КОМП'ЮТЕРНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

УДК 004.055
DOI https://doi.org/10.32782/tnv-tech.2024.1.1

ІННОВАЦІЙНІ МЕТОДИ ВІДОБРАЖЕННЯ ІНФОРМАЦІЇ У ВЕБ-БРАУЗЕРАХ

Антоненко А. В. *– кандидат технічних наук, доцент,*
доцент кафедри стандартизації та сертифікації сільськогосподарської продукції
Національного університету біоресурсів і природокористування України
ORCID ID: 0000-0001-9397-1209

Балвак А. А. *– аспірант кафедри комп'ютерної інженерії*
Державного університету інформаційно-комунікаційних технологій
ORCID ID: 0000-0002-6441-8225

Цвик О. С. *– аспірант кафедри комп'ютерної інженерії*
Державного університету інформаційно-комунікаційних технологій
ORCID ID: 0000-0001-7786-1712

Ємелін Д. М. *– магістр кафедри комп'ютерної інженерії*
Державного університету інформаційно-комунікаційних технологій
ORCID ID: 0009-0002-6084-0768

Гришковець Є. П. *– бакалавр кафедри комп'ютерної інженерії*
Державного університету інформаційно-комунікаційних технологій
ORCID ID: 0009-0009-6346-966X

Стаття присвячена методам відображення інформації в браузерах, недоліки та переваги кожного з методів, можливість їхнього комбінування і як саме це покращує користувацький досвід та оптимізує використання ресурсів серверу та клієнту. В сучасному світі, де доступ до Інтернету став невід'ємною частиною нашого повсякденного життя, веб-додатки та веб-сайти мають велике значення для різних видів діяльності, від освіти та розваг до бізнесу та комунікації. Однак, з розвитком Інтернету та зростанням кількості користувачів, виникає все більше проблем зі швидкістю завантаження веб-сторінок

та веб-додатків, а також з обробкою великої кількості даних. Тому, оптимізація методів відображення інформації в браузерах є важливим завданням для веб-розробників та власників сайтів. У статті детально розглянуто різні методи відображення інформації в браузерах, їх переваги та недоліки, а також можливість їх комбінування для досягнення найкращого результату. Методи відображення інформації в браузерах визначає яким чином дані, які будуть показані користувачу, будуть оброблені, як саме будуть розподілені ресурси сервера та клієнта. Розвиток даних методів дозволяє покращити процес використання обраховувальних та інтернет ресурсів, заощаджувати інтернет-трафік та створювати більш приємні з точки зору користувацького досвіду веб-додатки, що є ключовим фактором для поширення продукту та збільшення на нього попиту. Зокрема, нами розглянуто традиційний метод відображення статичних веб-сторінок, а також новіші методи, такі як SSR, CSR, ISR. У роботі розглянуто, як комбінування різних методів може допомогти покращити користувацький досвід та оптимізувати використання ресурсів серверу та клієнта. У статті визначено, який метод відображення інформації в браузерах відповідає вашим потребам, як його використовувати для досягнення максимальної ефективності та як покращити користувацький досвід веб-додатків.

Ключові слова: *рендеринг, відображення інформації, браузер, веб-розробка, методи відображення інформації, оптимізація.*

Antonenko A. V., Balvak A. A., Tsvyk O. S., Yemelin D. M., Hryshkovets Ye. P. Innovative methods of information display in web-browsers

The article is devoted to the methods of displaying information in browsers, the disadvantages and advantages of each of the methods, the possibility of combining them and how exactly this improves the user experience and optimizes the use of server and client resources. In today's world, where access to the Internet has become an integral part of our daily lives, web applications and websites play an important role in various activities, from education and entertainment to business and communication. However, with the development of the Internet and the increase in the number of users, there are more and more problems with the loading speed of web pages and web applications, as well as with the processing of large amounts of data. Therefore, optimizing methods of displaying information in browsers is an important task for web developers and site owners. The article discusses in detail various methods of displaying information in browsers, their advantages and disadvantages, as well as the possibility of combining them to achieve the best result. The methods of displaying information in browsers determine how the data that will be shown to the user will be processed, exactly how the server and client resources will be distributed. The development of these methods allows improving the process of using computing and Internet resources, saving Internet traffic and creating more pleasant web applications from the point of view of user experience, which is a key factor for the distribution of the product and increasing its demand. In particular, we have considered the traditional method of displaying static web pages, as well as newer methods such as SSR, CSR, ISR. The paper considers how combining different methods can help improve user experience and optimize the use of server and client resources. The article defines which method of displaying information in browsers meets your needs, how to use it to achieve maximum efficiency, and how to improve the user experience of web applications.

Key words: *displaying information, browser, web development, methods of displaying information, optimization.*

Вступ. В сучасному світі інформація є одним із найцінніших ресурсів сьогодення, щодня ми поглинаємо її з найрізноманітніших джерел, одним із головних яких є веб-сайти. І саме швидкість доступу до інформації на них є одним із ключових факторів, які впливають на попит тою, чи іншою інформацією. Проблема відображення інформації в браузерах виникає, коли веб-сторінка має складний вміст, такий як великі об'єми даних, зображень, відео та іншого мультимедіа. Коли користувач запитує веб-сторінку, браузер повинен відобразити весь цей вміст. Процес рендерингу може зайняти значну кількість часу, що призводить до поганої продуктивності та довгих часів завантаження сторінки.

Постановка проблеми. Проблема особливо актуальна для мобільних пристроїв, де швидкість зв'язку може бути обмежена, що ще більше ускладнює рендеринг сторінки. Прикладом може слугувати веб-портал з безліччю статей та

інтерактивних елементів, які дозволяють користувачу динамічно змінювати відображення інформації.

Для вирішення цієї проблеми, розробники використовують різні методи оптимізації рендерингу, такі як ліниве завантаження (lazy loading), кешування, оптимізація зображень та відео та інші. Також, розробники можуть використовувати різні технології, такі як серверний рендеринг (server-side rendering) та клієнтський рендеринг (client-side rendering), щоб покращити продуктивність та швидкість завантаження сторінки.

Метою роботи є розроблення методології відображення інформації в веб-браузерах.

Об'єкт дослідження – методи відображення інформації в веб браузерах.

Предмет дослідження – наявні шаблони відображення інформації.

Аналіз останніх досліджень і публікацій. Наукове обґрунтування та розроблення методології відображення інформації є актуальним завданням, розв'язання якого дозволить робити веб-додатки більш якісними, доступнішими та швидщими, що в свою чергу сприятиме кращому поширенню продукції.

Значний внесок у дослідженні та концептуалізації методів відображення інформації внесли дослідження іноземних розробників та авторів: Ching S., Sbarski P., Verma K., Abdullah I., Valverde A., Robinson L., Skuratsivska O., Samaranayake M., Malcolm., Fayock C., Grigoryan A., Breux G., Miller J., Osmani A., Cocca J., Swastik Y., Prashant M., Gathony M. та ін. [1–15].

Виклад основного матеріалу дослідження. Інноваційні методи рендерингу даних в браузерах активно розвиваються, оскільки вони можуть покращити продуктивність веб-додатків і забезпечити кращий користувацький досвід. Деякі з існуючих методів рендерингу включають Static Site Generation, Server-side Rendering, Client-side Rendering та ISR [16–17].

Static Site Generation (SSG) – це метод, який використовується для відображення статичного(попередньо згенерованого) контенту, який передається з сервера на клієнт.

Server-side Rendering (SSR) – це метод, який використовується для генерації HTML на сервері, перед його відправкою на клієнт. Це дозволяє зменшити час завантаження сторінки та покращити її SEO.

Client-side Rendering (CSR) – це метод, коли весь HTML, CSS та JavaScript завантажуються на клієнті, і весь контент генерується на стороні клієнта. Це забезпечує швидку відповідь додатку, але може знизити його SEO.

ISR – метод оптимізації статичного рендерингу веб-сторінок. Основна ідея застосування ISR полягає в тому, щоб побудувати сайт змісту (зміст, який не змінюється з часом), а потім динамічно генерувати лише ті сторінки, які є потрібними.

Розглянемо недоліки та переваги кожного з методів відображення інформації на веб-сторінках.

1. SSG (Static Site Generation) – це процес попередньої генерації (будування) HTML-сторінок сайту на стадії розробки. Замість того, щоб генерувати сторінки динамічно на сервері за запитом клієнта (SSR), SSG генерує статичні HTML-файли, які можуть бути повністю обслуговані без необхідності запуску сервера.

SSG є відмінним вибором для статичних сайтів або сайтів з декількома статичними сторінками, що не потребують частого оновлення. Це дозволяє збільшити швидкість завантаження сторінок, зменшити навантаження на сервер та підвищити їх безпеку. Веб-програми на основі JavaScript, як ми їх традиційно знаємо, працюють, запускаючи бібліотеки, як-от React, або сценарії під час виконання

в браузері. Коли браузер отримує сторінку, зазвичай це простий HTML без великої кількості вмісту. Потім завантажуються сценарії, щоб завантажувати вміст на сторінку, цей процес також відомий як гідратація.

За допомогою Static Generation такі інструменти, як Next.js, намагаються відобразити цю сторінку здебільшого так, як це було б у браузері, але під час компіляції. Це дає нам можливість обслуговувати весь вміст під час першого завантаження. Під час цього процесу сценарії все ще гідратують сторінку, але в ідеалі з меншою кількістю змін або взагалі без змін. SSG також може бути використаний в поєднанні з динамічними елементами на сторінці, що дозволяє отримати найкращі з обох підходів. Наприклад, використовуючи JavaScript, можна отримати динамічний контент, який не залежить від сервера, але все ж може бути використаний на сторінках, що були згенеровані статично [18].

Рендеринг на стороні сервера може покращити продуктивність першого завантаження та забезпечити можливість простого проходження сторінки пошуковими системами. Основними перевагами SSG є швидкість завантаження сторінок, зменшення навантаження на сервер, зниження ризику атак з боку зловмисників, підвищення безпеки. Основними недоліками SSG є не підходить для сайтів з великою кількістю динамічних елементів та потребує попередньої генерації сторінок під час розробки, що може бути.

2. У SSR веб-сервер генерує HTML-вміст веб-сторінки на сервері та надсилає його в браузер клієнта.. Застосування SSR дозволяє отримати зріз веб-сторінки з мінімальним завантаженням JavaScript на клієнтську сторону. Рендеринг на стороні сервера, або SSR, це можливість рендерити вашу програму, або принаймні її частину, на сервері, а не в браузер, що в свою чергу дозволяє користувачам з застарілими пристроями отримувати робочий та швидкий сайт та покращує SEO. Відтворення на сервері створює HTML на вимогу для кожної URL-адреси, але може бути повільнішим, ніж просто обслуговування статичного відтвореного вмісту. Якщо ви можете докласти додаткових зусиль, рендеринг сервера та кешування HTML може значно скоротити час рендеринга сервером. Перевагою серверного рендерингу є можливість отримувати більше «живих» даних і відповідати на більш повний набір запитів, ніж це можливо за допомогою статичного рендерингу. Сторінки, які потребують персоналізації, є конкретним прикладом типу запиту, який погано працюватиме зі статичним рендерингом [19].

До переваг SSR відносять:

– зниження TTFB (Time To First Byte) – часу, необхідного для отримання першого байту від сервера;

– покращення SEO – пошукові роботи можуть легше і швидше індексувати вміст сторінки, оскільки HTML-код сторінки генерується на сервері;

– підвищення доступності – веб-сторінка може бути показана користувачу, навіть якщо JavaScript не підтримується або відключений у браузері;

– кращу продуктивність на повільних пристроях або з поганим зв’язком з мережею.

До недоліків SSR відносять:

– складність налаштування серверної інфраструктури;

– більший розмір відповіді сервера, що може призвести до затримок при завантаженні сторінки на повільних з’єднаннях;

– потребує великої кількості серверних ресурсів для обробки великої кількості запитів користувачів;

– необхідність перезавантаження сторінки при зміні контексту взаємодії з користувачем;

– застосування SSR може допомогти знизити TTFB та поліпшити SEO, але потребує додаткової серверної інфраструктури і може збільшувати розмір відповіді сервера, що може призвести до затримок при завантаженні сторінки на повільних з'єднаннях;

– пропускна здатність SSR сервера значно менша за пропускну здатність CSR.

3. CSR (Client-Side Rendering) – це метод відображення сторінок веб-додатка, де HTML та JavaScript генеруються на стороні клієнта, тобто в браузері. Даний метод став популярним з розвитком так званих Single Page Application (SPA). Саме в цих додатках дані динамічно завантажуються на сторінці користувача в залежності від взаємодії з сайтом. За допомогою рішення для рендерингу на стороні клієнта ви перенаправляєте запит до одного файлу HTML, і сервер доставлятиме його без вмісту (або з екраном завантаження), доки ви не отримаєте весь JavaScript і не дозволите браузеру скомпілювати все перед рендерингом вмісту. [20]

Основна перевага CSR – це швидка відповідь на дії користувача, оскільки сторінка не перезавантажується при кожному запиті, а зміст змінюється динамічно. Це дає користувачу більш зручний та швидкий досвід використання додатку. Також CSR зменшує навантаження на сервер, оскільки лише необхідні дані запитуються та передаються з сервера. При рендерингу на клієнтській стороні (CSR) браузер завантажує JavaScript, а потім рендерить вміст на клієнті. Основною перевагою CSR є його інтерактивність та можливість змінювати стан додатка без запиту до сервера. Також важливою характеристикою є те, шо дані можуть бути отримані на основі таких подій, як завантаження сторінки або натискання кнопки, це може збільшити час завантаження/взаємодії програми є

Однак, існують деякі недоліки CSR. Негативний вплив на SEO та рейтинг сторінки, оскільки веб-сторінки CSR здебільшого порожні та містять лише посилання на код JS, який генерує HTML, веб-сканери можуть розглядати їх як порожні сторінки. Також CSR може бути менш безпечним, оскільки JavaScript виконується на стороні клієнта, і це може бути використано для атак.

Іншим недоліком CSR є те, що при використанні цього методу веб-додаток може стати важким та повільним для завантаження, особливо при великій кількості контенту, оскільки весь контент повинен бути завантажений на стороні клієнта перед відображенням. CSR є потужним інструментом для покращення швидкості та ресурсоємкості веб-додатка, але він має деякі недоліки, які можуть знизити його ефективність та безпеку.

4. ISR (Incremental Static Regeneration) – це метод генерації статичних сайтів, який поєднує переваги SSG та SSR. При використанні ISR, статична сторінка генерується за першим запитом, а потім оновлюється за необхідності через певний час або відповідно до встановленого розкладу. ISG дозволяє вам створювати або оновлювати статичні сторінки після того, як ви їх побудували через визначений проміжок часу. Таким чином, вам не потрібно буде перебудовувати весь сайт, лише ті сторінки, які цього потребують. Проте даний метод має свої недоліки, одним із яких є те що виконані зміни не відразу відображаються на сторінці, але тільки після певного інтервалу часу, завдяки кешуваню. Також даний метод не є ефективним для сайтів з великою кількістю динамічних елементів, оскільки це призведе до повільної генерації сторінок.

До переваг ISR належать:

– швидкість: із застосуванням ISR сторінки можуть бути відображені відразу з моменту запиту, якщо вони вже були згенеровані раніше – це зменшує час очікування для користувачів;

– потужність: ISR дозволяє динамічно генерувати статичні сторінки з даними, які змінюються з часом – це означає, що ви можете створювати складні додатки з динамічною вмістом і все ще зберігати швидкість, яку забезпечують статичні сайти;

– легкість у використанні: використання ISR дуже легко, оскільки ви можете використовувати різні фреймворки та бібліотеки, що дозволяє розробникам використовувати свої улюблені інструменти.

До недоліків ISR належать:

– надійність: ISR може мати проблеми з надійністю, оскільки статичні сторінки можуть стати застарілими в момент, коли користувачі їх запитують – це може призвести до того, що користувачі бачитимуть застарілий вміст;

– складність: ISR може бути складним у використанні, оскільки ви повинні знати, як він працює, щоб коректно використовувати його;

– потреба у підтримці: ISR потребує певної підтримки з боку сервера, щоб запускати рендеринг сторінок відповідно до заданого роз міру. Це може потребувати додаткових зусиль з боку розробника, щоб забезпечити належну підтримку та обслуговування серверів;

– обмежені можливості: ISR не підходить для всіх типів веб-сайтів та додатків (найкраще підходить для статичних сайтів, які мають мінімальну кількість динамічного вмісту, так як збільшення обсягу динамічного вмісту може значно збільшити час генерації сторінок);

– затримка оновлення: у випадку, якщо статична сторінка зберігається у кеші, оновлення контенту може затримуватися до моменту видалення сторінки з кешу або до зміни її TTL – це може вплинути на користувачів, які хочуть побачити найсвіжіший контент на сайті.

Аналізуючи вище указані методи візуалізації веб-сторінок, ми бачимо що всі вони мають свої переваги та недоліки і підходять для певного спектру задач. Тому необхідно розвиток більш універсальних методів відображення, які перш за все сприятимуть покращенню користувацького досвіду, сприятимуть кращому просуненню програмного продукту, та при можливості спростять у зроблять більш універсальною розробку веб-додатків. На даний момент уже є такий універсальний метод. Isomorphic Rendering (також відомий як Universal Rendering або Server-Side Rendering with Client Rehydration) – це комбінація технологій SSR і CSR, що дозволяє використовувати обидва методи рендерингу для отримання кращої продуктивності та ефективності веб-додатків. Ізоморфний рендеринг – це спосіб використовувати один і той самий код для рендерингу сторінок як на сервері, так і на клієнті.

До основних переваг Isomorphic Rendering належать:

– швидкість: застосування методу рендерингу на сервері дозволяє зменшити час завантаження сторінки, тоді як CSR дозволяє зменшити час оновлення сторінки без перезавантаження сторінки;

– пошукова оптимізація: Isomorphic Rendering дозволяє створювати веб-додатки, які краще індексуються пошуковими системами, оскільки весь контент рендериться на сервері;

– збільшення відповідальності: Isomorphic Rendering дозволяє зменшити навантаження на сервер, оскільки CSR може бути використаний для оновлення лише окремих частин сторінки.

До основні недоліків Isomorphic Rendering відносять:

– складність розробки: Isomorphic Rendering може бути складним у використанні, оскільки потрібно дотримуватись певних правил для належного використання SSR та CSR;

– потреба у ресурсах: Isomorphic Rendering може вимагати більше ресурсів для рендерингу на сервері, оскільки весь контент повинен бути готовий для відправки на клієнтську сторону;

– низька масштабованість: Isomorphic Rendering може бути менш масштабованим, оскільки при використанні SSR може виникати навантаження на сервер, коли багато користувачів одночасно запитують сторінки.

Даний метод справді є більш універсальним і дозволяє створювати веб додатки з гарною швидкістю, зберігаючи при цьому динамічність. Проте він має свої недоліки, наприклад як от складність розробки. Є певні обмеження для даного методу розробки, наприклад в React сторінка, яка підготовлюється на сервері повинна бути ідентична тій, що візуалізується на клієнті. Звісно динамічні елементи в процесі можуть змінювати це, але в цілому структура сторінок повинна бути ідентична. Розробка і обхід даного обмеження дозволить створювати більш гнучкіші додатки. Також питання навантаження на сервер стає більш актуальною проблемою, оскільки вже сервер затрачує ресурси на більш глибоку візуалізацію контенту сторінки. Дане питання вирішується масштабованістю ресурсів та оптимізацією їх використання. Вирішення даних проблем дозволить створювати більш якісні продукти використовуючи менше ресурсів.

Висновки. Отже, методи візуалізації інформації у веб-браузерах є актуальною проблемою, оскільки саме вирішення її дозволить створювати більш якісну продукцію, яка при цьому використовуватиме значно менші ресурси, що і є ключовим фактором в багатьох бізнес стратегіях. Основною проблемою рендерингу, на суб'єктивну думку, є саме процес оптимізації використання ресурсів серверу та клієнту, тобто необхідно створити такий метод який збільшить швидкість самого додатку(це можливо завдяки пре-рендеригу на стороні серверу), та при цьому збереже максимальну динамічність цього ж додатку. Інноваційний метод відображення Isomorphic Rendering якраз і вирішує питання поєднання цих двох аспектів, проте і в нього ще є свої недоліки, наприклад, під час рендерингу сторінки коли фреймворк на якому написаний сайт перетворить статичну сторінку в динамічну. Вирішенням даної проблеми є перетворення елементів сторінки в динамічні лише які зараз знаходяться на екрані користувача. Ця та інші проблеми приводять до висновку що і надалі небхідний розвиток та розробка нових методів візуалізації зображень на веб-сайтах.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. Ching S. React: Up & Running: Building Web Applications. 2020. P. 238.
2. Sbarski P. Serverless Architectures on AWS: With examples using AWS Lambda. 2017. P. 240.
3. Verma K. Next.js Quick Start Guide: Server-side rendering done right. 2021. P. 192.
4. Твердохліб А.О., Коротін Д.С. Ефективність функціонування комп'ютерних систем при використанні технології блокчейн і баз данних. *Таврійський науковий вісник. Серія: Технічні науки*, 2022. № 6.
5. Abdullah I. React Server-Side Rendering. 2020. P. 274.
6. Valverde A. Full Stack React: The Complete Guide to ReactJS and Friends. P. 691.
7. Robinson L. A Complete Guide To Incremental Static Regeneration (ISR) With Next.js. 2021. P. 154.
8. Skuratsivska O. Improving page speed of the isomorphic Vue.js application. 2021. P. 224.
9. Samaranayake M. A Step-By-Step Guide to Server-Side Rendering with VueJS. 2022. P. 240.

10. Цвик О.С. Аналіз і особливості програмного забезпечення для контролю трафіку. *Вісник Хмельницького національного університету. Серія: Технічні науки*, 2023. № 1.

11. Malcolm A. Client-Side Rendering vs Server-Side Rendering vs Static-Site Generation. 2020. P. 176.

12. Fayock C. What is Static Site Generation? How Next.js Uses SSG for Dynamic Web App. 2020.

13. Новіченко Є.О. Актуальні засади створення алгоритмів обробки інформації для логістичних центрів. Таврійський науковий вісник. Серія: Технічні науки, 2023. № 1.

14. Grigoryan A. The Benefits of Server Side Rendering Over Client Side Rendering. 2017. P. 188.

15. Breux G. Client-side vs. Server-side vs. Pre-rendering for Web Apps. 2022. P. 168.

16. Miller J., Osmani A. Rendering on the Web. 2022. P. 288.

17. Cocca J. Rendering Patterns for Web Apps – Server-Side, Client-Side, and SSG Explained. 2023. P. 198.

18. Swastik Y. Web rendering patterns in a nutshell. 2022. P. 168.

19. Prashant M. Server Side Rendering (SSR) vs. Client Side Rendering (CSR) vs. Pre-Rendering using Static Site Generators (SSG) and client-side hydration. 2021. P. 211.

20. Gathony M. Understanding Next.js Rendering Methods: CSR, SSR, SSG, ISR. 2022. P. 223.

REFERENCES:

1. Ching S. (2020). React: Up & Running: Building Web Applications. P. 238.

2. Sbarski P. (2017). Serverless Architectures on AWS: With examples using AWS Lambda. P.240.

3. Verma K. (2021). Next.js Quick Start Guide: Server-side rendering done right. P. 192.

4. Tverdokhlib A.O., Korotin D.S. (2022). Efektyvnist funktsionuvannia kompiuternykh system pry vykorystanni tekhnolohii blokchein i baz dannykh. Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky, (6).

5. Abdullah I. (2020). React Server-Side Rendering. P. 274.

6. Valverde A. Full Stack React: The Complete Guide to ReactJS and Friends. P. 691.

7. Robinson L. (2021). A Complete Guide To Incremental Static Regeneration (ISR) With Next.js. P. 154.

8. Skuratsivska O. (2021). Improving page speed of the isomorphic Vue.js application. P. 224.

9. Samaranayake M. (2022). A Step-By-Step Guide to Server-Side Rendering with VueJS. P.240.

10. Tsvyk O.S. (2023). Analiz i osoblyvosti prohramnoho zabezpechennia dlia kontroliu trafiku. Visnyk Khmelnytskoho natsionalnoho universytetu. Ceriia: Tekhnichni nauky, (1).

11. Malcolm A. (2020). Client-Side Rendering vs Server-Side Rendering vs Static-Site Generation. P. 176.

12. Fayock C. (2020). What is Static Site Generation? How Next.js Uses SSG for Dynamic Web App.

13. Novichenko Ye.O. (2023). Aktualni zasady stvorennia alhorytmiv obrobky informatsii dlia lohistychnykh tsentriv. Tavriiskyi naukovyi visnyk. Seriia: Tekhnichni nauky, (1).

14. Grigoryan A. (2017). The Benefits of Server Side Rendering Over Client Side Rendering. P. 188.

15. Breux G. (2022). Client-side vs. Server-side vs. Pre-rendering for Web Apps. 2022. P. 168.

16. Miller J., Osmani A. (2022). Rendering on the Web. P. 288.
17. Cocca J. (2023). Rendering Patterns for Web Apps – Server-Side, Client-Side, and SSG Explained. P. 198.
18. Swastik Y. (2022). Web rendering patterns in a nutshell. P. 168.
19. Prashant M. (2021). Server Side Rendering (SSR) vs. Client Side Rendering (CSR) vs. Pre-Rendering using Static Site Generators (SSG) and client-side hydration. P. 211.
20. Gathony M. (2022). Understanding Next.js Rendering Methods: CSR, SSR, SSG, ISR. P. 223.