Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної підготовки

випускна кваліфікаційна робота бакалавра

«ІНФОРМАЦІЙНА СИСТЕМА УПРАВЛІННЯ КОНТЕНТОМ ЕЛЕКТРОННОЇ КОМЕРЦІЇ»

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проекту

№	П.І.Б.	Група
1	Кадочніков Д. С.	КН-П-192
2	Крупиця Д. В.	КН-П-192
3	Єфремова М. Ю.	КН-Д-121
4	Тімченко В. О.	КН-П-192

Автор звіту

_____ Кадочніков Даніїл Сергійович

Одеса – 2023

АНОТАЦІЯ

УДК 004.9
К-84

Кадочніков Д. С. Інформаційна система управління контентом електронної комерції: Автореферат випускної кваліфікаційної роботи на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп'ютерні науки». - Одеса: ОТУШ, 2023 - 13 с.

Робота присвячена дослідженням, розвиненню та впровадженню інформаційної системи, головним призначенням якої є керування контентом електронної комерції. Додаткові функції системи включають реєстрацію, авторизацію та автентифікацію, засоби обмеження доступу до контенту, ведення журналів активності користувачів системи.

Актуальність роботи визначається швидкістю розвинення електронної комерції та просуванням електроних систем управління контентом українського виробництва, які матимуть змогу створювати конкуренцію на ринку електронної комерції. Особливу увагу звертає на себе той факт, що ці системи постійно розвиваються та модернізуються. Це дозволяє класифікувати задачі як такі, що формулюються в умовах швидкого розвинення інформаційних технологій. Додатково, актуальність диктується потребою впровадження мобільних інформаційних систем та реалізації у них покращених засобів захисту інформації, зокрема, персональних даних

Об'єктом дослідження у рамках даної роботи виступає алгоритм складання комплексної оцінки результатів за багатьма критеріями в умовах неповної визначеності вихідних умов. Предметом дослідження обрано різноманітність контенту, його зберігання, та видача цих даних за запитом, виконуючи всі умови(існують чи дані, чи є доступ до цих даних і т.п.). У якості задач дослідження було встановлене наступне:

- провести огляд інформаційних систем, які керують контентом електронної комерції.
- визначити програмні засоби, які дозволяють додавати дані про контент, зберігати їх, видавати за запитом.
- скласти власне програмне забезпечення, реалізувавши у ньому результати, досягнуті у попередніх задачах, провести випробування за різних умов та для різних пристроїв.
- впровадити у систему програмні засоби захисту даних для покращення показників інформаційної безпеки створеної системи.

При складанні звіту використано 7 посилання на друковані та електронні джерела інформації.

ЗМІСТ

ВСТУП

Розвиток і поширення інформаційних технологій супроводжується перенесенням великої кількості послуг у цифровий простір. Стратегія розвитку інформаційного суспільства в Україні визначає, що «системи управління контентом електронної комерції є необхідним інструментом соціально-економічного прогресу, одним з основних чинників інноваційного розвитку економіки» [1].

Інтернет комерція розвивається дуже швидко, тому кожна організації які займаються комерцією мають звернути на це увагу. Тому потрібно запропонувати їм сучасну систему управління контентом [2].

Швидкий розвиток електронної комерції потребує постійної модернізації систем управління контентом. Вона повинна ставати більш функціональною, безпечною та комфортною для використання [3]. Це актуалізує додаткові дослідження і розробки в зазначеній області.

В якості предмету даної роботи було обрано досить поширену систему управління контентом, до функцій якої належать пошук, перегляд, та замовлення товарів. При розміщенні контенту має враховуватись вузька спеціалізація платформи.

Також функції переважної більшості подібних системи мають передбачати можливість збереження історії маніпуляцій з контентом, пошуку пропозицій, а також фактів їх прийняття чи відхилення. У перспективі - засобів оплати. Відповідно, це передбачає засоби авторизації та автентифікації, а також убезпечення персональних даних від несанкціонованого доступу інших користувачів.

Детальне технічне завдання до проекту наводиться далі. У рамках даного звіту розглядається клієнтська частина, що являє собою веб-застосунок. Звіт складається з трьох розділів у яких описується загальна характеристика системи, деталі створення клієнтської частини та результати її розгортання та тестування. Висновки відбивають результати, що досягнуті у рамках даної частини проекту, загальний висновок є сукупністю звітів усіх учасників проекту.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЕКТУ

Загальний опис.
Програмне забезпечення (ПЗ) призначене для створення розподіленої системи електронної комерції. Головна функція ПЗ полягає в наданні функціональної системи перегляду та взаємодії з контентом для клієнтів. Цей реферат

Призначення:
ПЗ створюється за прикладом маркетплейсу, у той же час має складатись за принципом універсальності, мати змогу змінити цільове призначення.

Вимоги до функціоналу:
Реєстрація користувачів:
Користувач ПЗ має змогу зареєструватись за допомогою авторизаційних даних: номеру телефону та електронної пошти.

Особистий кабінет покупця
Так само, являє собою основний робочий засіб. Має реалізовувати можливості

- Перегляд історії покупок
- Перегляд списку “Бажане”
    - додавання товарів зі списку в “Корзину”
    - видалення товарів зі списку
- перегляд та зміна особистої інформації
    - логін
    - пароль
    - ім’я
    - адреса електронної пошти
    - номер телефону
- Вихід з акаунту

Інструмент переглядання та взаємодії з контентом

ПЗ має надати інструмент доступу до контенту. Клієнт має мати можливість переглядати товари, додавати їх до списків, зокрема “Бажане” та “Корзина”. Ці списки є індивідуальними для кожного клієнта.

Платіжний засіб
ПЗ надає користувачу вибір засобів оплати. Це може бути оплата онлайн, або оплата після отримання товару.

Архітектура
ПЗ має бути спроектоване за клієнт-серверною архітектурою:

- Backend - серверне програмне забезпечення, що забезпечує збереження та оброблення поточних даних, взаємодію з базою даних. Надає програмний інтерфейс взаємодії (API) для використання клієнтом
- WebApp - застосунок, доступний з веб-простору (сайт), що забезпечує візуальний інтерфейс користувача і комунікує з Backend за допомогою API.

.

Технології

Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контроля версій. Структура файлів та каталогів повинна відбивати архітектуру ПЗ, на зразок:

- root
    - Backend
        - Конфігурації
        - Контролери
        - Документація
        - Моделі даних
        - Сервіси
        - Методи взаємодії з БД (CRUD)
        - Допомічні модулі
    - Web
        - Веб-компоненти загального використання
        - Веб-компоненти сторінкок
        - Сервіси взаємодії з API
        - Моделі даних

Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології створення:

- Backend - C#,.NET, MSSQL, ASP.NET
- Web - JavaScript, HTML, CSS,або JavaScript фреймворки, зокрема React, Angular

Вихідний код повинен оформлятися у відповідності до загальноприйнятих стандартів, на зразок стандартів GNU http://www.gnu.org/prep/standards/, у тому числі з додержанням вимог до коментування та документування коду.

РОЗДІЛ 1
ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

У відповідності до технічного завдання (ТЗ) система була спроектована за клієнт-серверною розподіленою архітектурою. У складі системи було умовно відокремлено наступні архітектурні частини:

- «Mockup»,
- «Backend»,
- «Frontend».

Частина «Mockup» є набором моделей та описів поведінки користувача. Модель подає інтерфейси взаємодії користувачів різного типу з інформаційною системою, траєкторії рухів та переходів між різними інтерфейсами. При створенні інтерфейсів застосовано дизайнерський інструмент Figma. Розробка інтерфейсу користувача відбувалась із урахуванням принципів розумної навігації, включно із підходами UI/UX

Частина Mockup доступа за адресою: https://www.figma.com/file/kXnNSvll8JZYSx44PQyAbs/%D0%94%D0%B8%D0%BF%D0%BB%D0%BE%D0%BC?type=design&node-id=0%3A1&t=BHFerLjmWwv6IIPJ-1.

Частина «Backend» реалізована як централізований вузол збереження та оброблення даних, що створюються та модифікуються протягом життєвого циклу інформаційної системи. Дана частина включає у себе базу даних проекту та програмний інтерфейс (API) доступу до основних її функцій, зокрема, керування контентом, авторизації, реєстрації, опрацювання замовлень. Частина «Backend» виконана на базі програмного забезпечення .NET та СУБД «MS SQL Server Management Studio». Додатково використан програмний пакет «Entity Framework»

Клієнтська частина системи «Frontend», призначена для надання користувачеві доступу до функцій системи через засоби візуального інтерфейсу. Інтерфейс розрахований на використання за допомогою ПК, однак передбачена можливість реалізації аналогічного функціоналу для мобільних пристроїв.

Вихідні коди та ресурси даної частини розміщені на репозиторії https://github.com/curiousvlxd/SoundParadise.Web. Подальший звіт присвячений детальному опису цієї частини.

РОЗДІЛ 2
РЕАЛІЗАЦІЯ ГРАФІЧНОГО ІНТЕРФЕЙСУ КОРИСТУВАЧА

База даних інформаційної системи реалізована з використанням наступних технологій:

- Засоби розробки: мова програмування TypeScript з подальшим компілюванням у JavaScript, середовище Visual Studio Code
- Мова верстання веб-сторінок HTML
- Мова стилізації веб-сторінок CSS
- Фреймворк для побудови односторінкових веб-додатків Angular 2
- Сервіс хмарних обчислень «Azure App Services».

Основна задача клієнтської частини застосунку - це встановлення зрозумілої, чіткої та повної комунікації між користувачем (клієнтом магазину) та віддаленим сервером. Ця ціль досягається за допомогою декількох компонентів:

- графічних елементів: кнопки, поля вводу тексту, динамічні текстові написи, меню, спливаючі вікна тощо
- код, що опрацьовує взаємодію користувача з графічними елементами: натискання кнопки миші, введення тексту, відкриття меню тощо
- код, що контролює цілісність даних, введених користувачем (валідація форм), структурує дані певним чином перед відправкою на сервер Backend
- код, що безпосередньо комунікує із сервером, надсилаючи та отримуючи дані у вигляді JSON (JavaScript Object Notation)
- код, що отримує відповідь від сервера Backend та генерує новий набір графічних елементів, або змінює існуючий.

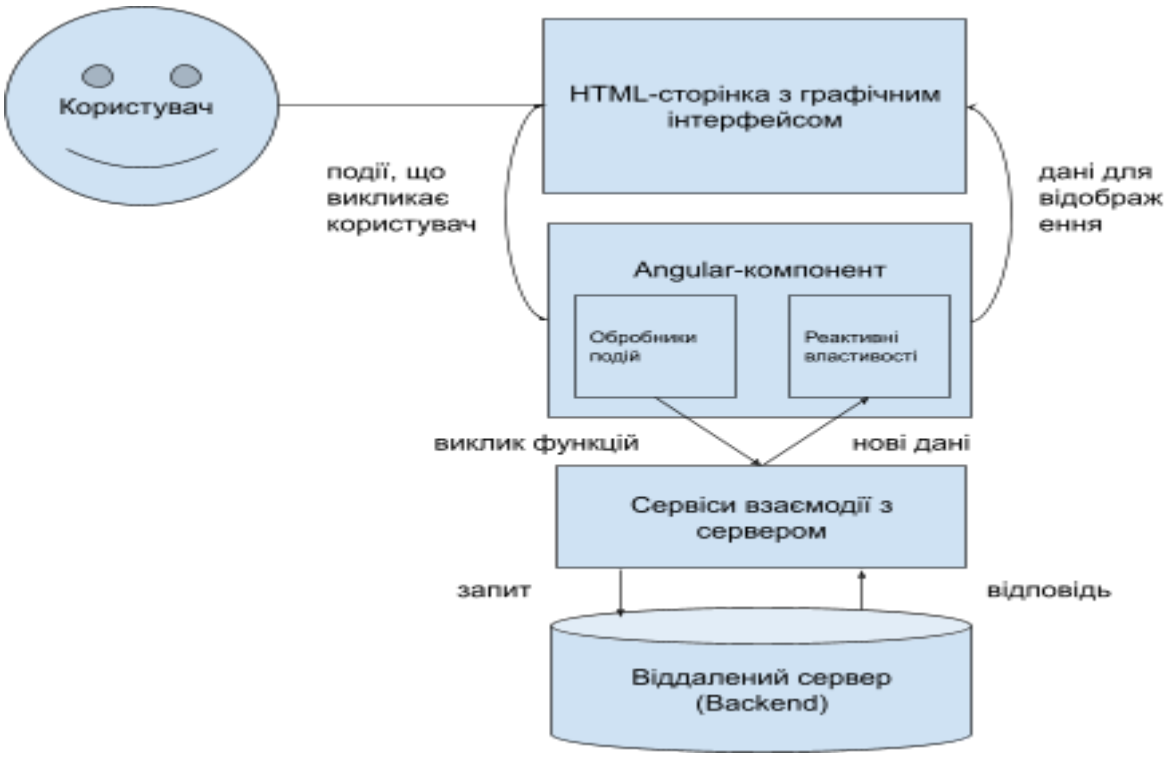

Рисунок 1 - діаграма взаємодії частин системи UI

Далі наведено програмний код для більш детального розуміння процесу роботи кожної частини. Як приклад, використано Angular-компонент “SearchBarComponent”, який є реалізацією пошукового рядка. Компонент дає можливість пошуку товарів у системі електронної комерції. Також він спрощує цей процес, надаючи динамічні спливаючі підказки про можливі продукти, які користувач бажає отримати.

```
<div [class.suggestions-open]="suggestionsOpen" class="bar">
  <img src="../assets/search.svg"/>
        <input     (input)="updateSuggestions()"     [(ngModel)]="query"
[class.suggestions-open]="suggestionsOpen"
        aria-label="Я шукаю..." placeholder="Я шукаю..." type="text">
    <a   (click)="search()"   [class.suggestions-open]="suggestionsOpen"
id="search-btn" role="button">Знайти</a>
</div>
<div *ngIf="suggestionsOpen" class="suggestions card">
  <div *ngFor="let s of suggestions">
    <li><a [href]="'product/'+s.id">{{s.name}}</a></li>
    <hr/>
  </div>
</div>
```

Фрагмент коду 1

Фрагмент коду 1 є описом макету компонента. Цей код написаний варіацією мови HTML, яка була розширена інструментами шаблонізації сторінок, що надає фреймворк Angular, такими як:

- двобічна синхронізація даних (data binding) за допомогою ngModel
- директива [class.*] для визначення поточного стилю компонента або його частини
- (event) - синтаксис для встановлення обробників подій, у цьому випадку це натискання на кнопку “Пошук”

Взаємодія між цим макетом та програмним кодом TypeScript, що визначає поведінку компонента та дані, якими заповнюється макет, відбувається за допомогою вбудованих засобів фреймворка Angular. Реактивну синхронізацію даних (data binding), зокрема, забезпечує пакет RxJS.

Реактивність є ключовою властивістю сучасних JavaScript/TypeScript фреймворків. Широке використання цих фреймворків [4] свідчить про необхідність ефективної організації потоків даних (data flow). Реактивні компоненти, які присутні у тому чи іншому вигляді у більшості таких фреймворків, є рішенням цієї проблеми.

```
import {Component} from '@angular/core';
import {Product} from 'src/product';
import {SearchService} from '../search.service';

@Component({
  selector: 'app-search-bar',
  templateUrl: './search-bar.component.html',
  styleUrls: ['./search-bar.component.css']
})
export class SearchBarComponent {
  suggestions: Array<Product> = [];
  query: string = "";

  constructor(private searchService: SearchService) {}

  get suggestionsOpen() :boolean {
    return this.suggestions.length > 0;
  }

  updateSuggestions() {
    if (this.query.length < 2) {
      this.suggestions = [];
```

```
    } else {
      this.searchService.getSuggestions(this.query).subscribe(data => {
        this.suggestions = data as Array<Product>;
      });
    }
  }

  search() {
    window.location.replace("/search?query=" + this.query)
  }
}
```

Фрагмент коду 2

Фрагмент коду 2 демонструє логіку пошукового рядку. Обробником події (input) є функція updateSuggestions, що використовує сервіс SearchService та оновлює список підказок. Код цього сервіса представлено нижче.

```
import {HttpClient} from '@angular/common/http';
import {Injectable} from '@angular/core';
import { environment } from '../environments/environment';

@Injectable({
  providedIn: 'root'
})
export class SearchService {

  constructor(private http: HttpClient) {
  }

  getSuggestions(query: string, limit = 4) {
                return     this.http.get(environment.BASE_API_URL     +
'products/search-suggestions?query=' + query + "&limit=" + limit);
  }

  getFullResults(query: string, limit:number=100) {
                return     this.http.get(environment.BASE_API_URL     +
'products/search-products?query=' + query + "&limit=" + limit);
  }
}
```

Фрагмент коду 3

За подібною схемою було реалізовано 20+ компонентів у головному модулі:

- галерея-карусель постерів
- сторінка “Кошику”, яка дає можливість переглядати та редагувати список товарів, які планує замовити користувач
- окремий товар у “Кошику”
- великий каталог товарів
- маленький каталог товарів
- сторінки категорії та підкатегорії
- картка категорії
- картка товару
- кнопка “додати в Бажане”
- нижня частина сайту (футер)
- верхня частина сайту з меню (хедер)
- головна сторінка
- сторінка помилки 404
- сторінка створення замовлення
- сторінка результатів пошуку
- сторінка товару

Деякий пов’язаний між собою функціонал був винесений у окремі модулі Angular:

- модуль профілю користувача - містить такі компоненти як список “Бажане”, налаштування профілю, список платіжних карт користувача, історія замовлень
- модуль аутентифікації та реєстрації
- модуль компонентів загального використання, наприклад: кнопка, модальне вікно, індикатор завантаження.

Для комунікації з API використовуються сервіси:

- AuthService - аутентифікація та реєстрація
- CardService - редагування списку платіжних засобів
- CategoryService - отримання списку категорій та підкатегорій
- FavoritesService - редагування списку “Бажане”
- OrdersService - створення та отримання замовлень
- ProductsService - отримання товарів
- SearchService - пошук товарів у базі даних
- UserService - отримання інформації про поточного користувача

Перед надсиланням на сервер запити обробляються за допомогою об’єкта-перехоплювача - HttpRequestInterceptor.

До кожного HTTP-запиту додаються заголовки (headers), що забезпечують можливість використання Cookie та доступу до ресурсів, що вимагають авторизації:

```
.set('Access-Control-Allow-Origin', "*")
.set(Access-Control-Allow-Credentials", "true"
.set('Cookie', jwt)
.set('Authorization', "Bearer " + jwt)
```

Фрагмент коду 4

Змінна “jwt” зберігає токен JSON Web Token, що є відкритим стандартом для передачі безпечних інформаційних пакетів між двома сторонами у форматі JSON. JWT-токен генерується на сервері та надсилається клієнту. Після його отримання на клієнтському боці токен зберігається як Cookie. Доступ до cookie браузера реалізований через бібліотеку ngx-cookie.

РОЗДІЛ 3
РЕЗУЛЬТАТИ ВИПРОБУВАНЬ

Результатом праці став програмний продукт, який є клієнтською частиною проекту онлайн-магазину (електронної комерції). Побудова продукту має комплексний характер: компоненти можуть мати декілька рівнів вкладеності.

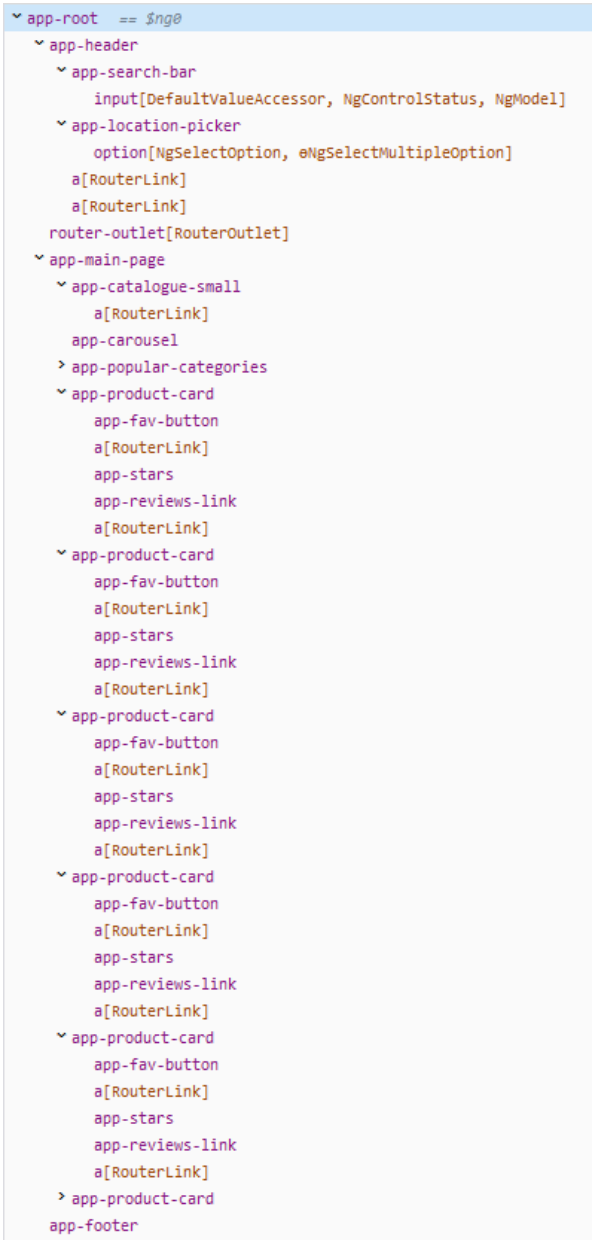

Рисунок 2 - ієрархія Angular-компонентів на головній сторінці сайту

Провівши дослідження, я дійшов висновку, що модульна архітектура програмного продукту є оптимальною в контексті електронної комерції. Таким чином, JavaScript фреймворки для створення SPA (single page application) [5], такі як Angular є ефективними засобами побудови веб-застосунків у цій предметній області.

Angular пропонує надійну та масштабовану архітектуру, яка добре підходить для побудови комплексних та багатих на функції веб-сайтів електронної комерції. Двостороння функція зв'язування даних спрощує процес оновлення та синхронізації даних між інтерфейсом інтерфейсу та моделлю даних програми. На веб-сайті електронної комерції, де оновлення в режимі реального часу мають важливе значення, ця функція дозволяє безперебійній синхронізації інформації про товари, ціни та інвентаризації.

Реактивні форми забезпечують гнучкий та надійний спосіб перевірки форм та користувацьких даних у онлайн-магазинах. Ця функція спрощує перевірку та управління складними формами, такими як форми створення замовлення або форми реєстрації користувачів, забезпечення точності даних та зменшення помилок під час процесу закупівлі.

У підсумку, всі функції веб-застосунку, що були регламентовані технічним завданням було реалізовано та випробувано. Публічна версія сайту розміщена за адресою https://soundparadise.store, для хостингу був використаний сервіс Microsoft Azure. Вихідний код застосунку розміщений у репозиторії https://github.com/curiousvlxd/SoundParadise.Web/tree/dev.

ВИСНОВКИ

Проведено огляд інформаційних систем, які керують контентом електронної комерції.

Спроектовано, реалізовано та випробувано систему перегляду та взаємодії з контентом електронної комерції. Система складена за розподіленою клієнт-серверною архітектурою та складається з трьох частин, розміщених за адресами:

- «Mockup» https://www.figma.com/file/gDr7qqU2ZRd0wt3giXqOnf,
- «Backend», https://github.com/curiousvlxd/SoundParadise.Web/tree/dev/SoundParadise.Api
- «Frontend» https://github.com/curiousvlxd/SoundParadise.Web/tree/dev/SoundParadise.Client

Клієнтська частина додатку має графічний інтерфейс користувача та дозволяє кінцевому користувачу комунікувати з віддаленим сервером, що працює з даними електронної комерції.

У систему впроваджено засоби покращення інформаційної безпеки: використання протоколу HTTPS, токенів JWT, підтвердження персональних даних користувача, зокрема, електронна пошта. Це ускладнює несанкціонований доступ до акаунтів зловмисниками.

Систему було випробувано шляхом інсталяції у хмарні сховища та проведення тестових сеансів комунікації. Результати випробування підтвердили загальну працездатність створеної системи.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Стратегія розвитку інформаційного суспільства в Україні. Розпорядження Кабінету Міністрів України від 15 травня 2013 р. № 386-р. // Урядовий портал. URL: https://www.kmu.gov.ua/npas/246420577

2. Стратегія кібербезпеки України. Указ Президента України від 14 травня 2021 року // Урядовий кур'єр від 28.08.2021 — № 165. URL: https://zakon.rada.gov.ua/laws/show/447/2021/print

3. E-commerce в Україні: цифри, факти, перспективи розвитку онлайн-торгівлі // URL:https://www.site2b.ua/web-blog/e-commerce-v-ukraine-cifry-fakty-perspektivy-razvitiya-onlajn-torgovli.html

4. Front-end frameworks popularity (React, Vue, Angular and Svelte) URL: https://gist.github.com/tkrotoff/b1caa4c3a185629299ec234d2314e190

5. Що таке SPA-додатки https://wezom.com.ua/ua/blog/chto-takoe-spa-prilozheniya