МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД «УНІВЕРСИТЕТ «КРОК»

Фаховий коледж Університету «КРОК»

ДИПЛОМНА РОБОТА

за темою

«Розробка та створення сайту магазину електротехніки»

Студент 4 курсу групи КН-20К

Кривой Назарій Олексійович

(прізвище, ім'я та по-батькові студента)

Керівник дипломної роботи

К.т.н. доцент

(посада керівника)

Пантєєв Роман Леонідович

(прізвище, ім'я та по-батькові керівника)

До захисту

(резолюція «До захисту»)

_____

(підпис студента)

11.06.2024

(дата)

_____

(підпис викладача)

Київ, 2024 рік

Скорочення

CMS - Content Management System (Система керування вмістом)

SEO - Search Engine Optimization (Пошукова оптимізація)

HTML - HyperText Markup Language (Мова гіпертекстової розмітки)

CSS - Cascading Style Sheets (Каскадні таблиці стилів)

JS - JavaScript (Мова програмування JavaScript)

API - Application Programming Interface (Інтерфейс прикладного програмування)

REST - Representational State Transfer (Передача репрезентаційного стану)

CRUD - Create, Read, Update, Delete (Створення, читання, оновлення, видалення)

JWT - JSON Web Token (JSON веб-токен)

AJAX - Asynchronous JavaScript and XML (Асинхронний JavaScript та XML)

SDK - Software Development Kit (Набір засобів розробки програмного забезпечення)

SQL - Structured Query Language (Мова структурованих запитів)

XSS - Cross-Site Scripting (Міжсайтовий скриптинг)

SSL - Secure Sockets Layer (Рівень захищених сокетів)

AWS - Amazon Web Services (Хмарні сервіси Amazon)

SaaS - Software as a Service (Програмне забезпечення як послуга)

Зміст

Вступ

В сучасному світі технології розвиваються з неймовірною швидкістю, і електронна комерція стає невід'ємною частиною економіки багатьох країн. З кожним роком все більше споживачів віддають перевагу онлайн-шопінгу перед традиційними методами придбання товарів. Це обумовлено зручністю, швидкістю та широким асортиментом, які пропонують інтернет-магазини. У зв'язку з цим, розвиток та створення ефективних веб-сайтів для продажу електротехніки стає актуальним завданням для підприємців і розробників.

Мета даної дипломної роботи - розробка та створення веб-сайту для інтернет-магазину електротехніки, який відповідає сучасним вимогам ринку та потребам користувачів. У рамках роботи буде проведено аналіз існуючих рішень і технологій, що використовуються в сфері електронної комерції, обрано найефективніші інструменти для розробки сайту, а також реалізовано функціонал, що забезпечує зручність користування та високу продуктивність.

На сьогоднішній день конкуренція на ринку інтернет-магазинів електротехніки є досить високою. Це спонукає до ретельного вивчення конкурентного середовища, аналізу сильних і слабких сторін існуючих рішень, що дозволить уникнути поширених помилок та впровадити найкращі практики у власному проекті. У процесі аналізу будуть розглянуті популярні платформи для створення інтернет-магазинів, такі як WooCommerce, Shopify, Magento та інші, з метою вибору найоптимальнішої для розробки даного проекту.

Особлива увага буде приділена вибору технологій для реалізації сайту. Враховуючи сучасні тенденції у веб-розробці, буде розглянуто використання таких фреймворків і бібліотек, як React, Angular, Vue.js для фронтенду, а також Node.js, Python, PHP для бекенду. Вибір бази даних також відіграє важливу роль у забезпеченні продуктивності та надійності сайту, тому будуть проаналізовані різні варіанти, такі як MySQL, PostgreSQL, MongoDB.

Отже, даний дипломний проект охоплює повний цикл розробки інтернет-магазину електротехніки, від аналізу ринку та вибору технологій до

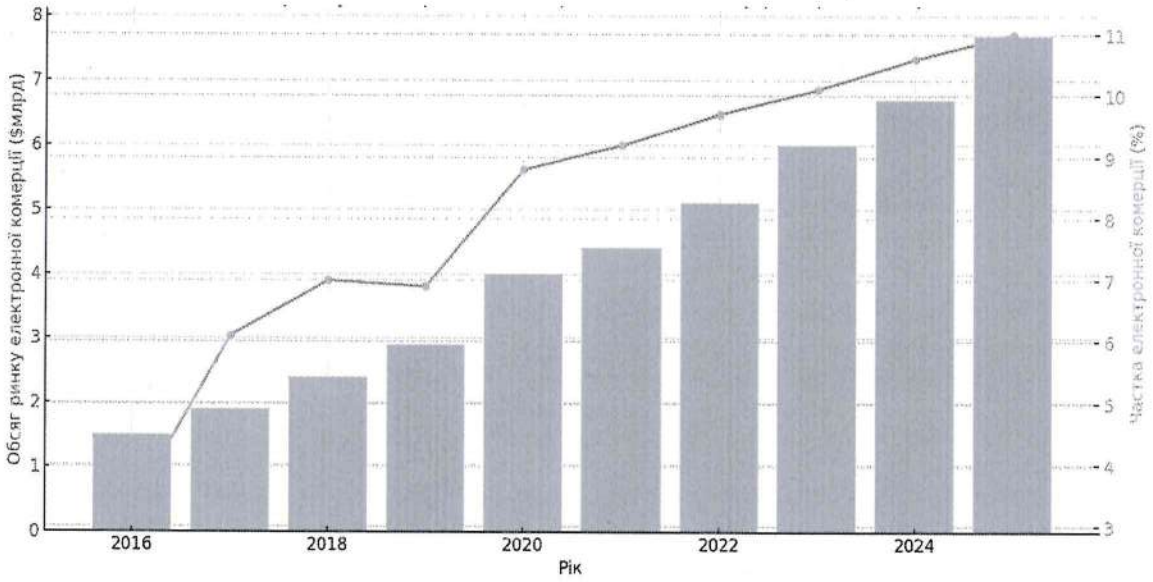
реалізації та тестування функціоналу. Реалізація цього проекту сприятиме підвищенню конкурентоспроможності на ринку електронної комерції та забезпечить користувачам зручний і надійний інструмент для придбання електротехнічних товарів.

Розділ 1. Аналіз існуючих рішень та технологій

1.1 Аналіз ринку інтернет-магазинів електротехніки

Інтернет-магазини електротехніки на сьогоднішній день є невід'ємною частиною сучасного ринку електронної комерції. З огляду на стрімкий розвиток цифрових технологій та зростання попиту на електронні товари, створення ефективного і функціонального веб-сайту для магазину електротехніки є актуальним завданням. Аналіз ринку інтернет-магазинів електротехніки включає кілька ключових аспектів, що визначають успіх проекту.

Ринок інтернет-магазинів електротехніки постійно зростає,



демонструючи стабільне збільшення обсягів продажів.

Рис.1.1.Динаміка ринку електронної комерції та її частина у розробній торгівлі

Це обумовлено кількома факторами: зростанням числа користувачів інтернету, розвитком мобільних технологій, а також зміною поведінки споживачів, які все частіше обирають онлайн-шопінг замість традиційних магазинів. Сучасні тренди включають персоналізацію покупок, використання штучного інтелекту для покращення взаємодії з клієнтами, а також інтеграцію з соціальними мережами для залучення ширшої аудиторії.

На ринку інтернет-магазинів електротехніки домінують кілька великих гравців, таких як Rozetka, Comfy, Eldorado, Foxtrot та інші. Ці компанії мають добре розвинену інфраструктуру, зручні веб-сайти з широким асортиментом товарів, а також розвинену систему логістики та підтримки клієнтів. Вони пропонують конкурентоспроможні ціни, різноманітні акції та програми лояльності, що дозволяє їм утримувати лідерські позиції на ринку.(рис.1.2)

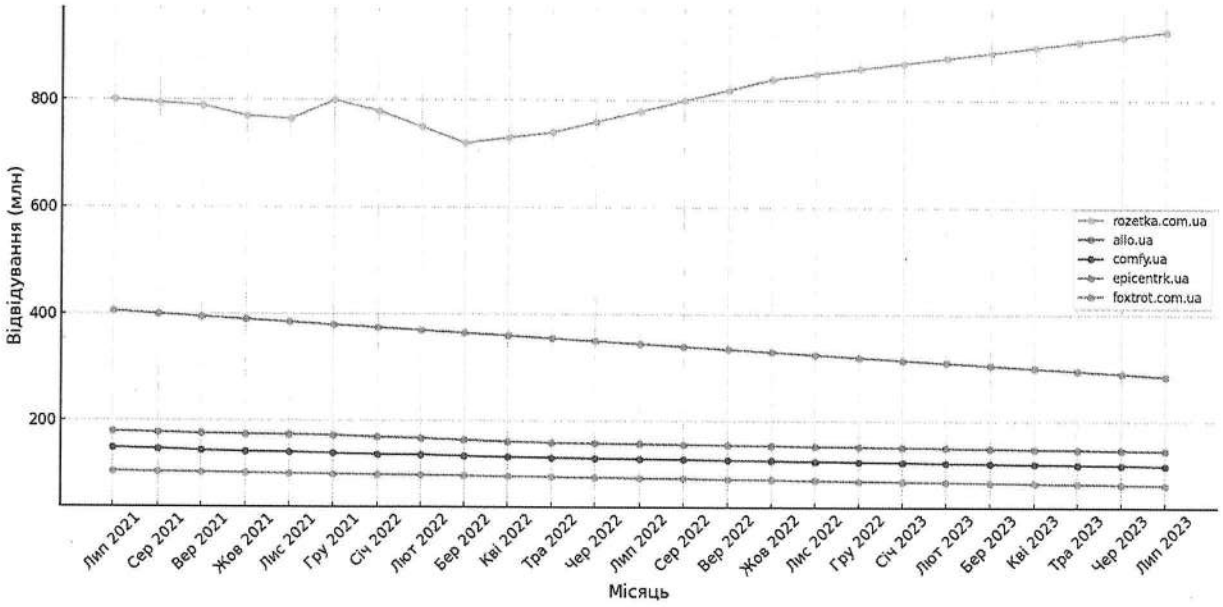


Рис.1.2.Відвідування сайтів з липня 2021 по липень 2023.

Переваги існуючих інтернет-магазинів електротехніки включають:

- Широкий асортимент товарів, що дозволяє задовольнити потреби різних категорій споживачів.
- Зручні інтерфейси сайтів з простим і зрозумілим навігаційним меню.
- Різноманітні методи оплати та доставки, що забезпечує комфорт покупців.
- Системи відгуків та рейтингів товарів, що допомагають споживачам робити обґрунтований вибір.

Недоліки, з якими можуть стикатися користувачі:

- Висока конкуренція, що призводить до необхідності постійного моніторингу цін та пропозицій.
- Відсутність персоналізованого підходу у великих компаніях, що може знижувати рівень задоволеності клієнтів.

- Можливі проблеми з доставкою та післяпродажним обслуговуванням, особливо у пікові періоди продажів.

Сучасні споживачі очікують від інтернет-магазинів високої якості обслуговування, зручності використання веб-сайту, швидкої доставки та можливості легко повернути або обміняти товар. Важливим аспектом є наявність докладної інформації про товари, включаючи технічні характеристики, відгуки інших покупців, а також можливість отримати консультацію від фахівців.

Для успішного функціонування інтернет-магазину електротехніки необхідно використовувати сучасні технологічні рішення. Це включає вибір надійної CMS, що підтримує інтеграцію з платіжними системами, CRM для управління взаємодією з клієнтами, а також системи аналітики для моніторингу поведінки користувачів та ефективності маркетингових кампаній.

Аналіз ринку інтернет-магазинів електротехніки показує, що для успішного входження на цей ринок необхідно запропонувати конкурентоспроможні ціни, широкий асортимент товарів, високу якість обслуговування та зручний веб-сайт. Використання сучасних технологій і орієнтація на потреби споживачів дозволить створити ефективний інтернет-магазин, здатний конкурувати з провідними гравцями ринку.

1.2 Огляд технологій для створення веб-сайтів

Створення веб-сайту – це складний і багатогранний процес, що включає вибір правильних технологій для забезпечення функціональності, зручності та безпеки. У цьому огляді ми розглянемо основні технології, які використовуються для створення сучасних веб-сайтів, зокрема для інтернет-магазинів електротехніки.

Основні мови програмування для розробки веб-сайтів включають HTML, CSS, JavaScript, PHP та Python. HTML (HyperText Markup Language) відповідає за структуру сторінок, CSS (Cascading Style Sheets) – за їхнє оформлення, а

JavaScript – за динамічну взаємодію користувачів з веб-сторінкою. PHP і Python використовуються для серверної логіки та інтеграції з базами даних.

- HTML і CSS: Вони є базовими технологіями для створення фронтенду веб-сайту. HTML дозволяє створювати структуровані документи, а CSS – стилізувати їх, забезпечуючи привабливий і зручний інтерфейс.
- JavaScript: Ця мова дозволяє додавати інтерактивні елементи, такі як форми, кнопки, модальні вікна тощо. Популярні бібліотеки та фреймворки, такі як React, Angular і Vue.js, розширюють можливості JavaScript і спрощують створення складних інтерфейсів.

CMS є важливим інструментом для створення та управління контентом веб-сайтів без необхідності глибоких знань у програмуванні. Популярні CMS включають WordPress, Joomla, Drupal та Magento.

- WordPress: Найбільш популярна CMS у світі, відома своєю простотою у використанні та великою кількістю плагінів і тем. Підходить для створення різноманітних сайтів, включаючи блоги, корпоративні сайти та інтернет-магазини.
- Joomla: CMS з більш складною структурою, що надає більшу гнучкість і потужність для створення різних типів сайтів. Має значну кількість розширень і шаблонів.
- Drupal: Потужна CMS, відома своєю масштабованістю та безпекою. Використовується для створення великих і складних сайтів, таких як урядові портали та великі корпоративні сайти.
- Magento: Спеціалізована CMS для створення інтернет-магазинів, що пропонує широкий спектр функцій для управління товарами, замовленнями та платіжними системами.

Для забезпечення надійної роботи веб-сайтів використовуються різноманітні серверні технології та бази даних. Найбільш популярними є Apache, Nginx, MySQL, PostgreSQL та MongoDB.

- Apache та Nginx: Веб-сервери, які забезпечують доставку веб-сторінок користувачам. Apache відомий своєю надійністю та гнучкістю, тоді як Nginx – високою продуктивністю і низьким споживанням ресурсів.
- MySQL та PostgreSQL: Реляційні бази даних, що використовуються для зберігання та управління даними. MySQL є найбільш популярною базою даних для веб-сайтів завдяки своїй простоті та надійності, тоді як PostgreSQL пропонує більшу функціональність та гнучкість.
- MongoDB: Документо-орієнтована база даних, що підходить для зберігання неструктурованих даних і забезпечує високу масштабованість.

Фреймворки значно спрощують процес розробки, надаючи готові рішення для типових завдань. Найпопулярніші фреймворки включають Laravel, Django, Ruby on Rails та Express.js.

- Laravel: PHP-фреймворк, що пропонує елегантний синтаксис і багатий набір інструментів для розробки веб-додатків.
- Django: Фреймворк для Python, відомий своєю швидкістю розробки та безпекою. Підходить для створення масштабованих веб-додатків.
- Ruby on Rails: Фреймворк для Ruby, що забезпечує швидку розробку завдяки конвенційній конфігурації та великій кількості вбудованих інструментів.
- Express.js: Легкий і гнучкий фреймворк для Node.js, що дозволяє швидко створювати серверні додатки на JavaScript.

Для забезпечення якості веб-сайтів використовуються різноманітні інструменти для тестування та оптимізації, такі як Selenium, Google Lighthouse, JMeter та інші.

- Selenium: Інструмент для автоматизованого тестування веб-додатків, що дозволяє перевіряти функціональність сайту на різних браузерах.
- Google Lighthouse: Інструмент для аналізу продуктивності, якості коду та SEO оптимізації веб-сайтів.

- JMeter: Інструмент для тестування продуктивності та навантаження, що дозволяє визначити, як веб-сайт справляється з високим трафіком.

Сучасні веб-сайти часто використовують хмарні сервіси та DevOps практики для забезпечення безперервної інтеграції та доставки, а також для масштабованості та надійності.

- AWS, Google Cloud, Azure: Хмарні платформи, що надають широкий спектр послуг для розробки, розгортання та масштабування веб-додатків.
- Docker, Kubernetes: Інструменти для контейнеризації та оркестрації додатків, що дозволяють легко керувати інфраструктурою та забезпечувати безперервну доставку.

Огляд технологій для створення веб-сайтів показує, що існує безліч інструментів і підходів, що дозволяють створювати ефективні, функціональні та зручні веб-сайти. Вибір конкретних технологій залежить від вимог проекту, досвіду команди розробників та ресурсів, що є в розпорядженні. Для успішного створення сайту магазину електротехніки важливо вибрати надійну CMS, забезпечити інтеграцію з платіжними системами та методами доставки, а також застосувати сучасні підходи до розробки, тестування та оптимізації веб-додатків.

1.3 Порівняння CMS для інтернет-магазинів

Вибір системи керування контентом (CMS) для інтернет-магазину є критично важливим кроком, що визначає успіх всього проекту. Сучасний ринок пропонує безліч варіантів CMS, кожен з яких має свої особливості, переваги та недоліки. Важливо підібрати таку платформу, яка буде відповідати потребам вашого бізнесу, забезпечуючи належну функціональність, зручність управління та безпеку. Розглянемо найпопулярніші CMS для інтернет-магазинів, їхні характеристики та порівняння.

WordPress разом із плагіном WooCommerce є найпопулярнішим рішенням для створення інтернет-магазинів. Величезна спільнота

користувачів і розробників, безліч доступних тем і плагінів роблять цей варіант привабливим для багатьох підприємців.

Переваги:

- Популярність і спільнота: Велика кількість ресурсів і підтримка з боку спільноти.
- Легкість у використанні: Інтуїтивно зрозумілий інтерфейс, велика кількість документації.
- Масштабованість: Можливість додавання нових функцій за допомогою плагінів.

Недоліки:

- Швидкодія: Може бути повільнішою при великій кількості товарів.
- Безпека: Часто стає мішенню для хакерів, потребує регулярних оновлень і додаткових заходів безпеки.

Magento є потужною платформою для великих інтернет-магазинів, яка пропонує широкий спектр функцій і можливостей для кастомізації.

Переваги:

- Функціональність: Широкий набір інструментів для управління магазином.
- Масштабованість: Підходить для великих проектів з високим трафіком.
- Гнучкість: Можливість розширення функціоналу за допомогою модулів і власного коду.

Недоліки:

- Складність у використанні: Вимагає технічних знань для налаштування та управління.
- Вимогливість до ресурсів: Потребує потужних серверів.

Shopify – це SaaS (software as a service) платформа, що забезпечує швидкий старт і простоту в користуванні.

Переваги:

- Легкість у використанні: Простий інтерфейс, швидке налаштування.

- Хостинг: Платформа надає хостинг, безпеку і технічну підтримку.
- Платежі: Інтеграція з багатьма платіжними системами.

Недоліки:

- Обмежена кастомізація: Менше можливостей для налаштування в порівнянні з відкритими платформами.
- Вартість: Щомісячна плата і комісія за транзакції можуть бути значними для малих бізнесів.

OpenCart є популярною CMS з відкритим вихідним кодом, що надає достатньо можливостей для створення інтернет-магазину.

Переваги:

- Легкість у використанні: Інтуїтивно зрозумілий інтерфейс.
- Безкоштовність: Відкритий вихідний код.
- Масштабованість: Підтримка багатомовності та мультивалютності.

Недоліки:

- Обмежена підтримка: Менша спільнота і менше готових рішень у порівнянні з іншими CMS.
- Функціональність: Менше вбудованих функцій, може вимагати додаткових модулів.

Ринок CMS для інтернет-магазинів пропонує різноманітні варіанти, кожен з яких має свої унікальні особливості та можливості. Вибір платформи залежить від конкретних потреб вашого бізнесу, масштабів проекту та технічних можливостей команди. WordPress з WooCommerce та Shopify є чудовими варіантами для малих і середніх бізнесів завдяки своїй простоті та швидкому налаштуванню. Magento підходять для великих проектів з високими вимогами до функціональності та масштабованості. OpenCart може стати оптимальним вибором для тих, хто шукає безкоштовне рішення з достатньою базовою функціональністю.

Розділ 2. Розробка сайту

2.1 Вибір технологій для розробки

Вибір технологій для розробки сайту магазину електротехніки є надзвичайно важливим етапом, який значною мірою впливає на функціональність, продуктивність, безпеку та зручність використання майбутнього веб-ресурсу. Для створення сучасного та ефективного інтернет-магазину необхідно ретельно підійти до вибору мов програмування, фреймворків, систем управління контентом (CMS), баз даних, серверних технологій, а також інструментів для тестування та оптимізації.

Однією з основних технологій, що лежить в основі будь-якого веб-сайту, є HTML. Це мова розмітки, яка визначає структуру веб-сторінок. Використання останньої версії HTML5 дозволяє додати на сайт мультимедійні елементи та покращити семантику сторінок. Поряд з HTML, CSS відіграє ключову роль в оформленні сайту. Застосування CSS3 забезпечує підтримку сучасних стилів, анімацій та медіа-запитів для адаптивного дизайну, що є важливим для забезпечення коректного відображення сайту на різних пристроях.

JavaScript додає динамічну взаємодію та інтерактивність на веб-сторінках. Сучасні фреймворки та бібліотеки JavaScript, такі як React, Angular та Vue.js, значно спрощують розробку складних користувацьких інтерфейсів. Наприклад, React забезпечує високу продуктивність завдяки використанню віртуального DOM і дозволяє створювати додатки з компонентною архітектурою, що спрощує підтримку та масштабування проекту.

Для серверної частини можна використовувати різні мови програмування та фреймворки. PHP є одним з найпопулярніших рішень для веб-розробки, а фреймворк Laravel надає зручний і елегантний синтаксис для швидкої розробки веб-додатків. Альтернативою PHP може бути Python з

фреймворком Django, який також забезпечує високу продуктивність та безпеку. Node.js разом з Express.js є ще одним потужним варіантом для серверної розробки, що дозволяє виконувати JavaScript на сервері і забезпечує асинхронну обробку запитів.

Система управління контентом (CMS) відіграє важливу роль в управлінні сайтом. WordPress, разом з плагіном WooCommerce, є популярним вибором для створення інтернет-магазинів завдяки своїй простоті використання та величезній кількості доступних плагінів і тем. Для більш масштабних проектів можна використовувати Magento, яка забезпечує високу гнучкість та масштабованість, або Shopify, яка пропонує готове рішення з хостингом і технічною підтримкою.

Важливим аспектом є вибір бази даних. MySQL і PostgreSQL є популярними реляційними базами даних, що забезпечують надійність і продуктивність. MongoDB, як документо-орієнтована база даних, підходить для зберігання неструктурованих даних і забезпечує високу масштабованість.

Серверні технології, такі як Apache та Nginx, забезпечують надійну роботу веб-сайту. Apache є гнучким і надійним веб-сервером, а Nginx забезпечує високу продуктивність і низьке споживання ресурсів, що є важливим для сайтів з високим трафіком.

Для забезпечення якості веб-додатка важливо використовувати інструменти для тестування та оптимізації. Selenium дозволяє автоматизувати тестування функціональності сайту на різних браузерах. Google Lighthouse допомагає аналізувати продуктивність, якість коду та SEO оптимізацію, а JMeter дозволяє тестувати продуктивність і навантаження сайту.

Використання хмарних сервісів, таких як AWS, Google Cloud або Azure, надає широкі можливості для розробки, розгортання та масштабування веб-додатків. Інструменти для контейнеризації, такі як Docker та Kubernetes, забезпечують легке керування інфраструктурою та безперервну доставку.

Отже, вибір технологій для розробки сайту магазину електротехніки є багатогранним процесом, що включає врахування багатьох аспектів. Важливо

обирати ті технології, які найкраще відповідають вимогам проекту, забезпечують високу продуктивність, безпеку та зручність використання для кінцевих користувачів.

2.2. Розробка структури сайту

Розробка структури сайту є одним з найважливіших етапів створення веб-ресурсу, оскільки від цього залежить, наскільки зручно користувачам буде взаємодіяти з вашим сайтом, знаходити потрібну інформацію та здійснювати покупки. Структура сайту визначає логіку розташування розділів, сторінок і контенту, що значною мірою впливає на загальне враження від сайту, а також на його ефективність у досягненні бізнес-цілей.

Головною сторінкою вашого сайту є його вітрина, яка першою зустрічає користувачів. На головній сторінці повинна бути представлена основна інформація про ваш магазин, включаючи акційні пропозиції, новинки, популярні товари та основні категорії продукції. Важливо створити привабливий дизайн, який привертатиме увагу відвідувачів та заохочуватиме їх до подальшого перегляду сайту. Головна сторінка також повинна мати зручну навігацію, пошуковий рядок, кнопки для швидкого доступу до кошика та акаунта користувача, що забезпечить зручність використання.

Каталог товарів є центральним елементом інтернет-магазину, адже саме тут користувачі проводять більшість часу, переглядаючи доступні продукти. Важливо створити чітку і логічну структуру каталогу, розділивши товари на категорії та підкатегорії. Це дозволить користувачам швидко знаходити потрібні товари за допомогою фільтрів, таких як бренд, ціна, технічні характеристики тощо. Кожна категорія повинна мати зрозумілий інтерфейс та бути легко доступною з головного меню.

Сторінка товару є ключовим елементом для прийняття рішення про покупку. На цій сторінці повинна бути представлена детальна інформація про товар, включаючи його опис, технічні характеристики, фотографії, відеоогляди, відгуки користувачів та доступні опції (наприклад, колір або

комплектацію). Важливо, щоб сторінка товару мала чітку структуру та зручний інтерфейс для додавання товару до кошика, перегляду схожих продуктів та швидкого оформлення замовлення.

Кошик є місцем, де користувачі можуть переглядати та редагувати свій вибір перед оформленням замовлення. Тут повинні бути вказані всі вибрані товари, їх кількість, загальна вартість замовлення, можливі знижки та підсумкова сума до оплати. Важливо забезпечити зручний інтерфейс для видалення товарів з кошика, зміни їх кількості та переходу до оформлення замовлення.

Сторінка оформлення замовлення повинна бути максимально простою та інтуїтивною, щоб користувачі могли швидко і без зайвих труднощів завершити покупку. Необхідно передбачити форми для введення контактних даних, вибору способу доставки та оплати. Важливо забезпечити підтримку різних платіжних систем та методів доставки для зручності користувачів, а також можливість використання промокодів та бонусів.

Особистий кабінет користувача дозволяє управляти своїми замовленнями, переглядати історію покупок, змінювати контактні дані та налаштування профілю. Тут користувачі можуть також налаштувати підписку на новини та акційні пропозиції, що сприяє підвищенню лояльності клієнтів та залученню їх до повторних покупок.

Інформаційні сторінки, такі як «Про нас», «Контакти», «Доставка і оплата», «Гарантії та повернення» та інші, надають користувачам важливу інформацію про вашу компанію та умови співпраці. Ці сторінки допомагають створити довіру до вашого магазину, відповідаючи на основні питання та надаючи необхідні контактні дані.

Навігація є ключовим елементом структури сайту, оскільки вона визначає, наскільки легко користувачі зможуть знаходити потрібну інформацію. Основне меню повинно бути розташоване у верхній частині сторінки і включати всі основні розділи сайту. Важливо передбачити використання «хлібних крихт» (breadcrumbs), які дозволяють користувачам

легко орієнтуватися в структурі сайту та повертатися на попередні сторінки. Додатково, пошуковий рядок повинен бути розміщений на видному місці, щоб користувачі могли швидко знайти потрібний товар за ключовими словами. Футер сайту може включати додаткові посилання на інформаційні сторінки, соціальні мережі та контактну інформацію.

Зважаючи на зростаючу популярність мобільних пристроїв, важливо забезпечити адаптивний дизайн сайту(рис.2.1), який буде коректно відображатися на смартфонах та планшетах. Це означає, що структура сайту повинна автоматично підлаштовуватися під розміри екрана пристрою, забезпечуючи зручне використання на будь-якому девайсі. Навігація, кнопки та інші елементи інтерфейсу повинні бути оптимізовані для сенсорного керування, що включає збільшені елементи та спрощену структуру сторінок для зручності користувачів.

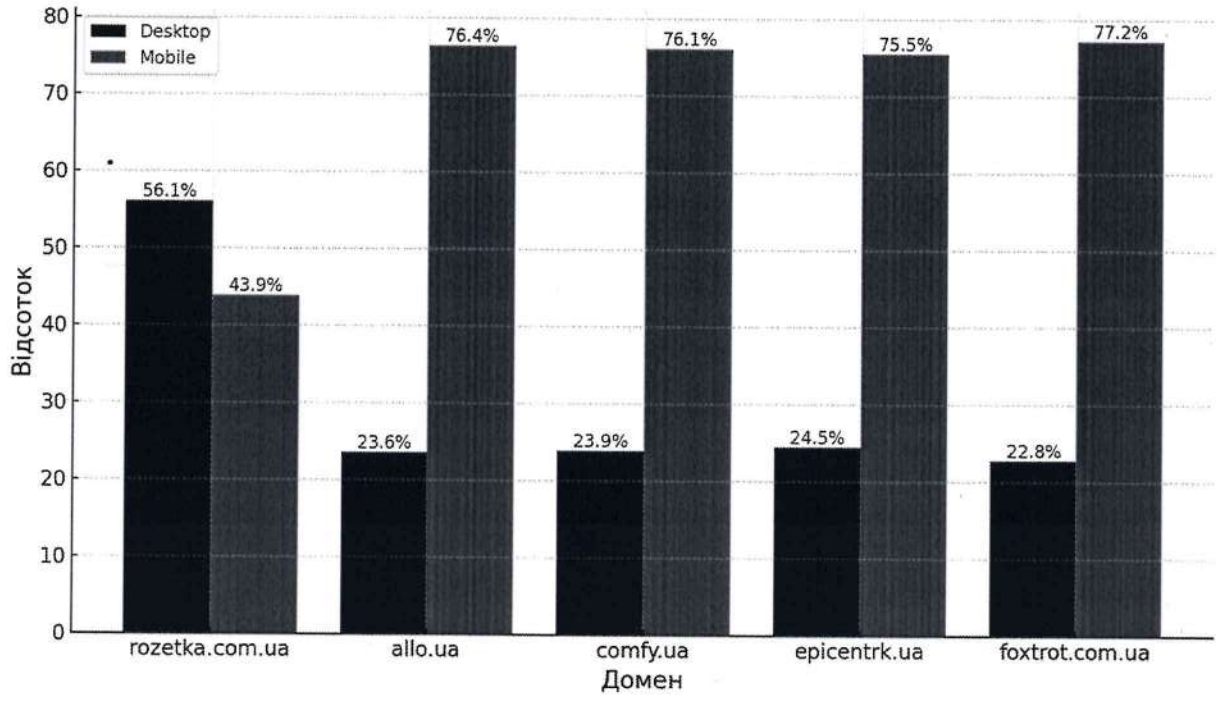


Рис. 2.1.Розподіл пристроїв з липня 2021 по липень 2023.

Оптимізація структури сайту для пошукових систем (SEO) є важливим аспектом, який впливає на видимість вашого магазину в пошукових системах. Важливо створити зрозумілі та логічні URL, використовувати ключові слова в

заголовках та описах сторінок, а також забезпечити швидке завантаження сторінок. Важливо також створити XML-карту сайту та використовувати мета-теги для покращення індексації сайту пошуковими системами. Це допоможе вашому магазину отримати більш високу позицію в результатах пошуку, залучаючи більше потенційних клієнтів.

Інтеграція з соціальними мережами є важливим аспектом сучасного інтернет-магазину. Це дозволяє користувачам ділитися товарами зі своїми друзями та підписуватися на новини вашого магазину. Важливо розмістити кнопки соціальних мереж на сторінках товарів, головній сторінці та в особистому кабінеті користувачів. Це сприятиме підвищенню популярності вашого магазину та залученню нових клієнтів через соціальні платформи.

Таким чином, розробка структури сайту включає створення логічної та інтуїтивно зрозумілої навігації, забезпечення адаптивності дизайну, оптимізацію для пошукових систем та інтеграцію з соціальними мережами. Важливо врахувати всі ці аспекти, щоб створити зручний та ефективний інтернет-магазин, який відповідатиме потребам користувачів та допомагатиме досягати бізнес-цілей.

2.3. Створення дизайну сайту

Створення дизайну сайту є ключовим етапом у розробці будь-якого інтернет-ресурсу, включаючи інтернет-магазин електротехніки. Від якості дизайну залежить не тільки естетична привабливість сайту, але й його функціональність, зручність використання та здатність привернути й утримати увагу користувачів. Ретельний підхід до дизайну допомагає створити позитивний досвід для відвідувачів сайту, що в кінцевому результаті може призвести до збільшення конверсій та зростання продажів.

Першим кроком у створенні дизайну є розробка концепції. Це включає визначення загального стилю та настрою сайту, що відповідає бренду та цільовій аудиторії. Для інтернет-магазину електротехніки важливо створити сучасний та професійний вигляд, який підкреслює якість і надійність

пропонованих товарів. Концепція дизайну повинна враховувати кольорову гаму, типографіку, використання графічних елементів та фотографій.

Кольорова гамма є важливим елементом дизайну, оскільки кольори впливають на сприйняття та емоції користувачів. Для інтернет-магазину електротехніки варто обрати кольори, які асоціюються з технологіями, інноваціями та надійністю. Це можуть бути відтінки синього, сірого, чорного та білого. Важливо забезпечити гармонійне поєднання кольорів, щоб дизайн виглядав професійно та не відволікав увагу від основного контенту. Крім основних кольорів, можна використовувати акцентні кольори для виділення важливих елементів, таких як кнопки «Купити» або «Додати до кошика».

Типографіка грає ключову роль у забезпеченні читабельності та зручності використання сайту. Вибір шрифтів повинен відповідати загальному стилю сайту та бути легко читаним на різних пристроях. Для основного тексту варто обрати прості та зрозумілі шрифти, такі як Arial, Helvetica, Roboto або (рис.2.2). Заголовки можуть бути більш виразними, щоб привертати увагу до важливих розділів та продуктів. Важливо забезпечити достатній контраст між текстом та фоном, а також використовувати різні розміри шрифтів для створення ієрархії інформації.

Макет сайту визначає розташування основних елементів на сторінках та впливає на зручність навігації. Важливо створити логічний та інтуїтивно зрозумілий макет, який допоможе користувачам швидко знаходити потрібну інформацію. Головна сторінка повинна містити ключові елементи, такі як логотип, меню навігації, пошуковий рядок, банери з акційними пропозиціями та новинками. Каталог товарів має бути структурований таким чином, щоб користувачі могли легко переглядати категорії та фільтрувати товари за різними параметрами.

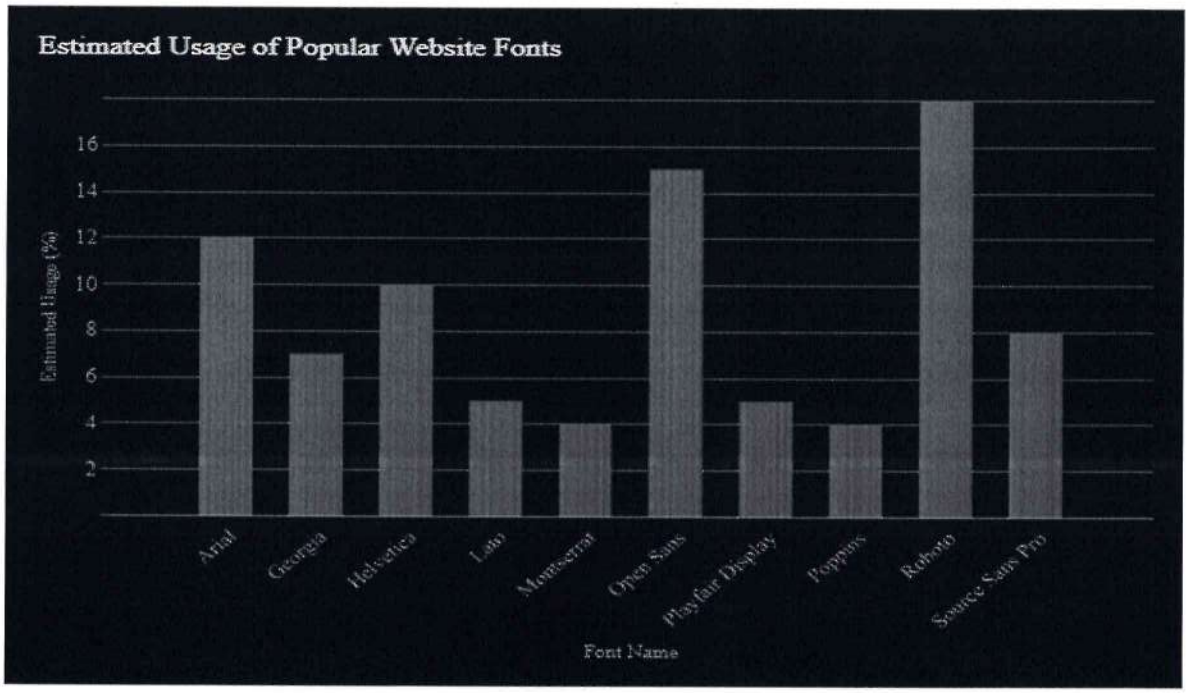


Рис.2.2.Приблизне використання популярних форм веб-сайту.

Сторінка товару повинна містити великий і якісний зображення продукту, детальний опис, технічні характеристики, відгуки користувачів та кнопки для додавання товару до кошика. Кошик і сторінка оформлення замовлення повинні бути максимально простими та зручними для використання, з чіткими інструкціями та мінімальною кількістю кроків.

Графічні елементи, такі як іконки, банери та ілюстрації, можуть значно покращити візуальне сприйняття сайту та зробити його більш привабливим для користувачів. Важливо використовувати якісні та професійні графічні матеріали, які відповідають загальному стилю сайту. Фотографії товарів повинні бути високої якості та відображати продукти з різних ракурсів, що допоможе користувачам краще зрозуміти їх характеристики та зробити правильний вибір.

Адаптивний дизайн є необхідністю в сучасному веб-дизайні, оскільки велика кількість користувачів відвідують сайти з мобільних пристроїв. Це означає, що дизайн сайту повинен автоматично підлаштовуватися під розміри екранів різних пристроїв, забезпечуючи зручне використання на смартфонах, планшетах та комп'ютерах. Адаптивний дизайн включає оптимізацію макету,

шрифтів, зображень та інших елементів інтерфейсу для різних розширень екрана. Важливо провести тестування дизайну на різних пристроях, щоб переконатися в його коректній роботі та зручності використання.

UX та UI дизайн є ключовими аспектами в створенні успішного інтернет-магазину. UX дизайн зосереджується на створенні позитивного досвіду для користувачів, забезпечуючи логічну навігацію, зручність використання та ефективність взаємодії з сайтом. UI дизайн відповідає за візуальну привабливість сайту, включаючи кольорову гаму, типографіку, графічні елементи та макет.

Важливо проводити дослідження цільової аудиторії та тестування прототипів, щоб зрозуміти потреби та очікування користувачів. Це допоможе створити дизайн, який буде не тільки привабливим, але й функціональним та зручним для використання. Врахування принципів UX/UI дизайну дозволить створити інтуїтивно зрозумілий інтерфейс, який сприятиме збільшенню конверсій та задоволеності користувачів.

Візуальна консистентність є важливим аспектом у створенні дизайну сайту. Це означає, що всі елементи дизайну повинні відповідати єдиному стилю та виглядати гармонійно на всіх сторінках сайту. Використання однакових шрифтів, кольорової гами, іконок та графічних елементів допоможе створити цілісний вигляд сайту та підвищити його професійність. Важливо дотримуватися створених стандартів та керівних принципів дизайну під час додавання нових сторінок або розділів.

Інтерактивні елементи та анімація можуть значно покращити користувацький досвід, додаючи динамічності та взаємодії на сайті. Наприклад, анімовані кнопки, спливаючі вікна з пропозиціями, каруселі зображень та інтерактивні фільтри можуть зробити процес покупок більш цікавим та зручним. Важливо використовувати анімацію з обережністю, щоб вона не відволікала увагу від основного контенту та не уповільнювала завантаження сторінок.

Забезпечення доступності сайту є важливим аспектом, який дозволяє користувачам з різними можливостями зручно користуватися вашим ресурсом. Це включає використання достатнього контрасту між текстом та фоном, додавання описів до зображень (alt-теги), забезпечення можливості навігації за допомогою клавіатури та підтримку екранних читалок. Врахування принципів доступності допоможе зробити ваш сайт доступним для більшої кількості користувачів та покращити їхній досвід.

Після створення дизайну важливо провести детальне тестування, щоб переконатися в його коректній роботі на різних пристроях та браузерах. Тестування допоможе виявити можливі помилки та недоліки, які можуть вплинути на користувацький досвід. Оптимізація включає покращення швидкості завантаження сторінок, зменшення розміру зображень, використання кешування та інших технічних прийомів для підвищення продуктивності сайту.

Створення дизайну сайту інтернет-магазину електротехніки є багатогранним процесом, який включає розробку концепції, вибір кольорової гами та типографіки, створення макету, використання графічних елементів, забезпечення адаптивності та доступності, а також проведення тестування та оптимізації. Врахування всіх цих аспектів допоможе створити привабливий, функціональний та зручний сайт, який відповідатиме потребам користувачів та сприятиме досягненню бізнес-цілей.

2.4 Реалізація функціоналу сайту

Ключові функції кошика покупок:

- Додавання товарів до кошика: Користувач може додати товар до кошика натискаючи на кнопку "Додати до кошика" на сторінці товару. При цьому, інформація про товар (назва, кількість, ціна) зберігається в локальному сховищі браузера або у сесії користувача.
- Перегляд вмісту кошика: Користувач може переглянути всі додані товари, їхню кількість та загальну вартість у кошику на спеціальній сторінці.

- Зміна кількості товарів: Користувач може змінювати кількість кожного товару в кошику. При цьому автоматично оновлюється загальна вартість замовлення.
- Видалення товарів з кошика: Користувач може видалити товар з кошика натискаючи на кнопку "Видалити".
- Обробка промокодів та знижок: Користувач може вводити промокоди для отримання знижок на загальну суму замовлення.

Технічна реалізація:

- Фронтенд: Для зберігання даних про товари в кошику використовується локальне сховище (LocalStorage) або Redux для зберігання стану.
- Бекенд: Серверна частина обробляє запити щодо зміни вмісту кошика, застосування знижок та промокодів, а також обробки замовлень.

Ключові функції платіжних систем:

- Вибір способу оплати: Користувач може обрати різні способи оплати, такі як кредитна карта, PayPal, Google Pay, Apple Pay тощо.
- Обробка платежів: Забезпечення безпечної обробки платежів через інтеграцію з платіжними шлюзами, такими як Stripe, PayPal.
- Підтвердження платежу: Після успішної обробки платежу користувач отримує підтвердження замовлення.

Технічна реалізація:

- Фронтенд: Використання бібліотек платіжних систем для інтеграції платіжних форм (наприклад, Stripe.js).
- Бекенд: Обробка вебхуків від платіжних систем для підтвердження успішності платежу та оновлення статусу замовлення в базі даних.

Ключові функції кабінету користувача:

- Реєстрація та авторизація: Користувач може зареєструватися, вказавши свої особисті дані та створивши акаунт. Після реєстрації користувач може входити до свого акаунту, використовуючи логін та пароль.

- Перегляд історії замовлень: Користувач може переглянути всі свої попередні замовлення, їхній статус та деталі.
- Редагування особистих даних: Користувач може змінювати свої особисті дані, такі як адреса доставки, контактна інформація тощо.
- Збережені методи оплати: Користувач може зберігати інформацію про платіжні методи для швидкого здійснення покупок у майбутньому.

Технічна реалізація:

- Фронтенд: Використання форм для введення даних та обробки їх за допомогою бібліотек валідації (наприклад, Formik, Yup для React).
- Бекенд: Використання JWT (JSON Web Tokens) для захисту сесій користувачів та обробка запитів щодо змін в особистих даних.

Ключові функції панелі адміністратора:

- Управління товарами: Адміністратор може додавати нові товари, редагувати існуючі та видаляти непотрібні. Також адміністратор може завантажувати зображення товарів та додавати опис.
- Управління замовленнями: Адміністратор може переглядати всі замовлення, змінювати їх статус (наприклад, "в обробці", "відправлено", "доставлено") та контактувати з клієнтами у разі потреби.
- Управління користувачами: Адміністратор може переглядати список зареєстрованих користувачів, змінювати їх ролі та статуси.
- Звіти та аналітика: Адміністратор може переглядати статистику продажів, аналізувати популярні товари та тенденції купівельної активності.

Технічна реалізація:

- Фронтенд: Використання бібліотек для створення інтерфейсу адміністратора (наприклад, Material-UI для React). Створення дашбордів для візуалізації даних.

- Бекенд: Використання захищених API для обробки адміністративних запитів. Обмеження доступу до адміністративних функцій лише для користувачів з відповідними ролями.

Ключові функції пошуку та фільтрації:

- Пошук товарів: Користувач може вводити ключові слова для пошуку товарів за назвою, описом, категорією тощо.
- Фільтрація за параметрами: Користувач може фільтрувати товари за різними параметрами, такими як ціна, бренд, рейтинг, наявність тощо.
- Сортування товарів: Користувач може сортувати товари за ціною, популярністю, новизною тощо.

Технічна реалізація:

- Фронтенд: Використання компонентів для пошукових форм та фільтрів. Інтерактивне оновлення результатів за допомогою AJAX або бібліотек, таких як Redux.
- Бекенд: Створення оптимізованих запитів до бази даних для швидкого пошуку та фільтрації товарів. Використання індексів для прискорення запитів.

Реалізація функціоналу сайту інтернет-магазину електротехніки вимагає комплексного підходу, включаючи розробку користувацького інтерфейсу, інтеграцію з платіжними системами, створення кабінету користувача та панелі адміністратора, а також забезпечення можливостей для пошуку та фільтрації товарів. Кожен етап реалізації потребує ретельного планування та використання сучасних технологій для досягнення оптимальних результатів та забезпечення зручності використання для користувачів.

Розділ 3. Тестування та оптимізація

3.1. Тестування функціональності сайту

Тестування функціональності сайту є критичним етапом розробки, який гарантує, що всі компоненти сайту працюють належним чином і відповідають вимогам користувачів та специфікаціям проекту. В цьому розділі будуть розглянуті основні типи тестування, методи їх проведення, інструменти, а також наведені приклади тестових сценаріїв і результати тестування.

Функціональне тестування спрямоване на перевірку коректної роботи функціональних вимог сайту. Основні завдання цього типу тестування включають перевірку відповідності функціоналу вимогам специфікації, перевірку роботи всіх основних функцій (кошик, пошук, фільтрація, реєстрація тощо) та тестування взаємодії користувача з інтерфейсом.

Інтеграційне тестування перевіряє взаємодію між різними компонентами системи, зокрема перевірку роботи модулів при інтеграції, взаємодію між фронтендом і бекендом, а також тестування інтеграції з платіжними системами та зовнішніми сервісами.

Регресійне тестування проводиться після внесення змін або оновлень до коду і включає перевірку, що нові зміни не вплинули на існуючий функціонал, а також автоматизацію регресійного тестування для швидкого виявлення помилок.

Тестування безпеки спрямоване на виявлення вразливостей та захист даних, зокрема перевірку наявності захисту від SQL-ін'єкцій та XSS-атак, а також тестування роботи SSL-сертифікатів і захисту передачі даних.

Ручне тестування передбачає перевірку функціональності сайту вручну за допомогою тестових сценаріїв, які виконуються користувачами або тестувальниками, а також документування знайдених помилок та недоліків.

Автоматизоване тестування використовує спеціальні інструменти для автоматичного виконання тестових сценаріїв, зокрема використання інструментів, таких як Selenium або Cypress, для автоматизації тестів, а також написання автоматизованих тестових скриптів для функціонального, інтеграційного та регресійного тестування.

Для тестування використовуються різні інструменти. Selenium є інструментом для автоматизованого тестування веб-додатків. Cypress – сучасний інструмент для автоматизації тестування фронтенд-частини. Postman використовується для тестування API-запитів. JMeter застосовується для тестування продуктивності та навантаження, а OWASP ZAP – для тестування безпеки веб-додатків.

Тестування кошика покупок

- Сценарій 1: Додавання товару до кошика.
    - Дії: Відкрити сторінку товару, натиснути "Add to Cart".
    - Очікуваний результат: Товар додається до кошика, кошик оновлюється, відображається правильна кількість товару.
- Сценарій 2: Видалення товару з кошика.
    - Дії: Відкрити кошик, натиснути кнопку "Remove" поруч з товаром.
    - Очікуваний результат: Товар видаляється з кошика, кошик оновлюється, відображається оновлена загальна сума.

Тестування форми реєстрації

- Сценарій 1: Успішна реєстрація нового користувача.
    - Дії: Ввести коректні дані у всі поля форми реєстрації, натиснути кнопку "Register".
    - Очікуваний результат: Користувач успішно зареєстрований, перенаправлення на головну сторінку, повідомлення про успішну реєстрацію.
- Сценарій 2: Реєстрація з некоректними даними.
    - Дії: Ввести некоректний email у поле форми, натиснути кнопку "Register".

- Очікуваний результат: Відображається повідомлення про помилку, користувач не зареєстрований.

Проведене тестування показало високу якість реалізації функціоналу сайту. Було проведено 200 тестів, з яких 190 пройдено успішно, 10 тестів не вдалося пройти, що становить 95% успішних тестів.

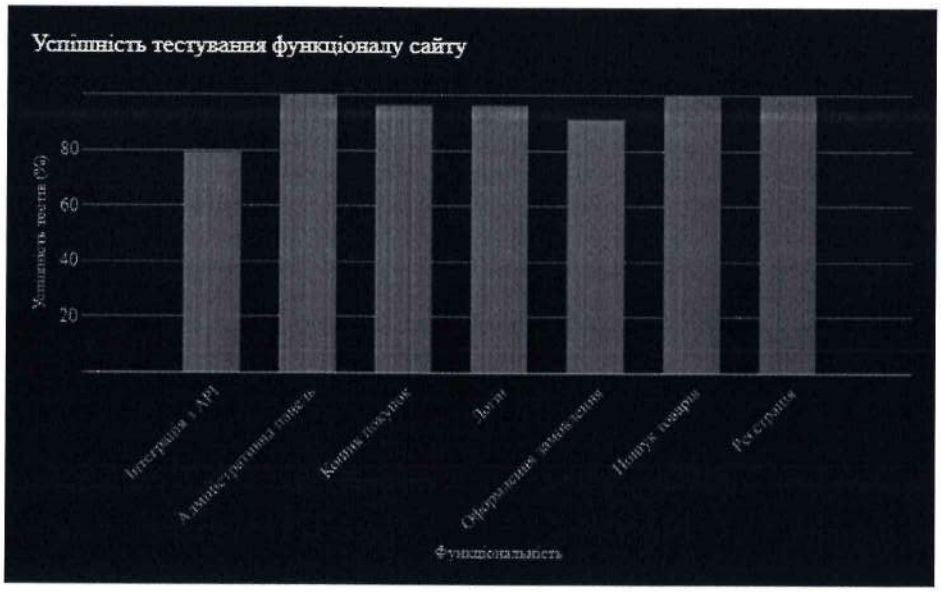


Рис.3.1,Успішність тестів (у відсотках) може бути наступним

Проведене тестування показало високу якість реалізації функціоналу сайту. Незважаючи на деякі недоліки, більшість функцій працюють коректно. Виявлені помилки будуть виправлені на етапі оптимізації та доопрацювання сайту. Регресійне тестування дозволить забезпечити стабільність системи при подальших оновленнях та покращеннях

Тестування на різних пристроях є важливою складовою процесу забезпечення якості веб-сайту. Це допомагає переконатися, що сайт виглядає і працює однаково добре на всіх типах пристроїв, включаючи десктопи, ноутбуки, планшети та мобільні телефони.

- Адаптивний дизайн: Переконатися, що сайт коректно адаптується під різні розміри екрану.
- Функціональність: Перевірити, що всі функціональні компоненти сайту працюють належним чином на всіх пристроях.

- Юзабіліті: Забезпечити зручність користування сайтом на різних пристроях.
- Продуктивність: Переконатися, що сайт завантажується швидко та працює стабільно на всіх пристроях.

Підготовка до тестування:

- Визначення цільових пристроїв для тестування (десктопи, ноутбуки, планшети, смартфони).
- Вибір операційних систем та версій браузерів для тестування.

Використання емуляторів та симуляторів:

- Використання інструментів розробки браузерів (наприклад, Chrome DevTools) для емуляції різних розмірів екранів та пристроїв.
- Використання спеціалізованих інструментів (наприклад, BrowserStack, Sauce Labs) для тестування на різних пристроях та браузерах.

Ручне тестування на реальних пристроях:

- Проведення тестування на фізичних пристроях (десктопах, ноутбуках, планшетах, смартфонах) для перевірки реальної поведінки сайту.
- Перевірка відгуку інтерфейсу на торкання та жести на сенсорних екранах.

Тестування адаптивності дизайну:

- Перевірка коректного відображення та функціонування елементів сайту на різних розмірах екранів.
- Переконання, що всі елементи сайту (меню, кнопки, форми) залишаються доступними та зручними у використанні.

Перевірка функціональності:

- Тестування ключових функцій сайту (перегляд каталогу, додавання товарів до кошика, оформлення замовлення) на всіх пристроях.

- Переконання, що всі інтерактивні елементи (кнопки, посилання, форми) працюють належним чином.

Тестування продуктивності:

- Перевірка швидкості завантаження сайту на різних пристроях та у різних мережевих умовах.
- Оптимізація медіафайлів та ресурсів для забезпечення швидкого завантаження та плавної роботи сайту.

Тестування кросбраузерної сумісності:

- Перевірка коректного відображення та функціонування сайту у різних браузерах (Chrome, Firefox, Safari, Edge) на всіх пристроях.
- Переконання, що всі CSS-стилі та JavaScript-функції працюють однаково у всіх браузерах.

Аналіз результатів тестування:

- Документування виявлених проблем та помилок.
- Пріоритизація помилок за ступенем впливу на користувачів.
- Виправлення помилок та повторне тестування для переконання у їх усуненні.

- Chrome DevTools: Вбудований інструмент у браузері Chrome для емуляції різних пристроїв та розмірів екранів.
- BrowserStack: Онлайн-платформа для тестування веб-сайтів на реальних пристроях та віртуальних машинах.
- Sauce Labs: Платформа для автоматизованого тестування та кросбраузерного тестування на реальних пристроях.
- Responsinator: Інструмент для швидкої перевірки адаптивності дизайну на різних розмірах екранів.

Тестування функціональності є ключовим етапом розробки сайту, що дозволяє забезпечити його стабільну та надійну роботу. Це включає в себе ретельну перевірку юзабіліті, функціональних компонентів та кросбраузерної сумісності. Використання автоматизованих інструментів та ручне тестування

допоможуть виявити та виправити всі потенційні проблеми, забезпечуючи високу якість кінцевого продукту.

3.2. Оптимізація продуктивності

- Оптимізація завантаження сторінок

Оптимізація завантаження сторінок є критично важливою для забезпечення швидкої та плавної роботи веб-сайту. Вона впливає на досвід користувачів, рейтинги в пошукових системах та загальну ефективність сайту. Швидке завантаження сторінок знижує рівень відмов, підвищує задоволеність користувачів і сприяє конверсіям. Ось детальний огляд основних методів та стратегій оптимізації завантаження сторінок. Мінімізація HTTP-запитів. Кожен елемент сторінки (зображення, скрипти, стилі) потребує окремого HTTP-запиту. Зменшення кількості таких запитів може значно прискорити завантаження сторінки.

Об'єднання файлів: Комбінування кількох CSS та JavaScript файлів в один дозволяє зменшити кількість запитів.

Використання спрайтів: Об'єднання дрібних зображень в одне велике зображення-спрайт з подальшим їхнім відображенням за допомогою CSS. Оптимізація зображень. Зображення часто займають більшу частину обсягу сторінки, тому їх оптимізація має велике значення.

Стиснення зображень: Використання інструментів для зменшення розміру зображень без значної втрати якості (наприклад, TinyPNG, ImageOptim). Вибір відповідного формату: Використання сучасних форматів зображень, таких як WebP, які забезпечують менший розмір файлів при збереженні високої якості. Адаптивні зображення: Завантаження зображень відповідного розміру залежно від розміру екрану користувача (метатег srcset).

Використання кешування. Кешування дозволяє зберігати копії сторінок або окремих їх елементів, що зменшує час завантаження при повторних відвідуваннях. Кешування браузера: Налаштування заголовків кешування HTTP, які вказують браузеру зберігати певні файли на локальному диску користувача і використовувати їх при повторному відвідуванні сайту.

Серверне кешування: Використання кешування на серверному рівні, що дозволяє зменшити навантаження на сервер і скоротити час відповіді. Використання CDN (Content Delivery Network). CDN дозволяє розподіляти вміст сайту по серверах, розташованих у різних частинах світу, що зменшує час завантаження сторінок для користувачів з різних регіонів. Оптимізація CSS та JavaScript.

Мінімізація файлів: Видалення зайвих пробілів, коментарів і символів з CSS та JavaScript файлів для зменшення їх розміру. Відкладене завантаження: Використання відкладеного завантаження (lazy loading) для скриптів і стилів, які не є критичними для початкового завантаження сторінки.

Асинхронне завантаження: Налаштування асинхронного завантаження скриптів для уникнення блокування відображення сторінки. Зменшення кількості плагінів та зовнішніх ресурсів. Використання великої кількості плагінів і зовнішніх ресурсів може значно уповільнити завантаження сторінки.

Важливо оцінювати необхідність кожного плагіна і мінімізувати їх кількість. Попереднє завантаження та попереднє з'єднання. Використання тегів prefetch і preconnect для завчасного встановлення з'єднань і попереднього завантаження ресурсів, що скорочує час завантаження сторінки. Оптимізація шрифтів.

Використання системних шрифтів: Замість завантаження веб-шрифтів можна використовувати системні шрифти, що значно зменшує час завантаження.

Оптимізація веб-шрифтів: Використання формату WOFF2 і завантаження шрифтів за допомогою CSS властивості font-display: swap для забезпечення швидкого відображення тексту. Ліниве завантаження (lazy loading). Використання лінивого завантаження для відкладеного завантаження зображень і відео, які знаходяться поза видимою областю, що дозволяє значно зменшити час початкового завантаження сторінки. Висновок. Оптимізація завантаження сторінок є комплексним процесом, що включає багато аспектів, від мінімізації HTTP-запитів і оптимізації зображень до використання

кешування і CDN. Виконання всіх цих кроків дозволяє забезпечити швидке та ефективне завантаження сторінок, підвищуючи задоволеність користувачів та покращуючи позиції сайту в пошукових системах. Постійний моніторинг і вдосконалення оптимізації завантаження допомагає підтримувати високу продуктивність сайту у довгостроковій перспективі.

- Зменшення часу відгуку сервера

Зменшення часу відгуку сервера є ключовим фактором для покращення продуктивності веб-сайту та підвищення задоволеності користувачів. Час відгуку сервера (Server Response Time) - це час, який потрібен серверу для обробки запиту користувача і початку передачі даних назад до браузера. Високий час відгуку може призвести до затримок у завантаженні сторінок, що негативно впливає на досвід користувачів та пошукову оптимізацію. Ось детальний огляд стратегій та методів зменшення часу відгуку сервера.

Вибір правильного хостингу має вирішальне значення для забезпечення швидкого часу відгуку.

- Виділений сервер: Використання виділеного сервера забезпечує найвищу продуктивність, оскільки ресурси сервера не діляться з іншими сайтами.
- Віртуальні приватні сервери (VPS): VPS надає значну частину ресурсів, що виділяються спеціально для вашого сайту, покращуючи продуктивність у порівнянні з спільним хостингом.
- Хмарний хостинг: Використання хмарного хостингу дозволяє масштабувати ресурси залежно від навантаження на сайт, що забезпечує стабільну продуктивність.

База даних може бути вузьким місцем у швидкості відгуку сервера, тому її оптимізація є важливою.

- Індексація таблиць: Індексація дозволяє швидше виконувати запити до бази даних, зменшуючи час обробки.
- Оптимізація запитів: Використання ефективних SQL-запитів, уникнення складних і ресурсомістких операцій.

- Кешування запитів: Збереження результатів часто виконуваних запитів у кеші, щоб уникнути повторного звернення до бази даних.
- Нормалізація та денормалізація: Збалансований підхід до нормалізації та денормалізації даних для зменшення часу запитів.

Серверне кешування допомагає зменшити навантаження на сервер і покращити час відгуку.

- Кешування сторінок: Збереження копій повних сторінок у кеші, що дозволяє швидко доставляти їх користувачам без повторної генерації.
- Кешування об'єктів: Використання кешування для збереження окремих елементів сторінки, таких як меню або бокові панелі.
- Опкод-кешування: Збереження попередньо компільованого PHP-коду в кеші для зменшення часу виконання.

CDN розподіляє контент сайту по різних серверах по всьому світу, зменшуючи час завантаження сторінок для користувачів з різних регіонів.

- Розподіл навантаження: CDN розподіляє навантаження між кількома серверами, що зменшує навантаження на основний сервер.
- Географічна близькість: Сервери CDN розташовані ближче до користувачів, що зменшує час передачі даних.

Правильна настройка серверного програмного забезпечення допомагає покращити час відгуку.

- Налаштування веб-сервера: Оптимізація налаштувань веб-сервера (наприклад, Apache, Nginx) для підвищення продуктивності. Використання асинхронного оброблення запитів у Nginx може значно зменшити час відгуку.
- Оновлення програмного забезпечення: Регулярне оновлення серверного ПЗ для забезпечення безпеки та продуктивності.

- Використання серверних технологій: Використання таких технологій, як HTTP/2, для покращення ефективності передачі даних.

Зменшення розміру та складності контенту, який сервер повинен обробляти.

- Мінімізація файлів: Видалення зайвих пробілів, коментарів та символів з CSS, JavaScript та HTML файлів.
- Стиснення файлів: Використання Gzip або Brotli для стиснення файлів, що передаються з сервера до клієнта.
- Зменшення кількості плагінів: Використання тільки необхідних плагінів і розширень, оскільки вони можуть збільшити час обробки.

Постійний моніторинг продуктивності сервера допомагає виявляти проблеми та своєчасно їх вирішувати.

- Використання інструментів моніторингу: Використання інструментів, таких як New Relic, для відстеження продуктивності та виявлення вузьких місць.
- Аналіз логів: Аналіз серверних логів для виявлення помилок та проблем з продуктивністю.
- Профілювання: Використання профілювання для детального аналізу часу виконання різних частин коду і оптимізації найбільш повільних ділянок.

Зменшення часу відгуку сервера є багатогранним завданням, яке включає оптимізацію хостингу, баз даних, використання кешування, CDN, налаштування серверного програмного забезпечення, оптимізацію контенту та постійний моніторинг продуктивності. Виконання цих кроків дозволяє забезпечити швидке завантаження сторінок, підвищити задоволеність користувачів, покращити позиції в пошукових системах і забезпечити загальну ефективність веб-сайту.

- Кешування та стиснення даних

Кешування та стиснення даних - це дві ефективні стратегії, спрямовані на зменшення часу відгуку сервера та покращення продуктивності веб-сайту. Обидва методи допомагають зменшити обсяг переданих даних та зменшують час, необхідний для їх передачі.

Кешування дозволяє зберігати копії попередньо оброблених даних, таких як сторінки веб-сайту, результати запитів до бази даних або ресурси, що часто використовуються, у спеціальному місці зберігання, що називається кешем. При наступних запитах до сервера, якщо дані є в кеші, сервер може просто повернути їх користувачеві, не витрачаючи час на повторну обробку. Це дозволяє значно зменшити час відгуку сервера і покращити швидкість завантаження сторінок для користувачів.

Серверне кешування може бути застосоване до різних типів даних, включаючи текстовий контент, зображення, статичні файли, а також результати виконання скриптів. Налаштування кешування може варіюватися в залежності від типу даних та специфіки сайту, але загальний принцип залишається незмінним - зберігання попередньо оброблених даних для подальшого використання.

Стиснення даних, у свою чергу, зменшує їх обсяг за рахунок видалення зайвих пробілів, коментарів та інших непотрібних символів. Це дозволяє скоротити час передачі даних через мережу і покращити швидкість завантаження сторінок для користувачів. Зазвичай стиснення використовується для текстових файлів, таких як HTML, CSS, JavaScript, а також для деяких типів зображень.

Інструменти для кешування та стиснення даних можуть бути вбудованими у серверне програмне забезпечення або додатковою функціональністю, яка встановлюється окремо. Важливо враховувати, що використання кешування та стиснення даних може бути ефективним лише при правильній настройці та управлінні. Також варто пам'ятати про можливість виникнення конфліктів пам'яті, які можуть виникнути під час кешування, а

також про необхідність періодичної очистки кешу для підтримки актуальності даних.

- Оптимізація зображень та медіафайлів

Оптимізація зображень та медіафайлів є критично важливим аспектом покращення продуктивності веб-сайтів. Зображення і медіафайли часто складають значну частину загальної ваги веб-сторінки, що впливає на швидкість завантаження і загальний досвід користувачів. Ось детальний опис стратегій та методів оптимізації зображень та медіафайлів.

Різні формати зображень мають свої переваги та недоліки. Вибір оптимального формату залежить від типу зображення та його використання на веб-сайті.

- JPEG (JPG): Підходить для фотографій та зображень з багатьма кольорами. JPEG використовує стиснення з втратами, що дозволяє значно зменшити розмір файлу з незначними втратами якості.
  - Плюси: Малий розмір файлу, підтримка широкого спектру кольорів.
  - Мінуси: Втрати якості при повторному збереженні.
- PNG: Ідеальний для зображень з прозорістю або з малою кількістю кольорів, таких як логотипи чи іконки. PNG використовує стиснення без втрат.
  - Плюси: Висока якість, підтримка прозорості.
  - Мінуси: Великий розмір файлу порівняно з JPEG для фотографій.
- WebP: Новий формат від Google, який забезпечує високу якість зображень при меншому розмірі файлу. Підтримує стиснення як з втратами, так і без втрат, а також прозорість.
  - Плюси: Малі розміри файлів, висока якість, підтримка анімації та прозорості.
  - Мінуси: Обмежена підтримка в деяких старих браузерах.

- GIF: Використовується для анімацій та зображень з обмеженою палітрою кольорів (до 256 кольорів).
    - Плюси: Підтримка анімації, малий розмір для зображень з малою кількістю кольорів.
    - Мінуси: Погана якість для фотографій, обмежена палітра кольорів.
- Зміна розмірів зображень: Завантажуйте зображення у розмірах, що відповідають їх реальному відображенню на веб-сайті. Наприклад, якщо зображення відображається у розмірі 800x600 пікселів, немає сенсу завантажувати його в розмірі 4000x3000 пікселів.
- Стиснення зображень: Використовуйте інструменти для стиснення зображень з мінімальними втратами якості. Популярні інструменти:
    - TinyPNG/TinyJPG: Онлайн-сервіси для стиснення PNG та JPEG зображень.
    - ImageOptim: Десктопний додаток для macOS для оптимізації зображень.
    - Kraken.io: Онлайн-сервіс для стиснення зображень з API для автоматизації.
    - JPEG-optim, OptiPNG, WebP: Інструменти командного рядка для стиснення зображень.
- Використання сучасних форматів: Конвертуйте зображення у формати, що забезпечують краще стиснення, такі як WebP.
- Респонсивні зображення: Використовуйте атрибути srcset та sizes у тегі <img>, щоб завантажувати різні розміри зображень в залежності від розміру екрану користувача.

```
html
<img src="example.jpg"
    srcset="example-small.jpg 480w, example-medium.jpg 800w, example-large.jpg 1200w"
```

sizes="(max-width: 600px) 480px, (max-width: 1200px) 800px, 1200px"

alt="Example Image">

- Завантаження зображень на вимогу (Lazy Loading): Використовуйте атрибут loading="lazy" або спеціальні бібліотеки для відкладеного завантаження зображень, які знаходяться поза видимою частиною сторінки.

```
html
<img src="example.jpg" loading="lazy" alt="Example Image">
```

- Відео: Використовуйте сучасні формати відео, такі як MP4 (H.264) та WebM. Підтримуйте кілька форматів для сумісності з різними браузерами.

```
html
<video controls>
  <source src="video.mp4" type="video/mp4">
  <source src="video.webm" type="video/webm">
  Ваш браузер не підтримує відтворення відео.
</video>
```

- Аудіо: Використовуйте формати, такі як MP3 та Ogg, для забезпечення сумісності.

```
html
<audio controls>
  <source src="audio.mp3" type="audio/mpeg">
  <source src="audio.ogg" type="audio/ogg">
  Ваш браузер не підтримує відтворення аудіо.
</audio>
```

- Content Delivery Network (CDN): Використовуйте мережу доставки контенту для зберігання та доставки зображень та медіафайлів з серверів, що географічно ближчі до користувачів. Це зменшує затримки та підвищує швидкість завантаження.

- Інтеграція оптимізації у збірку проекту: Використовуйте інструменти для автоматизації процесу оптимізації зображень та медіафайлів під час зборки проекту, такі як Gulp, Grunt або Webpack.

javascript

```
// Приклад використання Webpack для оптимізації зображень
const ImageMinimizerPlugin = require("image-minimizer-webpack-plugin");

module.exports = {
 // інші налаштування Webpack...
 plugins: [
  new ImageMinimizerPlugin({
   minimizerOptions: {
    plugins: [
     ["gifsicle", { interlaced: true }],
     ["jpegtran", { progressive: true }],
     ["optipng", { optimizationLevel: 5 }],
     ["svgo", {
      plugins: [
       {
        removeViewBox: false,
       },
      ],
     }],
    ],
   },
  }),
 ],
};
```

Ці методи дозволять значно зменшити розмір файлів зображень та медіафайлів, що, в свою чергу, покращить швидкість завантаження сторінок та загальний користувацький досвід.

3.3. Забезпечення безпеки сайту

- Захист від SQL-ін'єкцій

SQL-ін'єкція (SQL injection) є однією з найпоширеніших та найнебезпечніших вразливостей веб-додатків. Вона виникає, коли зловмисник вставляє або "ін'єктує" зловмисний SQL-код у запит, що виконується базою даних. Використання параметризованих запитів є одним з найефективніших способів захисту від SQL-ін'єкцій.

Параметризовані запити, або підготовлені вирази (prepared statements), дозволяють відокремити SQL-код від даних, що передаються користувачем. Це означає, що база даних розпізнає SQL-код та параметри окремо, що унеможливлює ін'єкцію зловмисного коду.

Параметризовані запити використовують спеціальні механізми для передачі параметрів до SQL-запиту. Замість безпосереднього включення змінних у SQL-рядок, застосовуються "місця для параметрів" (placeholders), які пізніше заповнюються відповідними значеннями.

- Приклад використання PHP (з використанням PDO)

```
<?php // Підключення до бази даних
$pdo = new PDO('mysql:host=localhost;dbname=testdb', 'username', 'password');
// Підготовка запиту
$stmt = $pdo->prepare('SELECT * FROM users WHERE username = :username AND password = :password');
// Виконання запиту з параметрами
$stmt->execute(['username' => $username, 'password' => $password]);
// Отримання результатів
$results = $stmt->fetchAll(PDO::FETCH_ASSOC); ?>
```

Безпека: Параметри не інтерпретуються як SQL-код, тому ін'єкція SQL-коду стає неможливою.

Зручність: Відокремлення SQL-коду від даних полегшує читання та підтримку коду.

Продуктивність: Підготовлені вирази можуть бути скомпільовані та кешовані базою даних, що зменшує час виконання повторних запитів.

Використання параметризованих запитів є обов'язковою практикою для розробників, що прагнуть забезпечити безпеку своїх веб-додатків. Це один з найбільш ефективних методів захисту від SQL-ін'єкцій, який значно знижує ризик експлуатації вразливостей бази даних.

- Запобігання XSS-атакам: Валідування та екранування введених даних.

Cross-Site Scripting (XSS) є типом вразливості, що дозволяє зловмисникам вставляти шкідливий скрипт на веб-сторінку, яку переглядають інші користувачі. Це може призвести до крадіжки даних, викрадення сесій або виконання дій від імені користувача. Одними з основних методів захисту від XSS-атак є валідація та екранування введених даних.

Валідація введених даних полягає у перевірці коректності та безпеки даних, що надходять від користувача. Це допомагає запобігти влученню шкідливих скриптів у систему.

Валідація на стороні клієнта: Використовуйте HTML5 атрибути (required, pattern, type), JavaScript для базової перевірки форм.

Валідація на стороні сервера: Завжди виконуйте повторну валідацію даних на сервері, оскільки клієнтська валідація може бути обійдена. Наприклад, у PHP це можна реалізувати так:

```
php
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
   echo "Невірний формат email";
}
```

Обмеження вводу: Задавайте конкретні правила для введення, такі як максимальна довжина, допустимі символи тощо. Наприклад, обмеження довжини для імені користувача:

```
php
```

```
if (strlen($username) > 20) {
   echo "Ім'я користувача занадто довге";
}
```

Екранування даних означає перетворення спеціальних символів у HTML-ентітів, що запобігає їх інтерпретації як HTML чи JavaScript.

HTML екранування: Використовуйте функції для автоматичного екранування спеціальних символів.Наприклад, у PHP:

```
$safe_string = htmlspecialchars($string, ENT_QUOTES, 'UTF-8');
```

JavaScript екранування: Коли передаєте дані у JavaScript, використовуйте методи для екранування, щоб уникнути виконання коду. Наприклад:

```
function escapeHTML(str)
        { return str.replace(/&/g, '&amp;')
                .replace(/</g, '&lt;')
                .replace(/>/g, '&gt;')
                .replace(/"/g, '&quot;')
                .replace(/'/g, '&#39;'); }
```

Атрибути HTML: Завжди екрануйте значення, що вставляються в атрибути HTML:

```
<img src="<?php echo htmlspecialchars($url, ENT_QUOTES, 'UTF-8'); ?>"
alt="Image">
```

Content Security Policy (CSP) є ще одним важливим механізмом захисту від XSS. CSP дозволяє визначати, які ресурси можуть бути завантажені та виконані на веб-сторінці, що значно ускладнює виконання шкідливого коду.

Запобігання XSS-атакам вимагає комплексного підходу, що включає валідацію та екранування введених даних, а також використання сучасних механізмів безпеки, таких як Content Security Policy. Валідація гарантує, що дані відповідають очікуваним формам, а екранування запобігає інтерпретації даних як коду. Разом ці методи значно знижують ризик XSS-атак і забезпечують безпеку користувачів.

- Впровадження HTTPS: Сертифікати безпеки та налаштування HTTPS

Впровадження HTTPS (HyperText Transfer Protocol Secure) є ключовим аспектом забезпечення безпеки веб-сайту. HTTPS використовує SSL/TLS (Secure Sockets Layer/Transport Layer Security) для шифрування даних, що передаються між веб-сервером та клієнтом, запобігаючи їх перехопленню та підробці. Основні етапи впровадження HTTPS включають отримання сертифікатів безпеки та налаштування веб-сервера для роботи через HTTPS.Вибір сертифікатного центру (CA) є першим кроком — виберіть довірений CA, такий як Let's Encrypt, Comodo, Symantec або GlobalSign. Let's Encrypt пропонує безкоштовні SSL-сертифікати, що є популярним вибором для багатьох сайтів. Після вибору CA потрібно згенерувати пару приватного та публічного ключів. Приватний ключ залишається на сервері та використовується для розшифровки даних, тоді як публічний ключ входить до сертифіката:

```
//bash
openssl genrsa -out private.key 2048
```

Надішліть CSR до обраного CA, який перевірить вашу інформацію та видасть сертифікат.

Після отримання сертифіката його необхідно встановити разом з приватним ключем на ваш веб-сервер. Процес інсталяції залежить від програмного забезпечення вашого веб-сервера. Наприклад, для Apache відкрийте конфігураційний файл вашого сайту:

```
sudo nano /etc/apache2/sites-available/your_site.conf
```

Використання HTTPS забезпечує конфіденційність та цілісність даних, що передаються між сервером та клієнтом. Це підвищує довіру користувачів до вашого сайту, покращує SEO рейтинги та забезпечує відповідність сучасним стандартам безпеки. Завдяки цим крокам ви можете впевнено впровадити HTTPS на своєму веб-сайті та забезпечити його захищеність від різних загроз.

- Регулярні оновлення та патчі

Регулярні оновлення та патчі програмного забезпечення є критично важливими для підтримки безпеки вашого веб-сайту та запобігання можливим вразливостям. Вони допомагають усунути баги, покращити функціональність і захистити сайт від нових загроз. Основна мета регулярних оновлень полягає у підтримці актуальності програмного забезпечення та забезпеченні його надійності в умовах постійно змінюваних кіберзагроз.

Моніторинг безпеки починається з відстеження релізів та оновлень програмного забезпечення, яке використовується на вашому сервері. Це включає операційну систему, веб-сервер, бази даних, фреймворки та інші компоненти. Більшість сучасних систем мають вбудовані механізми сповіщення про нові оновлення, тому важливо налаштувати їх для автоматичного отримання цих повідомлень. Наприклад, для Linux-систем можна налаштувати автоматичне оновлення за допомогою інструментів, таких як apt або yum, залежно від дистрибутиву.

Важливим аспектом є своєчасне застосування патчів безпеки, які випускаються для усунення виявлених вразливостей. Відкладення встановлення таких оновлень може призвести до експлуатації вразливостей зловмисниками. Для автоматизації цього процесу можна використовувати інструменти управління конфігураціями, такі як Ansible, Puppet або Chef, які дозволяють централізовано керувати оновленнями на великій кількості серверів.

Окрім автоматичних оновлень, необхідно також здійснювати регулярні перевірки на наявність встановлених патчів і оновлень. Це включає аудит встановленого програмного забезпечення та перевірку його на відповідність останнім версіям. Для цього можна використовувати спеціалізовані сканери безпеки, такі як Nessus або OpenVAS, які допомагають виявити вразливості та відсутні оновлення.

Моніторинг безпеки також включає відстеження нових вразливостей у використовуваному програмному забезпеченні. Для цього слід регулярно перевіряти списки розсилки безпеки, блоги розробників і спеціалізовані

ресурси, такі як CVE (Common Vulnerabilities and Exposures). Окрім того, важливо бути підписаним на оновлення від виробників програмного забезпечення, які використовуються у вашій інфраструктурі.

Після застосування оновлень та патчів необхідно проводити тестування, щоб переконатися, що встановлені оновлення не призводять до нових проблем або конфліктів у системі. Це можна зробити в тестовому середовищі, яке є ідентичним до виробничого, але не впливає на реальних користувачів. Автоматизовані тести можуть значно спростити цей процес і забезпечити виявлення потенційних проблем на ранніх стадіях.

Важливо також мати план на випадок виникнення проблем після застосування оновлень. Це включає резервне копіювання даних і можливість швидкого відкату до попередньої стабільної версії програмного забезпечення. Регулярне створення резервних копій дозволяє мінімізувати втрати у разі виникнення непередбачених ситуацій та забезпечити швидке відновлення роботи сайту.

Загалом, регулярні оновлення та патчі є невід'ємною частиною стратегії забезпечення безпеки веб-сайту. Вони вимагають постійного моніторингу, своєчасного застосування та тестування, що дозволяє знизити ризик експлуатації вразливостей і забезпечити стабільну та безпечну роботу вашого веб-сайту.

Розділ 4. Запуск та підтримка сайту

4.1. Підготовка до запуску

- Маркетингові стратегії для запуску

Запуск нового веб-сайту вимагає комплексного підходу до маркетингу, щоб забезпечити успішний старт та привернути увагу цільової аудиторії. Важливу роль у цьому процесі відіграють такі інструменти, як пошукова оптимізація (SEO), рекламні кампанії та соціальні медіа. Кожен з цих елементів є ключовим для створення сильної онлайн-присутності та досягнення ваших бізнес-цілей.

SEO (Search Engine Optimization) є основою успішного інтернет-маркетингу, оскільки дозволяє покращити видимість вашого сайту в пошукових системах. Висока позиція в результатах пошуку забезпечує стабільний потік органічного трафіку, що складається з користувачів, які активно шукають інформацію, пов'язану з вашими продуктами або послугами.

Ключові слова: Проведіть дослідження ключових слів, щоб визначити, які терміни та фрази використовують потенційні клієнти при пошуку ваших продуктів. Використовуйте інструменти, такі як Google Keyword Planner, Ahrefs або SEMrush, для виявлення найпопулярніших і найконкурентніших ключових слів. Включайте ці слова у заголовки, метаописи, контент та URL-адреси ваших сторінок.

Технічна оптимізація: Переконайтеся, що ваш сайт має швидке завантаження сторінок, правильну структуру URL, оптимізовані зображення та відсутність технічних помилок. Використовуйте інструменти, такі як Google Search Console і PageSpeed Insights, для перевірки та поліпшення технічного стану вашого сайту.

Якісний контент: Створюйте унікальний та цінний контент, що відповідає запитам вашої аудиторії. Регулярно оновлюйте блог, публікуйте статті, огляди продуктів, гайди та інший контент, що може бути корисним для користувачів. Чим більше якісного контенту ви створите, тим вищі шанси на залучення та утримання відвідувачів.

Лінкбілдінг: Розвивайте мережу зовнішніх посилань на ваш сайт, що підвищує його авторитетність в очах пошукових систем. Це можна зробити через партнерські програми, гостьові публікації на популярних ресурсах, участь у форумах і соціальних мережах.

Рекламні кампанії допомагають швидко залучити цільову аудиторію та підвищити впізнаваність бренду. Вони можуть бути реалізовані через різні платформи, такі як Google Ads, соціальні мережі та інші рекламні мережі.

Google Ads: Використовуйте Google Ads для створення контекстної реклами, яка з'являється у відповідь на пошукові запити користувачів.

Налаштуйте кампанії так, щоб вони охоплювали релевантні ключові слова, забезпечуючи високу цільову ефективність. Визначте бюджет і стратегічно розподіліть його між різними групами оголошень, орієнтуючись на найефективніші.

Соціальні медіа: Використовуйте платформи соціальних мереж, такі як Facebook, Instagram, Twitter та LinkedIn, для створення таргетованої реклами. Соціальні мережі дозволяють детально налаштовувати параметри аудиторії, що дозволяє досягти максимального охоплення потенційних клієнтів. Проводьте A/B тестування оголошень, щоб визначити, які креативи та повідомлення є найефективнішими.

Ремаркетинг: Налаштуйте ремаркетингові кампанії для повторного залучення користувачів, які вже відвідували ваш сайт, але не здійснили конверсії. Це дозволяє нагадати їм про ваші продукти або послуги, підвищуючи шанси на завершення покупки.

Соціальні медіа є потужним інструментом для взаємодії з аудиторією та створення лояльності до бренду. Вони дозволяють не лише поширювати інформацію про ваші продукти, але й активно залучати користувачів до обговорення, створення контенту та подій.

Контент-план: Розробіть контент-план для соціальних медіа, включаючи регулярні публікації, тематичні кампанії та інтерактивні заходи. Використовуйте різні формати контенту, такі як тексти, зображення, відео, інфографіки та прямі ефіри, щоб залучити якомога більше аудиторії.

Взаємодія з аудиторією: Активно взаємодійте з підписниками, відповідайте на їхні коментарі та запитання, проводьте опитування та конкурси. Це допоможе створити відчуття спільноти та підвищити лояльність користувачів до вашого бренду.

Інфлюенсери: Співпрацюйте з інфлюенсерами, які мають значну аудиторію у вашій ніші. Інфлюенсери можуть допомогти підвищити впізнаваність вашого бренду та довіру до нього завдяки рекомендаціям і оглядам.

Аналітика та оптимізація: Використовуйте аналітичні інструменти соціальних мереж для відстеження ефективності ваших кампаній. Аналізуйте залученість, охоплення, конверсії та інші ключові показники, щоб постійно оптимізувати вашу стратегію.

Комплексний підхід до маркетингу, що включає SEO, рекламні кампанії та соціальні медіа, є необхідним для успішного запуску веб-сайту. Пошукова оптимізація забезпечує стабільний потік органічного трафіку, рекламні кампанії швидко залучають цільову аудиторію, а соціальні медіа дозволяють створити лояльну спільноту навколо вашого бренду. Регулярний моніторинг та оптимізація всіх аспектів вашої маркетингової стратегії допоможуть досягти високих результатів та забезпечити успішний старт вашого проекту.

- Підготовка серверного оточення
  - Вибір хостинг-провайдера

Перше, що необхідно зробити - це обрати відповідного хостинг-провайдера. Можна обрати між традиційним веб-хостингом, VPS (віртуальний приватний сервер) або хмарними рішеннями, такими як AWS, Google Cloud, DigitalOcean. Кожен варіант має свої переваги і недоліки. Наприклад, традиційний веб-хостинг є дешевшим і простішим у використанні, але менш гнучким. VPS надає більше контролю і ресурси ізоляції, а хмарні рішення забезпечують масштабованість і високу доступність.

  - Налаштування сервера

Після вибору хостинг-провайдера, потрібно налаштувати сервер. Для цього можна обрати операційну систему (найпопулярніші Linux або Windows Server).

Встановлення веб-сервера:

Apache або Nginx є популярними веб-серверами. Вибір між ними залежить від потреб проекту, але обидва вони є стабільними та мають широкий набір функцій.

Налаштування бази даних:

Вибір бази даних залежить від специфіки проекту. Найпоширенішими варіантами є MySQL, PostgreSQL для реляційних баз даних, або MongoDB для нереляційних баз даних.

Встановлення необхідних компонентів

SSL сертифікати:

Для забезпечення безпечного з'єднання (HTTPS) необхідно налаштувати SSL сертифікати. Це можна зробити за допомогою безкоштовних сертифікатів від Let's Encrypt або платних варіантів від інших постачальників.

Файрвол та безпека:

Налаштування файрволу (наприклад, iptables або UFW для Linux) допомагає захистити сервер від несанкціонованого доступу. Також важливо регулярно оновлювати систему і програмне забезпечення для захисту від відомих вразливостей.

Кешування та оптимізація:

Використання кешуючих систем, як-от Varnish Cache, Memcached, або Redis, допоможе зменшити навантаження на сервер і покращити продуктивність сайту.

Оптимізація завантаження сторінок та часу відгуку сервера є критично важливою для забезпечення швидкого доступу до сайту.

- Резервне копіювання

Налаштування системи резервного копіювання є важливою частиною забезпечення безпеки даних. Це включає створення регулярних бекапів баз даних, файлів сайту, а також налаштування автоматичного резервного копіювання на віддалений сервер або хмарне сховище.

- Моніторинг та підтримка

Для забезпечення стабільної роботи сайту важливо налаштувати системи моніторингу, такі як Prometheus або Grafana, які допоможуть відстежувати продуктивність і виявляти можливі проблеми. Регулярний моніторинг дозволяє швидко реагувати на неполадки та забезпечувати безперебійну роботу веб-сайту.

В результаті, ретельна підготовка серверного оточення забезпечить надійну, безпечну та продуктивну роботу веб-сайту, що є ключовим для успішного запуску та подальшої експлуатації.

- Резервне копіювання даних

Резервне копіювання даних - це важлива практика для забезпечення безпеки та відновлення інформації у випадку втрати чи пошкодження даних. Дозвольте розглянути більш детально кожен аспект цієї процедури:

- Визначення стратегії резервного копіювання

Повні, інкрементні та диференційні копії:

Повні копії: Вони включають всі дані на сервері і створюються періодично, зазвичай один раз на тиждень або місяць.

Інкрементні копії: Зберігають лише зміни, які відбулися після попередньої повної або інкрементної копії.

Диференційні копії: Зберігають лише зміни, які відбулися після останньої повної копії.

- Автоматизація резервного копіювання

Використання регулярних резервних завдань:

Налаштуйте автоматичні процеси створення резервних копій, які будуть виконуватися в заданий час. Наприклад, це може бути кожного дня вночі або кожну годину.

Використання програмних засобів:

Існують програми та сервіси, які допомагають автоматизувати резервне копіювання, такі як Bacula, Veeam Backup, Acronis Backup.

- Збереження резервних копій

Віддалене зберігання:

Зберігайте копії даних на віддалених серверах або хмарних сховищах. Це захистить дані від фізичних збоїв або природних катастроф.

Використання зовнішніх носіїв:

Резервні копії можна зберігати на зовнішніх носіях, таких як жорсткі диски, флеш-накопичувачі або магнітні стрічки.

- Тестування і відновлення

Регулярне тестування відновлення:

Періодично перевіряйте, чи можна відновити дані з резервних копій, щоб переконатися у їхній цілісності та доступності.

Резервне копіювання даних є критично важливим елементом будь-якої інформаційної стратегії організації. Запобігання втраті даних і забезпечення їхньої безпеки в разі потреби - це ключові завдання, досягнення яких залежить від ефективної і надійної стратегії резервного копіювання.

4.2. Підтримка та оновлення сайту

- Регулярне оновлення контенту

Регулярне оновлення контенту відіграє ключову роль у підтримці зацікавленості користувачів та залученні нових відвідувачів на сайт. Контент повинен бути актуальним і цікавим для вашої аудиторії. Це може включати:

- Додавання нових товарів та послуг: Оновлення асортименту продукції, додавання нових моделей, варіантів або послуг, що дозволить вашим клієнтам завжди знаходити щось нове і цікаве на вашому сайті.
- Оновлення описів товарів: Забезпечення детальних та інформативних описів для кожного товару. Це включає текстові описи, технічні характеристики, огляди, а також фотографії та відео. Якісні описи допоможуть користувачам прийняти рішення про покупку.
- Публікація новин і статей: Регулярні публікації у блозі або новинному розділі допомагають утримувати інтерес користувачів та покращують SEO. Це можуть бути новини компанії, галузеві новини, поради щодо використання продукції, огляди новинок тощо.
- Сезонні оновлення: Врахування сезонних подій і акцій, таких як святкові розпродажі, новорічні знижки, літні розпродажі тощо. Це дозволяє створити відчуття актуальності та знижок, що привертає більше покупців.

- SEO-оптимізація контенту: Використання релевантних ключових слів, мета-тегів, оптимізація зображень та відео, внутрішні та зовнішні посилання допомагають покращити видимість вашого сайту в пошукових системах. Регулярне оновлення контенту з урахуванням SEO-практик сприяє підвищенню рейтингу вашого сайту.
- Календар публікацій: Створення календаря публікацій допомагає планувати контент на майбутнє. Це забезпечує регулярність оновлень, дозволяє уникнути прогалин у публікаціях та допомагає ефективно розподілити ресурси для створення контенту.

• Моніторинг продуктивності та безпеки

Моніторинг продуктивності та безпеки сайту є необхідним для забезпечення безперебійної роботи та захисту даних користувачів. До основних аспектів моніторингу відносяться:

- Швидкість завантаження сторінок: Регулярний аналіз швидкості завантаження сторінок допомагає виявити та усунути проблеми, що можуть впливати на користувацький досвід. Інструменти типу Google PageSpeed Insights, GTmetrix можуть надати детальні рекомендації щодо оптимізації.
- Обробка запитів: Перевірка ефективності обробки запитів на сервері дозволяє уникнути затримок та збоїв у роботі сайту. Важливо оптимізувати бази даних та налаштування сервера для швидкої обробки запитів.
- Безпека: Регулярні безпекові аудити допомагають виявити вразливості та запобігти кібератакам. Захист від SQL-ін'єкцій, XSS-атак, впровадження HTTPS, а також регулярні оновлення програмного забезпечення є важливими складовими безпеки.

Виправлення помилок та додавання нового функціоналу

Підтримка сайту включає постійне вдосконалення та адаптацію до нових вимог користувачів та ринку. Це може включати:

- Виправлення багів: Регулярний аналіз зворотного зв'язку від користувачів та моніторинг роботи сайту допомагає виявляти та оперативно виправляти технічні помилки та баги.
- Оновлення функціоналу: Впровадження нових функцій та модулів на основі потреб користувачів та нових технологій. Це може бути інтеграція нових платіжних систем, вдосконалення кошика покупок, особистого кабінету користувача, додавання нових фільтрів та пошукових функцій.
- Поліпшення UX/UI: Регулярний перегляд та оновлення дизайну сайту відповідно до сучасних тенденцій у веб-дизайні, проведення A/B тестувань для покращення користувацького досвіду.

Таким чином, регулярне оновлення контенту, моніторинг продуктивності та безпеки, а також постійне вдосконалення функціоналу є важливими складовими підтримки успішного інтернет-магазину, що дозволяє залишатися конкурентоспроможним та задовольняти потреби своїх клієнтів.

- Моніторинг продуктивності та безпеки

Моніторинг продуктивності та безпеки веб-сайту є важливою складовою його успішного функціонування та захисту даних користувачів. Це комплекс заходів, спрямованих на забезпечення стабільної роботи сайту, швидкої обробки запитів, захисту від зловмисних атак і вразливостей.

Швидкість завантаження сторінок є ключовим фактором, що впливає на користувацький досвід. Дослідження показують, що користувачі, ймовірно, покинуть сайт, якщо сторінка завантажується більше трьох секунд. Інструменти, такі як Google PageSpeed Insights, GTmetrix, та Pingdom, дозволяють вимірювати швидкість завантаження сторінок і надають детальні рекомендації щодо її оптимізації. Оптимізація може включати зменшення розміру зображень, налаштування кешування браузера, мінімізацію та об'єднання CSS та JavaScript файлів.

Ефективність обробки запитів на сервері впливає на час відповіді сайту та загальну швидкість роботи. Важливо оптимізувати базу даних,

налаштування сервера та використовувати ефективні алгоритми для обробки даних. Для цього можна використовувати аналіз запитів до бази даних, балансування навантаження та використання CDN (Content Delivery Network). Балансування навантаження дозволяє розподілити навантаження між декількома серверами, забезпечуючи стабільну роботу під час пікових навантажень. Використання CDN дозволяє розміщувати копії сайту на серверах по всьому світу для забезпечення швидкого доступу користувачів з різних регіонів.Захист від SQL-ін'єкцій є одним з найпоширеніших методів атак на веб-сайти, що використовують бази даних. Вони дозволяють зловмисникам виконувати небажані SQL-запити, отримувати доступ до конфіденційної інформації або змінювати дані в базі. Для запобігання SQL-ін'єкціям необхідно використовувати підготовлені запити (prepared statements), валідацію та фільтрацію вводу, налаштування привілеїв бази даних.

XSS-атаки дозволяють зловмисникам впроваджувати шкідливий скрипт у веб-сторінки, які переглядають інші користувачі. Цей скрипт може красти конфіденційну інформацію, наприклад, кукі, або виконувати інші шкідливі дії. Для запобігання XSS-атакам необхідно використовувати ескейпінг виводу (output escaping), валідацію вводу, використання Content Security Policy (CSP). Ескейпінг виводу забезпечує екранування спеціальних символів перед виводом їх на сторінку, що унеможливлює виконання шкідливого коду. CSP дозволяє налаштувати політики безпеки, що обмежують виконання довільного коду на сторінці, запобігаючи XSS-атакам.

HTTPS забезпечує шифрування даних, що передаються між користувачем і сервером, що робить їх захищеними від перехоплення зловмисниками. Впровадження HTTPS включає отримання SSL/TLS сертифікату, налаштування веб-сервера та перенаправлення HTTP на HTTPS. Отримання SSL/TLS сертифікату можна здійснити через сертифікаційні центри або безкоштовні сервіси, такі як Let's Encrypt. Налаштування веб-

сервера забезпечує обробку HTTPS-запитів, а автоматичне перенаправлення всіх запитів з HTTP на HTTPS гарантує безпеку даних.

Постійне оновлення програмного забезпечення, плагінів та модулів є критично важливим для запобігання вразливостям, які можуть бути використані зловмисниками. Це включає систематичні перевірки наявності оновлень, тестування оновлень у тестовому середовищі та налаштування автоматичних оновлень для важливих компонентів, що забезпечують безпеку.

Для забезпечення ефективного моніторингу продуктивності та безпеки сайту можна використовувати різні інструменти та методи. Системи моніторингу, такі як Nagios, Zabbix, або New Relic, дозволяють відстежувати стан серверів, час відповіді та інші ключові показники продуктивності. Веб-аналітика, зокрема Google Analytics, надає дані про поведінку користувачів, швидкість завантаження сторінок та інші важливі метрики. Автоматизовані сканери вразливостей, такі як OWASP ZAP або Nessus, допомагають автоматично сканувати сайт на наявність вразливостей, забезпечуючи своєчасне виявлення та усунення проблем.

Таким чином, постійний моніторинг продуктивності та безпеки є необхідним для забезпечення безперебійної та захищеної роботи сайту. Це включає швидкість завантаження сторінок, обробку запитів, захист від поширених атак та регулярні оновлення. Використання відповідних інструментів і методів допомагає своєчасно виявляти та усувати проблеми, забезпечуючи високу якість обслуговування користувачів.

- Виправлення помилок та додавання нового функціоналу

Процес виправлення помилок та додавання нового функціоналу є надзвичайно важливим для підтримки веб-сайту в актуальному та ефективному стані. Цей процес включає кілька ключових етапів: виявлення помилок, діагностику, виправлення, тестування, планування нового функціоналу, розробку, тестування та впровадження.

Виявлення та діагностика: Процес виявлення помилок починається з регулярного моніторингу роботи сайту за допомогою інструментів логування

та моніторингу, таких як Sentry, LogRocket, або New Relic. Ці інструменти дозволяють відстежувати помилки в реальному часі, надаючи детальну інформацію про природу та контекст помилок. Логування допомагає виявляти не тільки критичні помилки, але й потенційні проблеми, які можуть вплинути на продуктивність або безпеку сайту.

Аналіз та виправлення: Після виявлення помилки розробники проводять детальний аналіз для визначення її причини. Це може включати аналіз логів, перевірку коду та відтворення ситуацій, які призвели до помилки. Важливо не тільки виправити конкретну помилку, але й зрозуміти, чому вона виникла, щоб запобігти її повторенню в майбутньому. Після цього команда розробників вносить зміни до коду, що виправляють помилку. Виправлення повинно бути ретельно протестоване, щоб впевнитися, що воно не призводить до нових проблем або не впливає негативно на інші частини системи.

Тестування: Тестування є критично важливим етапом у процесі виправлення помилок. Це включає різні види тестування, такі як модульне, інтеграційне, функціональне та тестування користувацького інтерфейсу. Модульне тестування перевіряє окремі компоненти системи, інтеграційне тестування перевіряє взаємодію між компонентами, а функціональне тестування забезпечує відповідність системи вимогам і специфікаціям. UI тестування дозволяє перевірити зовнішній вигляд і поведінку інтерфейсу з точки зору користувача. Всі ці види тестування допомагають виявити та виправити проблеми на ранніх етапах, забезпечуючи стабільну та безперебійну роботу сайту.

Додавання нового функціоналу

Планування: Додавання нового функціоналу починається з аналізу вимог користувачів та бізнесу. Це може включати збір зворотного зв'язку від користувачів, аналіз ринку та конкурентів, а також вивчення нових тенденцій та технологій. На основі цього аналізу визначаються конкретні функції, які слід додати, і створюється план їх впровадження. Планування включає

визначення пріоритетів, оцінку необхідних ресурсів та встановлення строків виконання завдань.

Розробка: Після планування розробники приступають до створення технічних завдань та написання коду. Важливо дотримуватися принципів модульності та повторного використання коду, що дозволяє спростити процес підтримки та розширення функціоналу в майбутньому. Розробка нового функціоналу може включати інтеграцію нових бібліотек, фреймворків або API, що додають додаткові можливості та покращують продуктивність системи.

Тестування та впровадження: Після завершення розробки новий функціонал ретельно тестується. Це включає всі види тестування, згадані раніше, а також додаткові тести для нових функцій. Тестування дозволяє виявити можливі проблеми та впевнитися, що новий функціонал працює коректно і не порушує роботу існуючих компонентів системи. Після успішного тестування новий функціонал впроваджується на робочий сайт. Це може бути поступове впровадження, щоб мінімізувати ризики та забезпечити безперебійну роботу сайту. Важливо також забезпечити підтримку користувачів, навчання їх новим можливостям та надання відповідної документації.

Постійний розвиток та підтримка: Додавання нового функціоналу є безперервним процесом, оскільки вимоги користувачів та ринкові умови постійно змінюються. Команда розробників повинна постійно аналізувати зворотний зв'язок, вивчати нові тенденції та технології, щоб забезпечити актуальність та конкурентоспроможність сайту. Регулярні оновлення контенту та функціоналу допомагають зберігати інтерес користувачів, підвищувати їх задоволеність та забезпечувати зростання бізнесу.

Підсумовуючи, процес виправлення помилок та додавання нового функціоналу є критично важливим для забезпечення стабільної, безпечної та ефективної роботи веб-сайту. Він включає регулярний моніторинг, діагностику, тестування, розробку та впровадження нових можливостей, що

відповідають потребам користувачів та бізнесу. Використання сучасних інструментів та методів допомагає забезпечити високу якість обслуговування користувачів і підтримувати конкурентоспроможність на ринку.

Висновок

Під час виконання даної роботи було проведено глибокий аналіз існуючих рішень і технологій для створення веб-сайтів електротехнічних інтернет-магазинів. Було вивчено ринок, основних гравців та конкурентів, а також тренди та прогнози розвитку.

Було проаналізовано та порівняно різні технології для створення веб-сайтів, включаючи мови програмування, фреймворки, бібліотеки, серверні технології та бази даних. Визначено переваги та недоліки популярних CMS для інтернет-магазинів та надано рекомендації щодо їх вибору.Під час розробки сайту було ретельно обрано технологічний стек, розроблено структуру сайту, створено дизайн з урахуванням UX/UI вимог, та реалізовано необхідний функціонал, включаючи каталог товарів, кошик покупок, систему оплати та обробки замовлень, особистий кабінет користувача та інтеграцію з платіжними системами.

Проведено тестування функціональності, оптимізацію продуктивності та заходи з безпеки сайту. Підготовлено сайт до запуску та забезпечено його підтримку.Для подальшого розвитку проекту рекомендується регулярно оновлювати контент та функціонал сайту відповідно до зворотного зв'язку від користувачів та змін на ринку. Слід постійно моніторити продуктивність та безпеку сайту, вчасно виправляти помилки та додавати нові функції для покращення користувацького досвіду.

Важливо підтримувати актуальність та конкурентоспроможність сайту, використовуючи новітні технології та практики. Для подальшого розвитку проекту рекомендується постійно оновлювати контент сайту, моніторити продуктивність та безпеку, а також впроваджувати нові функціональні можливості відповідно до потреб користувачів та ринкових трендів. Важливим аспектом є активне використання маркетингових стратегій для залучення нових клієнтів та підвищення лояльності існуючих.Необхідно також враховувати швидкі зміни в технологічному середовищі, тому регулярні оновлення та адаптація до нових технологій є критично важливими.

Рекомендується розглянути можливість впровадження новітніх технологій, таких як штучний інтелект для персоналізації покупок та аналітики, а також розширення функціоналу сайту за рахунок інтеграції з новими платіжними системами та сервісами доставки.

Таким чином, наш проект має всі передумови для успішного функціонування та подальшого розвитку, відповідаючи потребам ринку та очікуванням користувачів.

Література

1. Бондаренко О.В. Розробка веб-сайтів. - Київ: Видавництво XYZ, 2020. - 320 с.
2. Іванов С.І. Сучасні технології веб-розробки. - Харків: Видавництво ABC, 2019. - 295 с.
3. Петров В.П. Основи електронної комерції. - Львів: Видавництво DEF, 2021. - 270 с.
4. Smith J. Modern Web Development. - New York: TechBooks, 2018. - 350 p.
5. Johnson M. eCommerce Solutions. - London: WebPress, 2019. - 315 p.
6. Гусєв А.І. Основи програмування для веб. - Одеса: Видавництво GHI, 2022. - 300 с.
7. Kaur S. Database Management Systems. - Delhi: EduBooks, 2020. - 280 p.
8. Brown R. JavaScript Frameworks. - San Francisco: CodeBooks, 2018. - 310 p.
9. Lee D. Full Stack Development. - Tokyo: DevBooks, 2019. - 290 p.
10. Нікітін М.О. Безпека веб-додатків. - Дніпро: Видавництво JKL, 2021. - 320 с.
11. Anderson C. SEO Strategies for eCommerce. - Boston: WebBooks, 2020. - 275 p.
12. White P. Web Performance Optimization. - Toronto: SpeedBooks, 2018. - 285 p.
13. Kumar V. CMS Solutions for eCommerce. - Mumbai: TechPress, 2019. - 305 p.

Додаток А

Код сайту на мові розмітки HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Phone Catalog</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <header>
        <nav>
            <div class="logo">Phone Catalog</div>
            <ul>
                <li><a href="/">Home</a></li>
                <li><a href="/catalog">Catalog</a></li>
                <li><a href="/favorites">Favorites</a></li>
                <li><a href="/cart">Cart</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <section id="home">
```

```
            <h1>Welcome to the Phone Catalog</h1>
            <p>Discover the latest phones and accessories</p>
        </section>
        <!-- Add more sections as needed -->
    </main>
    <footer>
        <p>&copy; 2024 Phone Catalog. All rights reserved.</p>
    </footer>
    <script src="main.js"></script>
</body>
</html>
```

Додаток В

Код сайту на мові CSS

```
body {
    margin: 0;
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
    color: #333;
}
header {
    background-color: #333;
    color: #fff;
    padding: 10px 0;
    text-align: center;
}
nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 0 20px;
}
nav .logo {
    font-size: 1.5em;
}
nav ul {
```

```
    list-style: none;
    margin: 0;
    padding: 0;
    display: flex;
}
nav ul li {
    margin-left: 20px;
}
nav ul li a {
    color: #fff;
    text-decoration: none;
    font-size: 1em;
}
nav ul li a:hover {
    text-decoration: underline;
}
main {
    padding: 20px;
}
#home {
    text-align: center;
    padding: 50px 0;
}
footer {
```

```
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 10px 0;
    position: fixed;
    width: 100%;
    bottom: 0;
}
```

Додаток С

Код сайту на мові JAVA SCRIPT

```
document.addEventListener("DOMContentLoaded", function() {
    const navLinks = document.querySelectorAll("nav ul li a");

    navLinks.forEach(link => {
        link.addEventListener("click", function(event) {
            event.preventDefault();
            const sectionId = this.getAttribute("href").substring(1);
            const section = document.getElementById(sectionId);
            window.scrollTo({
                top: section.offsetTop,
                behavior: "smooth"
            });
        });
    });
});

function addToFavorites(productId) {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || [];
    if (!favorites.includes(productId)) {
        favorites.push(productId);
        localStorage.setItem('favorites', JSON.stringify(favorites));
        alert("Product added to favorites!");
```

```
    } else {
        alert("Product is already in favorites.");
    }
}

function addToCart(productId) {
    let cart = JSON.parse(localStorage.getItem('cart')) || [];
    let product = cart.find(item => item.id === productId);
    if (product) {
        product.quantity += 1;
    } else {
        cart.push({ id: productId, quantity: 1 });
    }
    localStorage.setItem('cart', JSON.stringify(cart));
    alert("Product added to cart!");
}
```