних сервісів підтримки показує, що головними пріоритетами є використання хмарних технологій, мікросервісна архітектура, впровадження методологій DevOps та CI/CD, інтеграція інтелектуальних систем та забезпечення кібербезпеки. Ці тенденції спрямовані на підвищення ефективності, гнучкості, масштабованості та безпеки ІСП, що дозволяє організаціям швидко адаптуватися до змін у вимогах ринку та забезпечувати високий рівень підтримки своїх користувачів.

РОЗДІЛ 2
ВИВЧЕННЯ РИНКУ ТА ВИЗНАЧЕННЯ ТЕХНІЧНИХ ВИМОГ ІНФОРМАЦІЙНИХ СЕРВІСІВ ПІДТРИМКИ ПРОДАЖУ

2.1. Аналіз наявних інформаційних сервісів підтримки продажів комп'ютерної техніки.

В сучасних умовах розвитку інформаційних технологій та цифрової економіки, ефективна підтримка продажів комп'ютерної техніки є ключовим фактором успіху для багатьох компаній. Інформаційні сервіси, призначені для підтримки процесу продажів, забезпечують зручність для користувачів, оптимізують взаємодію з клієнтами та сприяють збільшенню прибутків. Розуміння існуючих рішень на ринку, їхніх сильних і слабких сторін, а також вивчення найкращих практик, є необхідним кроком для розробки ефективного і конкурентоспроможного сервісу.

Першим об'єктом для огляду є сайт "Telemart.ua" який спеціалізується на продажу комп'ютерної техніки та комплектуючих. Цей ресурс пропонує широкий асортимент товарів, включаючи настільні комп'ютери, ноутбуки, комплектуючі, периферійні пристрої та інші електронні пристрої [14].

Сайт має зручну систему навігації, яка дозволяє швидко знаходити потрібні товари за категоріями, брендами та специфікаціями. Доступний розширений пошук із фільтрами за різними параметрами. Кожен товар супроводжується детальним описом, технічними характеристиками, фотографіями високої якості та відгуками користувачів, що дозволяє потенційним покупцям зробити обґрунтований вибір при виборі комп'ютерної техніки (рис. 2.1).

Початок аналізу будь якого сайту починається із аналізу його дизайну, загального вигляду та функціональності. Інтерфейс сайту зрозумілий навіть для недосвідчених користувачів, з чітко структурованою інформацією та логічним розміщенням елементів.

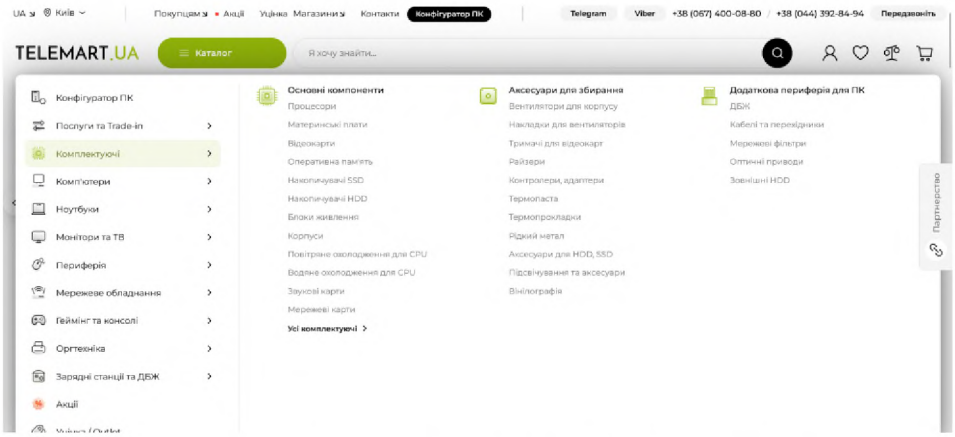

Рисунок 2.1 – Навігація на сайті Telemart.ua

Сайт має адаптивний дизайн, що забезпечує комфортне використання на мобільних пристроях. Користувачі можуть легко зв'язатися з представниками компанії через онлайн-чат, телефон або електронну пошту, отримуючи швидкі та професійні відповіді на свої запитання (додаток А).

Також, сайт підтримує різні методи оплати, включаючи банківські карти та електронні гаманці, а також пропонує декілька варіантів доставки товарів, що забезпечує зручність для клієнтів з різних регіонів. Впроваджена система бонусів та знижок для постійних клієнтів стимулює повторні покупки.

Однією з ключових зручностей сайту є кошик товарів. Інтерфейс кошика розроблений таким чином, щоб спростити процес оформлення замовлення. Користувачі можуть легко додавати або видаляти товари з кошика, регулювати кількість одиниць кожного товару, а також одразу бачити загальну суму покупки (рис. 2.2).

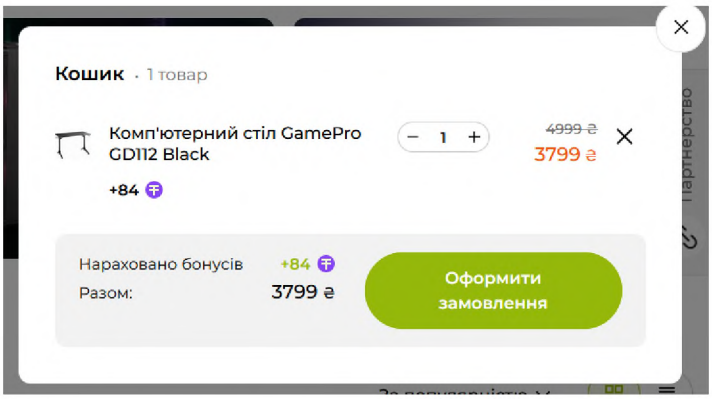

Рисунок 2.2 – Кошик для товарів

Кошик має чіткий та інтуїтивно зрозумілий дизайн. Наприклад, як видно на скріншоті, відображається загальна вартість товарів, кількість нарахованих бонусів, а також кнопка для оформлення замовлення, яка виділяється на тлі іншої інформації, що спонукає до завершення покупки. Крім того, доступна інформація про знижки та акції, що дозволяє користувачам бачити реальну економію.

Великий вибір товарів дозволяє задовольнити потреби різних категорій покупців, від домашніх користувачів до професіоналів. Високий рівень клієнтської підтримки та детальна інформація про товари підвищують задоволеність клієнтів та сприяють лояльності до бренду.

Наступним сайтом, який підлягає детальному аналізу, є сайт – "ItBox.ua".

Він має всі ті ж плюси, що і у першого сайту але є те чого там не було. Першим важливим елементом на цьому сайті, є – підбірка із популярними товарами (рис. 2.3). Це може допомогти певним групам користувачів, обрати товар, можливо котрий вони і не хотіли спочатку, але який їм тією чи іншою мірою, був потрібен.

Рисунок 2.3 – Підбірка популярних товарів

Також на сайті на "ItBox.ua", на відміну від "Telemart.ua", є блок із популярними брендами. Це повинно допомогти, як і людям які не гарно розбираються у техніці, так і людям котрі шукають техніку конкретного бренду (рис. 2.4).

Рисунок 2.4 – Популярні бренди

Проаналізувавши сайт "Telemart.ua" та "ItBox.ua", можна зробити висновок, що вони є ефективними інструментом підтримки продажів комп'ютерної техніки, які забезпечують зручність для користувачів та оптимізують процес вибору і покупки товарів.

2.2. Визначення цільової аудиторії та їх потреб

Визначення цільової аудиторії є критичним етапом у розробці будь-якого інформаційного сервісу, зокрема сервісу підтримки продажу комп'ютерної техніки. Чітке розуміння, хто саме буде користуватися сервісом, дозволяє розробникам створювати продукти, які відповідають очікуванням та потребам користувачів, що сприяє успіху проєкту (див. додаток А).

Щоб ефективно визначити свою цільову аудиторію, в першу чергу необхідно звернути увагу на її демографічні характеристики:

1) основна частина суб'єктів інформаційних послуг з продажу комп'ютерної техніки-це люди у віці від 18 до 45 рр. Молоді люди часто є основними споживачами нових технологій, а люди середнього віку мають фінансові кошти для покупки більш дорогих товарів;

2) комп'ютерні технології цікаві представникам обох статей, але чоловіки зазвичай складають основну частину аудиторії;

3) цільова аудиторія має середній або високий рівень доходу, що дозволяє їй купувати комп'ютерну техніку та аксесуари.

Також важливо враховувати поведінкові характеристики користувача:

1) більшість користувачів мають високий рівень технічної обізнаності, розуміють основні технічні характеристики та використовують Інтернет для пошуку та порівняння товарів;

2) цільова аудиторія часто робить покупки онлайн в пошуках вигідних пропозицій, рекламних акцій і знижок. Вони цінують зручність і швидкість онлайн-покупок;

3) користувачам потрібна детальна інформація про товар, включаючи технічні характеристики, огляди та порівняння, а також можливість отримати консультацію та підтримку.

Психологічні характеристики допомагають краще зрозуміти мотиви та потреби цільової аудиторії:

1) користувачі цікавляться новітніми технологіями, іграми, програмуванням, графічним дизайном і т. д.;

2) важливими цінностями є інновації, продуктивність, якість та надійність;

3) користувач купує комп'ютерне обладнання для роботи, досліджень, розваг та особистого розвитку.

Визначення цільової аудиторії та її потреб є важливим кроком у розвитку інформаційних сервісів, що підтримують продаж комп'ютерної техніки. Розуміючи демографічні, поведінкові та психологічні особливості користувачів, ми можемо створювати сервіси, які відповідають очікуванням і забезпечують зручність, інформаційну підтримку і задоволення від покупок. Такий підхід допомагає підвищити лояльність користувачів і загальний успіх проєкту.

2.3. Вибір технологій та платформ для розробки

У сучасному бізнес-середовищі, де ефективність і швидкість реагування на зміни є ключовими факторами успіху, використання простих, але потужних інструментів для розробки інформаційних сервісів є важливим аспектом. У цьому розділі ми розглянемо вибір засобів розробки для інформаційного сервісу підтримки продажів комп'ютерної техніки, зокрема обмежимося використанням лише HTML, CSS та JavaScript. Ці технології, хоча і є базовими, володіють достатньою гнучкістю та функціональністю для створення ефективних і привабливих вебдодатків.

HTML, CSS та JavaScript є основними технологіями, що лежать в основі будь-якого вебдодатку. Їх використання забезпечує низку важливих переваг:

1) HTML є основою будь-якої вебсторінки, забезпечуючи структуру і контент. CSS дозволяє оформити цю структуру, надаючи їй стиль і привабливий зовнішній вигляд. JavaScript, у свою чергу, додає інтерактивність та динамічну поведінку вебсторінкам. Ці технології є широко доступними, легкими у вивченні та використанні;

2) використовуючи HTML, CSS та JavaScript, можна створювати адаптивні дизайни, які виглядають добре на різних пристроях і розмірах екранів. Це особливо важливо для інформаційних сервісів, які повинні бути доступними як на настільних комп'ютерах, так і на мобільних пристроях;

3) сучасний JavaScript разом з HTML5 і CSS3 дозволяють реалізувати широкий спектр функціональних можливостей, таких як інтерактивні елементи, анімації, обробка подій, маніпуляція DOM та багато іншого. Це дозволяє створювати повноцінні вебдодатки без необхідності звертатися до більш складних фреймворків або бібліотек;

4) всі сучасні браузери підтримують HTML, CSS та JavaScript, що забезпечує сумісність вебдодатків на різних платформах без необхідності додаткових налаштувань або адаптацій;

5) використання базових технологій для розробки вебдодатків знижує вимоги до серверних ресурсів, оскільки більшість обробки даних виконується на стороні клієнта. Це може призвести до значної економії на хостингу та інших інфраструктурних витратах.

Для розробки інформаційного сервісу підтримки продажів комп'ютерної техніки, використання лише HTML, CSS та JavaScript дозволяє створити функціональний і привабливий інтерфейс, що покриває всі основні потреби користувачів. Наприклад, HTML забезпечує структуру сторінок, CSS відповідає за їхній зовнішній вигляд, а JavaScript додає необхідну інтерактивність.

Для написання та редагування коду важливо використовувати зручний та функціональний текстовий редактор або інтегроване середовище розробки (IDE). Один з найбільш популярних та потужних інструментів на сьогодні є Visual Studio Code (VSCode), який створений компанією Microsoft.

VSCode відзначається своєю легкою вагою та швидкодією, що робить його ідеальним для роботи на різних платформах, включаючи Windows, macOS та Linux. Він запускається швидко, не сповільнює систему і не займає багато ресурсів, що є критично важливим для продуктивної роботи розробників (рис. 2.5).

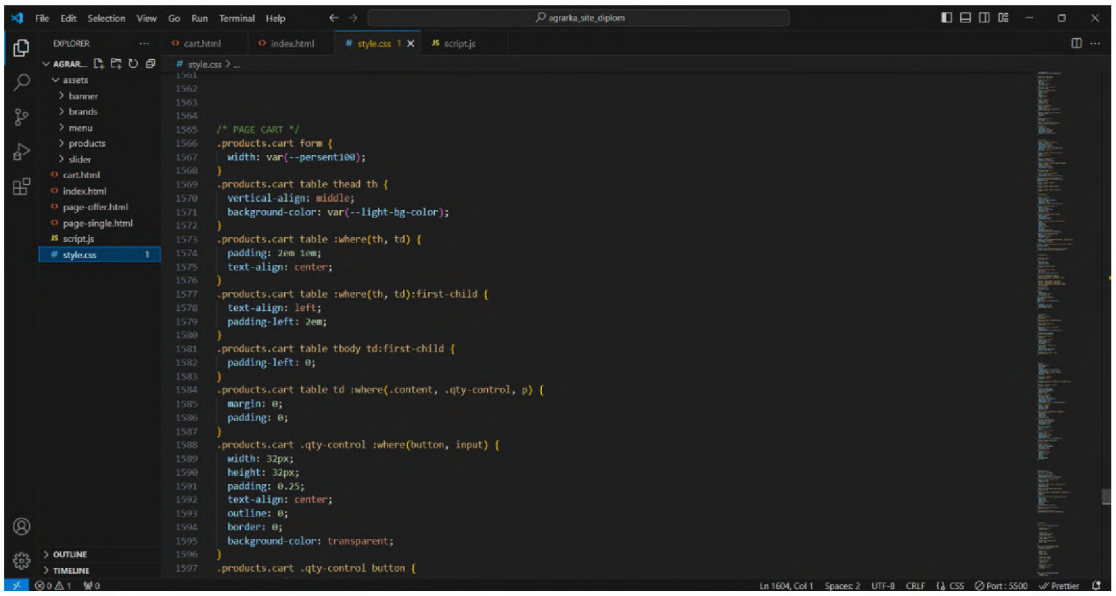

Рисунок 2.5 – Інтерфейс Visual Studio Code

Однією з найбільших переваг VSCode є його розширюваність. Користувачі можуть встановлювати різноманітні розширення та плагіни з Visual Studio Code Marketplace, які додають нові функції, підтримку різних мов програмування, інтеграцію з системами контролю версій, такими як Git, та багато іншого. Популярні розширення включають підтримку Python, JavaScript, CSS, і навіть інструменти для розробки вебдодатків на основі фреймворків Angular та React

VSCode є одним з найпопулярніших редакторів коду завдяки своїй швидкодії, розширюваності, та зручності у використанні. Він забезпечує

потужні інструменти для написання, редагування та відлагодження коду, що робить його незамінним інструментом для розробників усіх рівнів. Завдяки можливості налаштування та широкому вибору розширень, VSCode підходить для різних типів проєктів та технологій, що дозволяє створювати якісні та ефективні інформаційні сервіси підтримки продажу комп'ютерної техніки [15].

Загалом, для багатьох розробників VSCode став не просто інструментом, а справжнім партнером у вирішенні складних завдань. Розширюваність цього редактора дозволяє адаптувати його під конкретні потреби та вподобання кожного користувача. Незалежно від того, чи ви працюєте над великим проєктом або просто вивчаєте нову мову програмування, ви знайдете у VSCode потрібні інструменти та розширення, які полегшать вашу роботу. Більше того, активна спільнота розробників постійно внесе нові ідеї та функції, роблячи редактор ще потужнішим та зручнішим у використанні.

РОЗДІЛ 3
РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОГО СЕРВІСУ ПІДТРИМКИ ПРОДАЖУ КОМП'ЮТЕРНОЇ ТЕХНІКИ

3.1. Структура інформаційного сервісу підтримку продажу

Інформаційний сервіс підтримки продажу складається з кількох ключових елементів, кожен з яких відіграє важливу роль у забезпеченні зручності та функціональності для користувачів [16].

Header (шапка) знаходиться у верхній частині сторінки і зазвичай включає логотип компанії, основне меню навігації, контактну інформацію та, можливо, кнопку для входу чи реєстрації. Цей елемент дозволяє користувачам швидко орієнтуватися на сайті та знаходити потрібні розділи.

Main content area (основний вміст) займає центральну частину сторінки. Тут розміщуються основні інформаційні блоки: оголошення, новини, пропозиції, описи продуктів чи послуг, а також відгуки клієнтів. Цей розділ може бути розбитий на секції з картинками, текстами, відео чи інтерактивними елементами, що робить інформацію доступною та зрозумілою.

Sidebar (бічна панель) може бути розміщена зліва або справа від основного вмісту. Вона часто містить додаткові елементи, такі як посилання на популярні продукти, категорії товарів, рекламні блоки, новини або контактну інформацію. Бічна панель допомагає користувачам легко знаходити додаткову інформацію, не покидаючи основної сторінки.

Footer (нижній колонтитул) знаходиться внизу сторінки. Тут зазвичай розміщуються копірайти, посилання на політику конфіденційності, умови користування, контактна інформація, соціальні мережі та інші корисні посилання. Footer також може включати швидкі посилання на важливі розділи сайту, такі як FAQ або підтримка клієнтів.

Navigation bar (навігаційна панель) може бути частиною header або окремим елементом, що залишається видимим при прокручуванні сторінки.

Вона містить посилання на основні розділи сайту, дозволяючи користувачам швидко переміщатися між ними.

Search bar (пошуковий рядок) може бути розміщений у "header" або в іншому зручному місці. Пошук дозволяє користувачам швидко знаходити потрібну інформацію або продукти, вводячи ключові слова.

Shopping cart (корзина для товару) є невідє'мною частиною будь-якого сайту електронної комерції. Корзина зазвичай представлення у вигляді іконки, часто розміщеною у верхньому правому куті "header", яка показує кількість доданих товарів. При натисканні на іконку відкривається панель або окрема сторінка, де користувач може переглянути додані товари, змінювати їх кількість або видаляти товари. Корзина також містить кнопку для переходу до оформлення замовлення (checkout). Це забезпечує зручний спосіб управлінням покупками, дозволяючи користувачам бачити загальну вартість замовлення і швидко оформляти покупку [17].

Крім цих основних елементів, сайт може мати додаткові компоненти, такі як спливаючи вікна для спеціальних пропозицій, інтерактивні форми зворотного зв'язку, карти місцезнаходження та інші елементи, що покращують взаємодію з користувачем.

3.2. Розробка дизайну інформаційного сервісу підтримки продажу

Маючи функціональні вимоги до інформаційних сервісів для підтримки продажу комп'ютерної техніки і розуміючи кількість сторінок і призначення кожного з них, в більшості випадків, приступаючи до розробки самого сервісу, необхідно виконати ще один важливий крок. Перед розробкою сервісу, який також вимагає обговорення і схвалення з боку клієнтів - створенні макета сайту.

Для цього потрібно зрозуміти як працює UI/UX дизайн. Розуміння того, як працює UI/UX дизайн, відкриває двері до створення інформаційних

сервісів, що не лише відповідають потребам клієнтів, а й надають їм приємний та ефективний користувацький досвід.

UI (інтерфейс користувача) орієнтований на зовнішній вигляд та взаємодію з елементами інтерфейсу, такими як кнопки, поля введення, та інші компоненти, що користувач може бачити та взаємодіяти з ними. Дизайн UI включає в себе аспекти, такі як кольори, шрифти, розміщення елементів на сторінці та їх візуальну привабливість [18, 19].

UX (взаємодія користувача) більш зосереджений на взаємодії користувача з продуктом. Він вивчає, як користувачі використовують продукт, їхні очікування та реакції на різні елементи інтерфейсу. UX дизайнер старається забезпечити, щоб користувачі мали зручний та приємний досвід взаємодії з продуктом, навіть якщо вони зустрічаються з ним вперше [20, 21].

Розуміння цих принципів дозволяє розробникам створювати інформаційні сервіси, що не лише відповідають потребам користувачів, а й надають їм задоволення та комфорт взаємодії з продуктом.

Навіть якщо взяти до уваги функціональні вимоги і повністю зрозумієте, що повинен робити інформаційний сервіс, дуже ймовірно, що розробник створить абсолютно правильний сервіс, який буде робити все, що хотів замовник, але при цьому його зовнішній вигляд не буде відповідати очікуванням замовника. Тому, враховуючи, що в більшості інформаційних сервісів створення користувальницького інтерфейсу при розробці екрану може займати більше 3/2 всього часу, варто заздалегідь подбати про те, щоб зовнішній вигляд розроблюваного продукту відповідав задумом замовника.

Список програмного забезпечення, яке можна використовувати для розробки інтерфейсу користувача, дуже широкий, і воно також може включати графічні редактори, такі як Paint, але на практиці використовуються більш вузькоспеціалізовані дизайнерські послуги. Серед них варто виділити Skrech, Figma та Adobe XD.

Для створення дизайну була обрана Figma, оскільки вона є кросплатформною, працює в браузері, а безкоштовна версія має мінімальні

обмеження. Figma також дозволяє створювати інтерактивні прототипи, які можна використовувати для подальших презентацій. Компоненти (кнопки, форми) можуть бути підготовлені заздалегідь, і якщо клієнт змінить стиль кнопки або іншого елемента, він буде оновлений у всіх макетах шляхом зміни стилю компонента. Це також полегшує внесення змін до дизайну у разі частої зміни клієнта при розробці інформаційних сервісів (рис. 3.1) [22].

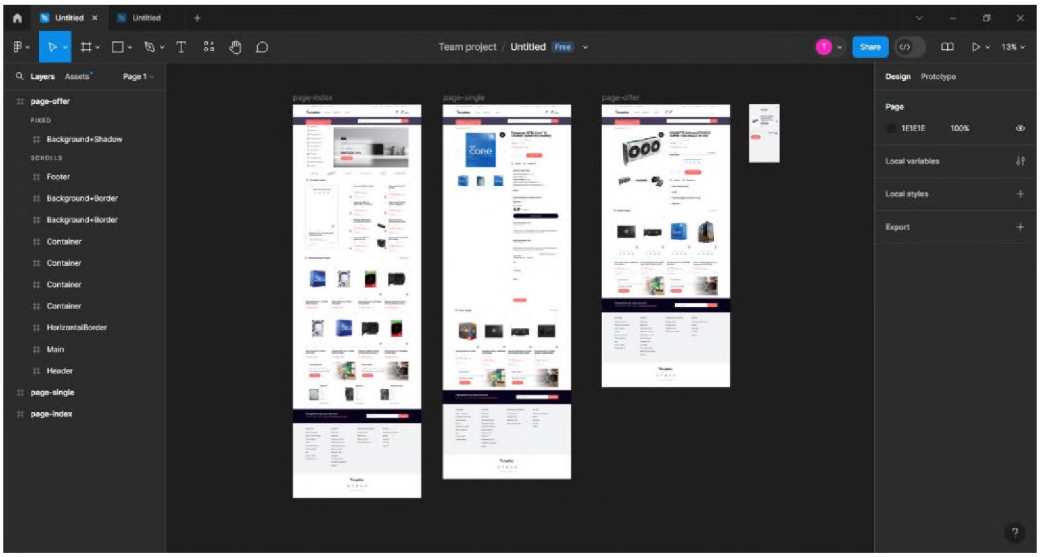

Рисунок 3.1 – Інтерфейс застосунку Figma

Для кожної сторінки було спроєктовано відповідний макет (додаток Б).

3.3. Розробка інформаційного сервісу підтримки продажу

Для початку роботи потрібно обрати місце но жорсткому диску де буде зберігатися інформаційний сервіс. Після обрання відповідного місця зберігання потрібно створити файли котрі буду в собі зберігати інформацію про сайт. Для початку потрібно створити декілька файлів: файл із розширенням ".html" – зберігає структуру сайту, ".css" – стилізацію окремих елементів на сторінці та ".js" – скрипти які потрібні для підтримки інтерактивності із користувачем.

Перший файл буде називатися "index.html" – це головна сторінка сайту, наступним створений файлом буде "style.css", котрий буде зберігати стилі для всіх сторінок сайту і останнім буде створено файл "script.js", відповідно – зі скриптами (рис. 3.2).

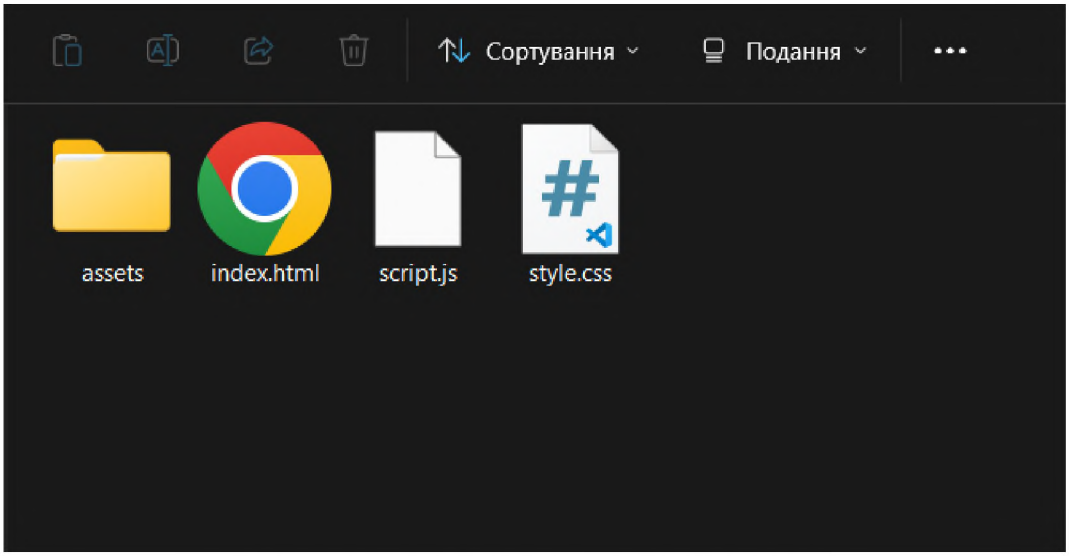

Рисунок 3.2 – Початкові файли ІСП

Також була створенна папка "Assets" для зберігання зображень, що використовуються на сайті.

Наступним кроком буде задання певних значень за допомогою селектора root. Це потрібно для того, щоб часто використані значення, наприклад – колір, не назначати кожного разу. Тепер його можна назначити за допомогою "var()" [23].

```
:root {
 --primary-color: #ff6b6b;
 --secondary-color: #794afa;
 --secondary-dark-color: #453c5c;

 --white-color: #ffffff;
 --light-bg-color: #f2f3f5;
 --light-text-color: #7c899a;
 --border-color: #e5e8ec;
 --dark-color: #0a021c;

 --font-small: 13px;
 --font-smaller: 11px;
```

Фрагмент стилю вище містить опис для декількох елементів. Вони будуть викликатися для різних властивостей, наприклад для стилізації тексту,

для font-size можна задати значення "var(--font-small: 13px)", це змінить розмір тексту. Цей фрагмент коду легко змінювати та додавати нові елементи, що є досить зручним інструментом у роботі із кодом.

Для використання іконок на сайті, можна підключити сторонню бібліотеку із ними, що дозволить звертатися до них за допомогою команд.

```
<link
 href="https://cdn.jsdelivr.net/npm/remixicon@4.2.0/fonts/remixicon.css"
rel="stylesheet"/>
<link
```

Цей фрагмент HTML-коду підключає зовнішній CSS-файл до вебсторінки. Тег "link" використовується для імпорту ресурсів, таких як файли стилів. Атрибут "href" вказує на URL-адресу зовнішнього CSS-файлу, який містить шрифти іконок Remix icon версії 4.2.0, розміщений на CDN "jsDelivr". Атрибут "rel" визначає відношення між поточною вебсторінкою і підключеним ресурсом, в даному випадку це "stylesheet", що означає підключення файлу стилів.

Наступним етапом у створенні сайту є створення каркасу із семантичних тегів, котрі будуть відповідати за певні ділянки сторінки. Наприклад тег "header" відповідає за верхню частину сайту, де розміщується логотип та кнопки для навігації.

Далі потрібно задати стилі для всієї сторінки.

```
body {
 font-family: "Rubik", sans-serif;
 font-size: 16px;
 font-weight: 400;
 line-height: 1.4;
 color: var(--dark-color);
 background-color: var(--white-color);
}
```

Цей фрагмент коду задає стилі для тега "body" на вебсторінці. Він визначає, що основний шрифт тексту буде "Rubik", а у випадку, якщо цей шрифт недоступний, використовується стандартний шрифт без зарубок "sans-serif". Розмір шрифту встановлюється на 16 пікселів, товщина шрифту на 400

(звичайний текст), а міжрядковий інтервал "line-height" дорівнює 1.4, що забезпечує зручне читання тексту. Колір тексту встановлюється за допомогою CSS-змінної "--dark-color", а колір фону задається змінною "--white-color". Це допомагає зберегти узгодженість стилів на всій вебсторінці. Таким чином будуть задаватися стилі для всіх елементів на сторінці.

Наступним етапом у розробці буде – створення випадаючого списку, котрий представлено на рисунку 3.3, це знадобиться для надання великої кількості інформації, яка не буде відображатися до виконання певних дій користувачем, наприклад на ведення курсора на певний об'єкт на сторінці. Наведений нижче фрагмент коду створює елемент списку з класом "has-child", який містить посилання на розділ "Популярне". Поруч із цим посиланням є іконка зі стрілкою вниз, що вказує на наявність підменю. Коли це підменю відкривається, воно показує великий контейнер з додатковою інформацією.

У контейнері є блок "wrapper", всередині якого є ще один блок "flexcol". Цей блок містить рядок "row", що включає заголовок "h4" з текстом "Процесори" та список "ul" з елементами "li". Кожен елемент списку є посиланням на конкретну модель процесора, наприклад, "Intel Core i9-13900K" або "AMD Ryzen 5 7600X". Наприкінці списку є посилання з класом "view-all", яке веде до сторінки з усіма пропозиціями, і поруч з цим посиланням є іконка зі стрілкою вправо.

```
<li class="has-child">
 <a href="#">Популярне
    <div class="icon-small"><i class="ri-arrow-down-s-line"></i></div>
   </a>
  <div class="mega">
   <container>
    <div class="wrapper">
    <div class="flexcol">
    <div class="row">
     <h4>Процесори</h4>
     <ul>
      <li><a href="">Intel Core i9-13900K</a></li>
      <li><a href="">Intel Core i5-13600K</a></li>
      <li><a href="">AMD Ryzen 5 7600X</a></li>
```

```
            <li><a href="">Intel Core i9-11900K</a></li>
            <li><a href="">AMD Ryzen 7 5800X3D</a></li>
            <li><a href="">AMD Ryzen 5 5600X</a></li>
            <li><a href="">Intel Core i5-12600K</a></li>
           </ul>
           <a href="#" class="view-all">Усі пропозиції <i class="ri-arrow-right-
line"></i></a>
            </div>
           </div>
```

Для того, щоб даний список з'являвся тільки при наведені курсора було використання змінення параметру "display" та за застосування псевдо тегу "hover" (рис. 3.3).

```
nav .mega {
 display: none;
}
nav li.has-child:hover .mega {
 display: block;
}
```

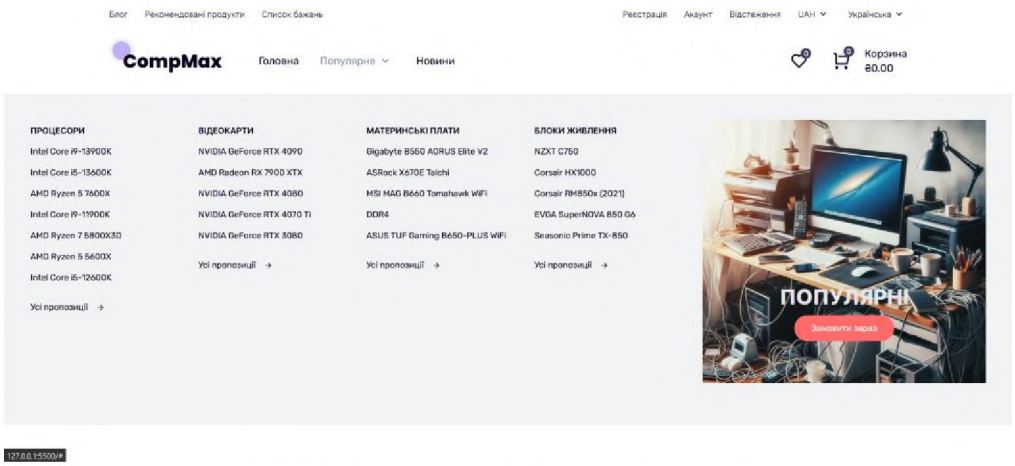

Рисунок 3.3 – Випадаючий список

Таким же чином створювався і інший випадаючий список в блоці "header-main mobile-hide" за допомогою тегу: "has-chaild", для якого була прорoблена певна стилізація та налаштування у файлі "style.css". Для розміщення бокового меню було використано семантичний тег: "aside".

У файлі "style.css " для класу "has-child" була проведена певна стилізація та налаштування. Стилізація включає в себе визначення зовнішнього вигляду та поведінки випадаючого списку, наприклад, кольору фону, шрифтів,

відступів, відображення та приховування елементів при взаємодії користувача.

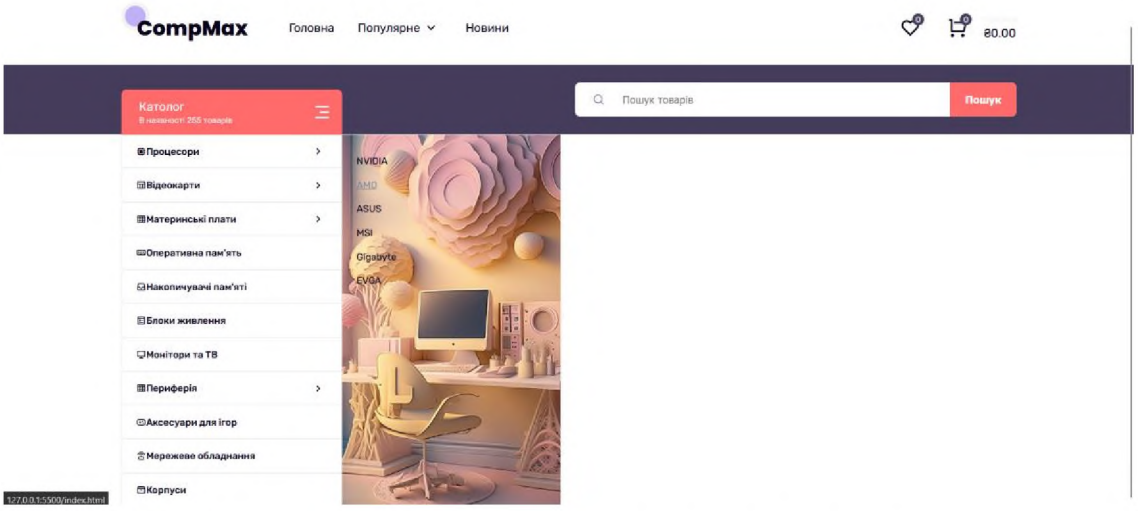

Рисунок 3.4 – Боковий випадаючий список

Для оптимізації цього списку під телефонні девайси було створено скрипт, що приставлений нижче.

```
function copyMenu() {
  //copy inside .dpt-cat to .departments
  var dptCategory = document.querySelector('.dpt-cat')
  var dptPlace = document.querySelector('.departments')
  dptPlace.innerHTML = dptCategory.innerHTML;
} copyMenu();
```

Цей скрипт – оптимізує відображення списку для мобільних пристроїв, копіюючи вміст одного HTML-елемента в інший. Спочатку створюється функція "copyMenu()", яка відповідає за копіювання. За допомогою методу "document.querySelector()", вибираються два елементи з класами ".dpt-cat" і ".departments". Вміст першого елемента, який знаходиться у ".dpt-cat", копіюється в другий елемент з класом ".departments" шляхом встановлення його "innerHTML" таким самим, як у вихідного елемента. Нарешті, функція "copyMenu()" викликається, щоб виконати копіювання після завантаження сторінки або наявності всіх необхідних елементів у DOM. Таким чином, скрипт забезпечує, що вміст списку відображається оптимізовано для

користувачів мобільних пристроїв (рис. 3.5). Це дозволить обслуговувати більшу кількість користувачів, у котрих не маж персонального комп'ютера.

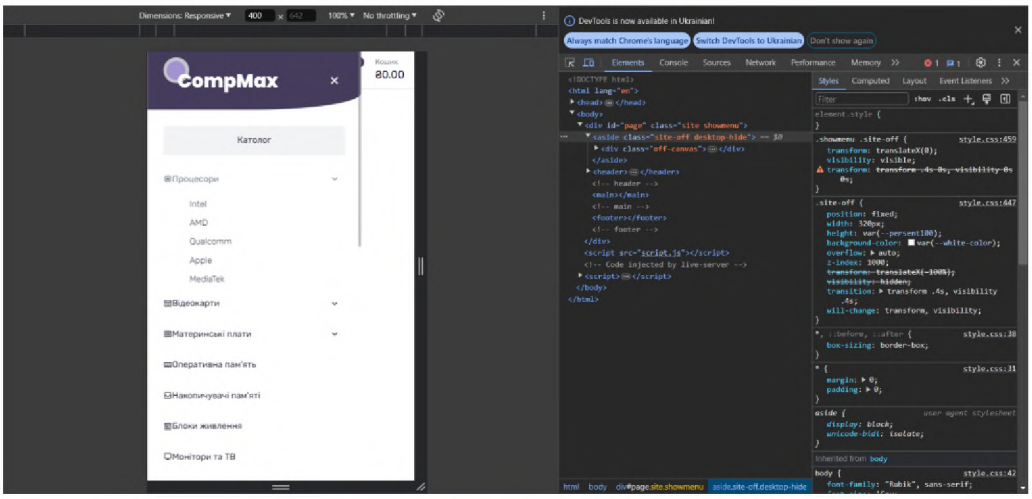

Рисунок 3.5 – Оптимізація під мобільні телефони

Цей фрагмент коду визначає стилі для бокового меню на вебсторінці, яке можна приховувати та показувати.

```
.site-off {
 position: fixed;
 width: 320px;
 height: var(--persent100);
 background-color: var(--white-color);
 overflow: auto;
 z-index: 1000;
 transform: translateX(-100%);
 visibility: hidden;
 transition: transform .4s, visibility .4s;
 will-change: transform, visibility;
}
.showmenu .site-off {
 transform: translateX(0);
 visibility: visible;
 transform: transform .4s 0s, visibility 0s 0s;
}
```

Серед них можна виділити основні:

1) Властивість "transform: translateX(-100%)"; – початкове положення меню поза екраном зліва;

2) Властивість "visibility: hidden"; – меню приховане;

3) Властивість "transition: transform.4s, visibility.4s"; – плавний перехід для властивостей transform і visibility тривалістю 0.4 секунди;

4) Властивість "will-change: transform, visibility"; – підказка браузеру, які властивості змінюватимуться, для оптимізації рендерингу.

Наступним важливим етапом у розробці інформаційного сервісу підтримки, є створення корзини для товарів (рис. 3.6).

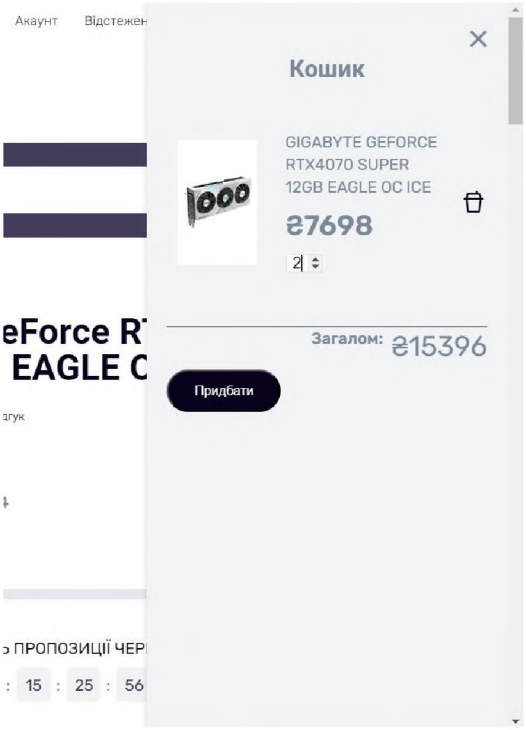

Рисунок 3.6 – Зовнішній вигляд корзини товарів

Для додання інтерактивності до елементів корзини покупок на вебсторінці потрібно створити спеціальну функцію під назвою "ready()". Функція виконується після завантаження сторінки або при певній події, наприклад, після того як DOM повністю завантажений.

```
function ready(){
  var removeCartButtons = document.getElementsByClassName('cart-remove');
  console.log(removeCartButtons)
  for (var i = 0; i < removeCartButtons.length; i++){

  var button = removeCartButtons[i];
    button.addEventListener('click', removeCartItem );

  }
```

```
  var quantityInputs = document.getElementsByClassName("cart-quantity");
  for (var i = 0; i < quantityInputs.length; i++) {
    var input = quantityInputs[i];
    input.addEventListener("change", quantityChanged);
  }
var addCart = document.getElementsByClassName('add-cart')
for (var i = 0; i < addCart.length; i++) {
  var button = addCart[i];
  button.addEventListener("click", addCartClicked); }
```

У функції "ready()" спочатку вибираються всі елементи з класом "cart-remove", які представляють кнопки для видалення товарів з корзини. Для кожної такої кнопки додається обробник події "click", який викликає функцію "removeCartItem" при натисканні.

Далі вибираються всі елементи з класом "cart-quantity", які представляють поля для зміни кількості товарів у корзині. Для кожного такого поля додається обробник події "change", який викликає функцію "quantityChanged" при зміні значення.

Нарешті, вибираються всі елементи з класом "add-cart", які представляють кнопки для додавання товарів у корзину. Для кожної такої кнопки додається обробник події "click", який викликає функцію "addCartClicked" при натисканні.

Отже, ця функція "ready()". забезпечує інтерактивність корзини покупок, дозволяючи користувачам видаляти товари, змінювати їхню кількість і додавати нові товари до корзини, реагуючи на відповідні дії користувача.

Наступним кроком у розробці кошика буде створення спеціальної кнопки, за допомогою якої можна буде додавати товари а не тільки видаляти їх. Для цього був створений код, що наведено нижче.

```
function addCartClicked(event){
  var button = event.target;
  var shopProducts = button.parentElement;
  var title = shopProducts.getElementsByClassName("product-title")[0].innerText;
  console.log(title); }
```

У функції "addCartClicked(event) " спочатку отримується елемент, на який натиснув користувач "event.target". Потім, через властивість "parentElement" кнопки, отримується батьківський елемент, який містить інформацію про товар "shopProducts". Після цього з елемента "shopProducts" витягується текст з першого елемента, що має клас "product-title", за

допомогою методу "getElementsByClassName("product-title")". Цей текст виводиться в консоль за допомогою "console.log(title)".

Також потрібно додати сумування коштів за для декількох товарів для цього також потрібно створювати нову функцію що буде за це відповідати. Відповідний код наведено нижче.

```
function updatetotal() {
  var cartContent = document.getElementsByClassName("cart-content")[0];
  var cartBoxes = cartContent.getElementsByClassName("cart-box");
  var total = 0;
  for (var i = 0; i < cartBoxes.length; i++) {
    var cartBox = cartBoxes[i];
    var priceElement = cartBox.getElementsByClassName("cart-price")[0];
    var quantityElement = cartBox.getElementsByClassName("cart-quantity")[0];
    var price = parseFloat(priceElement.innerText.replace("₴", ""));
    var quantity = quantityElement.value;
    total = total + (price * quantity);

    total = Math.round(total * 100) / 100;

    document.getElementsByClassName('total-price')[0].innerText = "₴" + total;
  }
}
```

Спочатку вибирається елемент, що містить вміст корзини, за допомогою класу cart-content. З цього елемента витягуються всі елементи з класом "cart-box", які представляють окремі товари в корзині. Ініціалізується змінна "total", яка зберігатиме загальну суму.

Потім для кожного товару (елемента з класом "cart-box") витягується його ціна та кількість. Ціна береться з першого елемента з класом "cart-price" і перетворюється на число, видаляючи символ валюти "₴". Кількість береться з першого елемента з класом "cart-quantity".

Ціна і кількість множаться, і результат додається до загальної суми total. Для точності загальна сума округлюється до двох знаків після коми.

Для заповнення сторінки товарами було створено клас row products mini в якому також було створено клас item що зберігає та виводить інформацію про товар на сторінку (рис. 3.7). При наведені курсора на товар його зображення збільшується у розмірі та з’являються невидимі до цього іконки. При натисканні на фото товару чи його назву, користувача перекидує на сторінку обраного продукту.

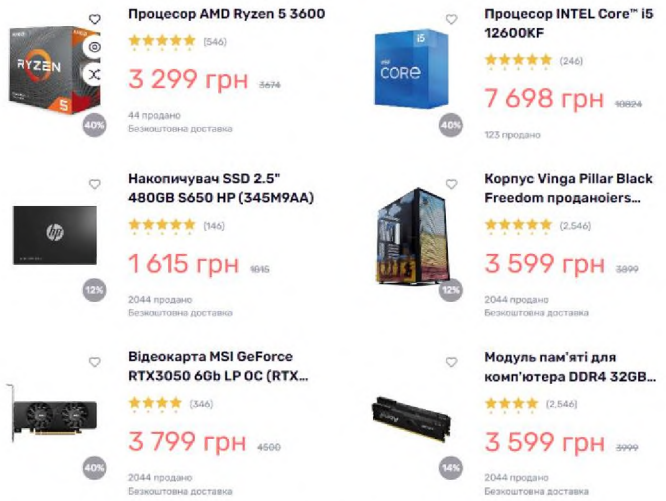

Рисунок 3.7 – Розміщення товарів на сайті

Для створення способу перелистування фото на сторінці продукту було підключено сторонній плагін - cdn.jsdelivr.net а саме Swiper.

Swiper – це безкоштовний і найсучасніший мобільний сенсорний слайдер з апаратним прискоренням переходів і дивовижною нативною поведінкою. Він призначений для використання в мобільних вебсайтах, мобільних вебдодатках і мобільних нативних/гібридних додатках [24].

Після цього у файлі "script.js" було додано 2 скрипти, що наведені нижче. "cdn.jsdelivr.net"

```
var productThumb = new Swiper ('.small-image', {
  loop: true,
  spaceBetween: 10,
  slidesPerView: 3,
  freeMode: true,
  watchSlidesProgress: true,
  breakpoints: {
    481: {
      spaceBetween: 32,
    }
  }
});
var productBig = new Swiper ('.big-image', {
  loop: true,
  autoHeight: true,
  navigation: {
```

```
    nextEl: '.swiper-button-next',
    prevEl: '.swiper-button-prev',
  },
  thumbs: {
    swiper: productThumb,
  }
})
```

Перша програма для перегляду невеликих зображень називається "productThumb". Ця карусель закільцьована "loop:true", має інтервал в 10 пікселів між слайдами "spaceBetween:10" і відображає 3 слайда одночасно "slidesPerView":. Він також використовує вільний режим "freeMode:true", який дозволяє користувачеві прокручувати зображення, не прив'язуючи його до певного слайда, і відстежує хід перегляду слайда "watchSlidesProgress:true". Налаштування "adaptive" визначає, що на екрані шириною більше 481 пікселя проміжок між слайдами збільшується до 32 пікселів (рис. 3.8).

Потім для великого зображення під назвою "productBig" створюється екземпляр "Swiper". Він також циклічний "loop:true" і автоматично регулює висоту слайда відповідно до його вмісту "autoHeight:true". Параметри навігації визначають елементи навігації по слайдах (кнопки "Далі" і "Назад"). Параметр "Thumbs" вказує на те, що в цій каруселі в якості мініатюри навігації використовується "productThumb".

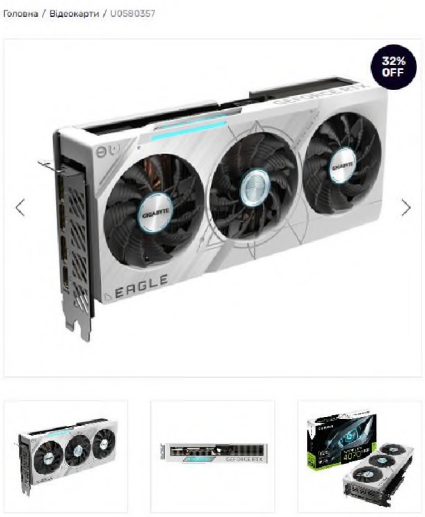

Рисунок 3.8 – Зображення товару на сторінці

Для виводу зображень товару на весь екран, потрібно використано сторонній плагін на основі JavaScript - "fslightbox.js". Для використання потрібно викликати її за допомогою спеціального коду CDN та замінити класи тегу "a", на "data-flightbox". В результаті отримуємо працюючий плеєр для перегляду фото продукту (рис. 3.9) [25].

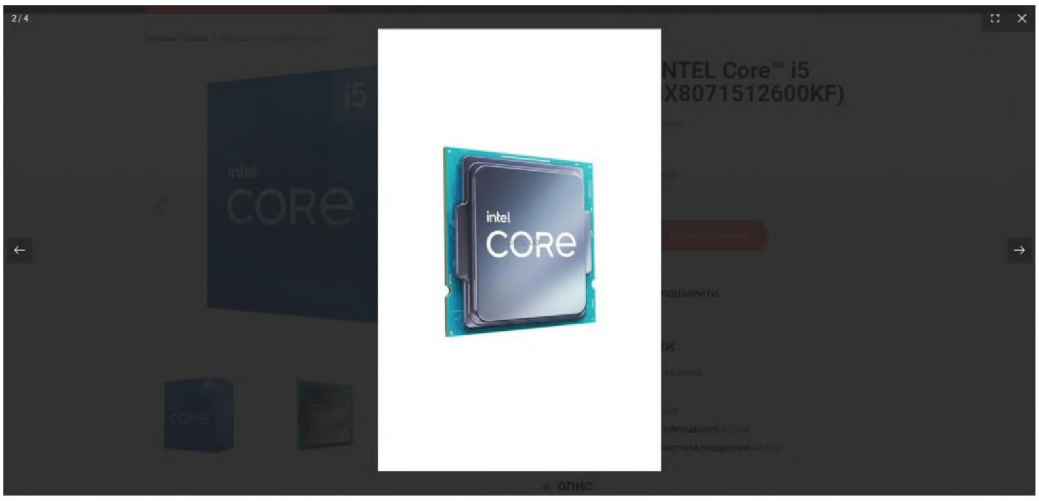

Рисунок 3.9 – Вивід зображення в повноекранний режим

Аналогічний підхід використовується і для інших екранів та компонентів, що необхідні для їх коректної роботи.

3.4 Економічна частина

Розрахунок економічної ефективності при розробці інформаційного сервісу підтримки продажу комп'ютерної техніки складається з кількох ключових етапів. Першим кроком є визначення цілей і завдань сервісу. Основними цілями розробки інформаційного сервісу є збільшення продажів комп'ютерної техніки, підвищення впізнаваності бренду компанії, покращення обслуговування клієнтів та автоматизація процесів продажу та обслуговування. Для досягнення цих цілей планується збільшення обсягу онлайн-замовлень на 20% протягом першого року після запуску сервісу, а

також скорочення часу обробки замовлень на 25%. Залучення нових клієнтів буде здійснюватися за допомогою ефективних онлайн-маркетингових стратегій, а рівень повторних покупок планується підвищити на 20% через створення системи лояльності.

Наступним етапом є аналіз ринку та конкурентів. Для ефективної реалізації інформаційного сервісу необхідно провести ретельний аналіз ринку комп'ютерної техніки, враховуючи поточні тенденції розвитку та прогнозовані зміни. Аналіз конкурентів включає вивчення їхніх сайтів, функціональних можливостей, рівня дизайну, зручності користування та маркетингових стратегій. Це дозволяє виявити сильні та слабкі сторони конкурентів і визначити конкурентні переваги майбутнього сервісу. На основі цього аналізу можна розробити унікальні пропозиції, які приваблять клієнтів і відрізнятимуть сервіс від інших на ринку.

Оцінка витрат на розробку і підтримку сервісу є наступним важливим етапом. Витрати на розробку інформаційного сервісу включають оплату праці розробників, дизайнерів, тестувальників та інших спеціалістів, залучених до проекту. Крім того, витрати на технічну підтримку включають хостинг, доменне ім'я та інші послуги, пов'язані з технічною підтримкою сайту. Витрати на маркетинг і просування сервісу включають SEO-оптимізацію сайту, контекстну рекламу, SMM-кампанії та інші види онлайн-маркетингу.

Для наочності, наведемо таблицю 3.1 з основними статтями витрат на розробку та підтримку інформаційного сервісу:

Таблиця 3.1 – Вартість розробки та підтримки інформаційного сервісу

Найменування	Сума, грн
Розробка сайту	150 000
Закупівля програмного забезпечення	50 000
Технічна підтримка (річна)	100 000
Маркетинг та просування	75 000
Хостинг та доменне ім'я (річна)	10 000
Загальна сума	385 000

Прогнозування доходів є наступним кроком. Воно включає оцінку потенційних доходів від онлайн-продажів. Це передбачає визначення середнього чеку та прогнозованої кількості покупок протягом першого року роботи сервісу. Наприклад, при середньому чеку в 2000 грн і прогнозованій кількості покупок в 3000 на рік, прогнозований дохід становить 6,000,000 грн.

Розрахунок економічних показників включає кілька ключових метрик: простий термін окупності (PBP), чисту приведену вартість (NPV), внутрішню норму прибутковості (IRR) та коефіцієнт рентабельності інвестицій (ROI).

PBP визначає час, за який початкові інвестиції у сервіс окупляться за рахунок отриманих доходів:

$$\text{PBP} = \frac{\text{Початкові інвестиції}}{\text{Середньорічній дохід}}, \tag{3.1}$$

$$\text{PBP} = \frac{400\,000}{6\,000\,000} = 0.067\text{р.} - \text{приблизно 1 місяць}, \tag{3.2}$$

NPV відображає різницю між теперішньою вартістю доходів і витрат протягом певного періоду. Припустимо, що проект триватиме 5 років, ставка дисконту - 10%:

$$\text{NPV} = \sum \left(\frac{\text{Доходи-Витрати}}{(1+r)} \right), \tag{3.3}$$

$$\text{NPV} = \frac{5\,600\,000}{1.1} + \frac{5\,900\,000}{1.21} + \frac{5\,900\,000}{1.331} + \frac{5\,900\,000}{1.4641} + \frac{5\,900\,000}{1.61051} \approx 22\,094\,773 \text{ грн}, \tag{3.4}$$

IRR визначає ставку дисконту, при якій NPV дорівнює нулю. Використовуючи фінансові калькулятори або програмне забезпечення, можна визначити, що IRR для даного проекту становить приблизно 85%.

ROI відображає відношення чистого прибутку до інвестицій, що дозволяє оцінити ефективність вкладених коштів:

$$\text{ROI} = \left(\frac{\text{Чистий прибуток}}{\text{Інвестиції}} \right) \times 100\%, \tag{3.5}$$

$$\text{ROI} = \left(\frac{6\,000\,000 - 400\,000}{400\,000} \right) \times 100\% = 1\,400\%, \tag{3.6}$$

Ці показники дозволяють зробити висновки про економічну ефективність проекту та доцільність його реалізації.

Оцінка ризиків включає визначення можливих ризиків, таких як зміни ринкової ситуації, технічні проблеми або зміни в поведінці споживачів.

Розробка планів на випадок виникнення ризиків включає створення резервного фонду для покриття непередбачених витрат, впровадження системи моніторингу та швидкого реагування на технічні проблеми, а також розробку стратегій адаптації до змін на ринку та поведінки споживачів.

Після завершення всіх етапів формування звіту включає зведення усіх розрахунків у один документ. Детальний опис цілей, завдань та результатів проекту дозволяє чітко представити всі аспекти економічної ефективності. Розрахунки економічної ефективності включають витрати, доходи та ключові економічні показники. Висновки щодо економічної ефективності проекту базуються на аналізі досягнення поставлених цілей та виконання завдань.

Прийняття рішення базується на аналізі отриманих даних та оцінці результатів розрахунків. Важливо враховувати відповідність результатів поставленим цілям та завданням проекту. На основі проведеного аналізу та розрахунків приймається рішення про доцільність подальшого інвестування у розробку та впровадження сервісу.

ВИСНОВКИ

При виконанні кваліфікаційної роботи було проведено огляд історії появи та розвитку перших вебсайтів. Було виконано аналіз існуючих технологій розробки, які використовуються при розробці різних вебсайтів та інформаційних системах, такі як HTML CSS та JavaScript.

При підготовці другого розділу, було зібрано інформацію та приклади інформаційних сервісів, що будуть використовуватися як приклад, на протязі всієї роботи. Були проаналізовані та обрані, технології і програмне забезпечення, котре знадобиться для виконання розділів дипломної роботи.

У третьому розділі, на основі зібраної раніше інформації, було обрано набір інструментів, які найбільш повно підходять для розробки комерційного вебсайту, а також коротко описано структуру проєкту і деякі особливості обраного набору технологій щодо створення дизайну і коду, які формують цей проєкт. Був повністю розроблений дизайн та код. Було додано кошик для товарів та спеціальний плеєр для перегляду зображень товарів у повному розмірі. Також була додана оптимізація інтерфейсу під мобільні пристрої.

При виконанні кваліфікаційної роботи були використані такі технології, як HTML5, CSS, JavaScript, та сторонні плагіни такі як:

1) Плагін "cdn.jsdelivr.net swiper", для додавання функції перелистування фото на сторінці товару;

2) Плагін "fslightbox.js", для додання функції перегляду фото у повноекранному режимі.

Проведено розрахунок економічної ефективності проєкту. Загальні витрати на розробку та підтримку інформаційного сервісу підтримки продажу склали 385 000 грн. Оцінка потенційних доходів дозволила визначити середній чек та прогнозовану кількість покупок, а також додаткові доходи від реклами. PBP склав 2 роки, NPV була позитивною, а IRR перевищила очікуваний рівень.