Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра_____Програмної інженерії_____
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти - другий (магістерський)

Дослідження методів структурної оптимізації веб-систем
(тема)

Виконала: студентка 2 курсу, групи ПЗСм-18-1
_____Чурсіна Т. С._____
(прізвище, ініціали)

спеціальності 121- Інженерія програмного забезпечення_
(код і повна назва спеціальності)

Освітньо-професійної програми____
(тип програми)

Програмне забезпечення систем_____
(повна назва освітньої програми)

Керівник _____доц. Назаров О. С,_____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри, проф. _____ З.В.Дудар

2019 р.

Харківський національний університет радіоелектроніки

Факультет_Комп'ютерних наук_____

Кафедра_Програмної інженерії_____

Рівень вищої освіти - другий (магістерський)

Спеціальність _121-Інженерія програмного забезпечення_____
(код і повна назва)

Тип програми освітньо-професійна програма

Освітня програма_ Програмне забезпечення систем_____

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 20 ___ р.

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові _Чурсіній Тетяні Сергіївні_____
(прізвище, ім'я, по батькові)

1. Тема роботи _ Дослідження методів структурної оптимізації веб-систем
затверджена наказом університету від "31" жовтня 2019 р № 1610 Ст
2. Термін подання студентом роботи до екзаменаційної комісії
17 грудня 2019 р.
3. Вихідні дані до роботи *фактори структурної оптимізації веб-систем, рівняння часу завантаження веб-сторінки, пояснювальна записка. Використовувати ОС Ubuntu, середовище об'єктно-орієнтованого проектування, хостинг для розгортання системи.*
4. Перелік питань, що потрібно опрацювати в роботі *мета роботи, аналіз стану розв'язання проблеми і постановка задачі, огляд і вибір методів дослідження, виділення досліджуваних факторів оптимізації , розробка експериментальної веб-системи, дослідження ступеню впливу факторів оптимізації, побудова рівняння часу завантаження веб-сторінки.*

5 Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім’я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Дослідження факторів оптимізації веб-системи	доц. Назаров О. С.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка *
1.	Аналіз стану розв’язання проблеми	20 жовтня 2019р.	
2.	Огляд і вибір методів дослідження	27 жовтня 2019р.	
3.	Розробка експериментальної веб-системи	02 листопада 2019р.	
4	Дослідження факторів оптимізації веб-системи	20 листопада 2019р.	
5.	Підготовка пояснювальної записки	30 листопада 2019р.	
6.	Підготовка презентації та доповіді	03 грудня 2019р.	
7.	Попередній захист	05 грудня 2019р.	
8.	Нормоконтроль, рецензування	11 грудня 2019р.	
9.	Занесення диплома в електронний архів	13 грудня 2019р.	
10.	Допуск до захисту у зав. кафедри	13 грудня 2019р.	
* заповнюється вручну після виконання чергового пункту			

Дата видачі завдання _____ _____2019 р.

Студент _____
(підпис)

Керівник роботи _____ __доц. Назаров О. С._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Атестаційна робота магістра містить: 61с., 23 рис.,18 табл., 18 джер.

GOOGLE PAGE SPEED INSIGHTS, LARAVEL, PHP, ВЕБ-СИСТЕМА, РЕГРЕСІЙНИЙ АНАЛІЗ, ФАКТОРИ ОПТИМІЗАЦІЇ, ШВИДКІСТЬ ЗАВАНТАЖЕННЯ СТОРІНКИ.

Метою роботи є дослідження вагомості факторів впливу на час завантаження сторінки веб-системи.

Методи розробки базуються на технології Laravel на мові програмування PHP. Методами дослідження є експерименти, вимірювання, порівняння та математичне моделювання.

У результаті дослідження було встановлено ступінь впливу кожного з обраних факторів оптимізації на час завантаження сторінки розробленої експериментальної веб-системи.

GOOGLE PAGE SPEED INSIGHTS, FACTORS OF OPTIMIZATION, LARAVEL, PHP, REGRESSION, WEB-SYSTEM, WEB PAGE LOAD TIME.

The purpose of this work is to investigate the weight of factors influencing the load time of a web page.

Development methods are based on Laravel Framework in PHP programming language. Research methods are experiments, measurements, comparisons and mathematical modeling.

The research found the degree of influence of each of the selected optimization factors on the load time of the page for the developed experimental web system.

ЗМІСТ

Вступ 6
1 Аналіз стану розв'язання проблеми 8
2 Огляд і вибір методів дослідження 10
2.1 Огляд методів наукового дослідження 10
2.2 Методи емпіричного дослідження 11
2.3 Математичне моделювання 12
2.4 Вибір методів дослідження 13
3 Аналіз проведених теоретичних досліджень 14
3.1 Аналіз інструменту Google PageSpeed Insights 14
3.2 Виділення досліджуваних факторів 15
3.3 Математична модель часу завантаження веб-сторінки 17
4 Аналіз результатів практичних досліджень 20
4.1 Опис розробленої експериментальної системи 20
4.2 Опис розробленого інструменту 24
4.3 Опис проведених експериментів 25
Висновки 42
Перелік джерел посилання 43
Додаток А Слайди презентації 45
Додаток Б Апробація результатів роботи 52
Додаток В Фрагмент оптимізованого програмного коду 55
Додаток Г Відгук та рецензії 59

ВСТУП

Одним із найпопулярніших напрямків розробки програмного забезпечення на сьогоднішній день є розробка веб-орієнтованих систем. Серед факторів, що визначають популярність веб-додатку, одними з найважливіших є швидкість завантаження його сторінок, а також швидкість реакції сторінки на дії користувача.

Так, згідно досліджень компанії Google, якщо сайт завантажується більше трьох секунд, 53% його покинуть. У 2010 Google анонсував: швидкість сайту - фактор, який буде враховуватися при пошуковому ранжуванні. Одразу після анонса Google представив спеціальні інструменти для веб-розробників, які допоможуть аналізувати і збільшувати швидкість завантаження веб-сторінок [1].

Більшість дослідників погоджуються, що оптимальний час завантаження веб-сторінки - від 1,5 до 3 секунд. Якщо вона завантажується довше, то більше половини користувачів залишають її, так і не дочекавшись [2]. Тому проблема пошуку методів оптимізації швидкості завантаження веб-сторінок є актуальною на сьогоднішній день.

Метою даної роботи є знаходження факторів організації структури та програмного коду веб-системи, що впливають на швидкість процесу повного завантаження веб-сторінки, та визначення вагомості впливу кожного з них на цей процес. Для досягнення поставленої мети необхідно виконати такі задачі:

– здійснити огляд існуючих публікацій на дану тему;

– визначити недоліки відомих рекомендації щодо оптимізації швидкості завантаження веб-сторінки;

– на основі вивченої літератури виділити фактори впливу;

– реалізувати веб-систему, що досліджуватиметься;

– експериментально дослідити вплив кожного фактору та їх комбінацій на швидкість завантаження сторінки реалізованої системи;

– виконати розрахунки коефіцієнтів впливу кожного фактора за допомогою методів математичного моделювання;

– зробити висновки щодо розрахунків та адекватності моделі.

Об’єктом дослідження є розроблена експериментальна веб-система, а саме – її найбільш навантажена сторінка.

Предметом дослідження є час завантаження веб-сторінки.

Для дослідження були використані емпіричні та емпірико-теоретичні методи наукових досліджень, а саме:

– було проведено набір експериментів з урахуванням різних факторів впливу та їх комбінацій, багаторазово виміряний час завантаження веб-сторінки протягом кожного експерименту та порівняно результати вимірювань (емпіричні методи);

– обчислено коефіцієнт впливу кожного фактору за допомогою математичного моделювання (емпірико-теоретичний метод).

Наукова новизна даної роботи полягає в тому, що були виділені способи оптимізації саме з точки зору програмного коду і структури системи та визначена їх вагомість. Більшість існуючої літератури описує специфічні методи, що пов’язані с оптимізацією структури бази даних або з кешуванням даних та не є універсальними. Також у джерелах не було знайдено інформації щодо ступеню впливу використання того чи іншого методу на швидкість завантаження веб-сторінки. Саме ця інформація і є одним з результатів даного дослідження.

Результати даного дослідження є корисними для веб-розробників, оскільки, як вже зазначалось вище, для кожної веб-системи час завантаження веб-сторінки є одним із основних показників, що можуть впливати на її популярність та конверсію.

Стислі положення результатів даного дослідження було опубліковано в матеріалах конференції MEICS-2019 (одна публікація).

1 АНАЛІЗ СТАНУ РОЗВ'ЯЗАННЯ ПРОБЛЕМИ

Навіть зараз, коли швидкість інтернету у більшості провайдерів перевищує 100 Мбіт/сек, а мобільні оператори розвивають 5G технології, питання про швидкість завантаження сайту залишається актуальним. Цей параметр істотно впливає на конверсію, показник відмов, відвідуваність і інші важливі для бізнесу характеристики [3].

З користувацької точки зору необхідно, щоб сайт працював швидко, оскільки це забезпечує економію часу та привносить позитивний досвід. У компанії Google провели дослідження за допомогою нейронної мережі, яку навчили поведінці користувачем на великих обсягах даних, з метою передбачення конверсій та показників відмов. У результаті виявилось, що у разі часу завантаження сторінки від 1 до 10 секунд ймовірність відмов збільшується на 123%. Навіть у разі затримки до 3 секунд вона зростає на третину[4].

Дослідження впливу швидкості завантаження сайтів на утримання користувачів проводились також іншими компаніями-гігантами з різних галузей. Так, за даними компанії Akamai, що є провідним постачальником послуг для акселерації веб-сайтів, 40% відвідувачів залишають сайт, якщо час його завантаження перевищує 3 секунди. А дослідження компанії Walmart, що є один із найбільших ритейлерів світу, встановило, що у інтернет-магазину є лише 0,25 секунди, щоб утримати користувача, інакше він перейде до сайту конкурентів. Компанія Amazon, яка є основним конкурентом Walmart, у своєму дослідженні з'ясувала, що сповільнення завантаження сторінок сайту на одну секунду призводить до втрати конверсії на 1%, що еквівалентно втраті $1,6 млрд в межах річного прибутку компанії [4].

Іншим важливим є те, що швидкість завантаження сайту є одним із факторів ранжування. Співробітники компанії Google підтверджують, що тривалий час завантаження негативно впливає на позиції у видачі пошукових результатів.

Підсумовуючи відомі результаті досліджень, думки спеціалістів сходяться відносно того, що оптимальний час завантаження сайту – 2-3 секунди. Ідеальним часом є час реакції користувача, тобто 0,5 секунд [5]. Проте для

високонавантажених веб-додатків оптимальна швидкість може відрізнятися. У такому випадку слід ураховувати середній час завантаження додатків конкурентів у конкретній області та орієнтуватись на нього.

Існує набір різноманітних рекомендацій стосовно оптимізації швидкості завантаження сайтів. Найрозповсюдженішими є рекомендації щодо реалізації кешування, проте цей метод не є універсальним. Основним недоліком підходу кешування є неможливість масштабування, тобто неможливість використання кешу на інших серверах, що є критичним для великих розподілених систем. Саме тому в даній роботі кешування досліджуватись не буде.

Іншою частою рекомендацією є оптимізація запитів до бази даних, а також вдосконалення фізичної структури бази даних. Ці аспекти також є темою для окремого дослідження.

Ефективним методом оптимізації веб-систем є удосконалення апаратної частини, наприклад, придбання та налаштування потужного серверу або набору серверів, використання більш надійного хостингу тощо. Проте у зв’язку зі специфікою діяльності було вирішено в даному дослідженні зупинитися на аспектах, що безпосередньо залежать від веб-розробника, а саме – на оптимізації коду програмної системи, як серверного, так і клієнтського.

Відомо, що більша частина часу завантаження сторінки витрачається саме на клієнтську частину. Серверні витраті зазвичай складають до 500 мс. Оскільки це значення швидкості реакції користувача, даний час не є помітним. Проте зовсім відкидати оптимізацію серверного коду є помилковим підходом, оскільки ступінь впливу кожного аспекту оптимізації на загальний час завантаження є невідомим, а відповідна інформація не була знайдена у джерелах. Тому в даному дослідженні пропонується виділити фактори оптимізації серверного та клієнтського коду і встановити вагомість кожного з них.

2 ОГЛЯД І ВИБІР МЕТОДІВ ДОСЛІДЖЕННЯ

2.1 Огляд методів наукового дослідження

Термін «метод» походить від грецького слова methods, що означає шлях, спосіб пізнання, дослідження, простежування. Метод – це сукупність прийомів і операцій пізнання і практичного перетворення дійсності; спосіб досягнення певних результатів у пізнанні і практиці [6].

Метод дослідження, або метод пізнання – це спосіб досягнення мети у науковій діяльності. Наука, що вивчає дані методі, називається методологією [6].

Незалежно від типу науково-пізнавальної діяльності, в основі будь-якого наукового методу лежать три основні принципи - об'єктивність, систематичність і відтворюваність. Об'єктивність передбачає запобігання впливу суб’єктивних уявлень на процес наукового пізнання. Під систематичністю мається на увазі впорядкованість науково-пізнавальної діяльності, тобто процес наукового пізнання виконується системним, впорядкованим чином. Відтворюваність передбачає можливість повторити (відтворити) усі етапи наукового процесу під керівництвом інших дослідників, отримавши подібні, несуперечливі результати, і тим самим перевіривши їх достовірність. Якщо результати не відображаються, то вони ненадійні і, отже, не можуть вважатися достовірними [7].

Методи дослідження можна класифікувати за наступники ознаками:

– за рівнем пізнання – емпіричні та теоретичні;

– за точністю припущень – детерміністичні та ймовірнісно-статистичні;

– за здійснюваними у пізнанні функціями – методи систематизації, пояснення і прогнозування;

– за областю дослідження – фізичні, соціальні, біологічні і т.д. [6].

У структурі загальнонаукових методів і прийомів найчастіше виділяють три рівні:

– методи емпіричного дослідження;

– методи теоретичного дослідження;

– загальні (використовуються як на емпіричному, так і на теоретичному рівні дослідження: аналіз, синтез, індукція, дедукція, моделювання).

До методів емпіричного дослідження належать спостереження, порівняння, вимірювання та експеримент.

Методи теоретичного дослідження містять у собі такі методи, як ідеалізація, формалізація, сходження від абстрактного до конкретного, уявний експеримент та аксіоматичний метод [6].

2.2 Методи емпіричного дослідження

Емпіричні методи пізнання відіграють велику роль в науковому дослідженні. Вони не лише є основою для підкріплення теоретичних передумов, але часто складають предмет нового відкриття, наукового дослідження [7].

Головною метою емпіричного пізнання є отримання даних спостереження і формування фактів науки, на основі яких потім будується емпіричний базис наукового знання і розвивається система теоретичних побудов. Таким чином, емпіричне дослідження здійснюється на базі практичного оперування з об'єктами, виключає безпосереднє спостереження і первинну логічну обробку даних спостереження. В результаті всіх цих процедур з'являються наукові факти [7].

До методів емпіричного дослідження належать такі методи: спостереження, порівняння, експеримент, вимірювання [6].

Спостереження – це вивчення предметів або явищ, що спирається здебільшого на дані органів чуття. Кінцева мета даного процесу – отримати знання про суттєві властивості об’єкту пізнання. Спостереження безпосередньо пов’язано з вимірюванням, яке є процесом знаходження відношення даної величини до іншої однорідної величини, що є прийнятою за одиницю вимірювання. Результат вимірювання відображається числом [7].

Порівняння – це пізнавальна операція, що лежить в основі суджень про схожість або різницю між об’єктами. За допомогою порівняння визначаються якісні та кількісні характеристики предметів [7].

Експеримент – це активне та цілеспрямоване втручання в хід процесу, що вивчається, відповідні зміни об’єкту або його відтворення у спеціально створених та контрольованих умовах. У ході експерименту об’єкт, що вивчається,

ізолюється від впливу побічних обставин і представляється у «чистому вигляді» [7].

2.3 Математичне моделювання

Термін «модель» походить від латинського слова «modulus», що перекладається як міра, взірець, норма. Модель – це форма відображення певного фрагменту дійсності (предмета, явища. процесу чи ситуації), що містить значну кількість властивостей об’єкта, що моделюється, і може буди представлена в абстрактній або матеріальній формі. Модель створюється з метою одержання та збереження інформації про об’єкт-оригінал. Метод дослідження об’єктів пізнання шляхом побудови, дослідження та використання їх моделей називається моделюванням [8].

Математичною моделлю називається наближений опис будь-якого класу явищ, об'єктів зовнішнього світу, виражений допомогою математичних понять і символіки [9].

Математична модель є формалізованим описом системи, що дозволяє вивчати її математичними методами. Зазвичай вона складається з сукупності співвідношень (рівнянь, нерівностей, логічних умов, формул і т.д.), що визначають характеристики станів системи в залежності від її параметрів, вхідних сигналів, початкових і граничних умов, часу та ін. Формалізована схема реальної системи дозволяє розкласти її на найпростіші елементи. Процеси, що відбуваються в цих елементах, як і взаємозв'язок між ними, спрощуються.

Для побудови математичної моделі реальної системи всі наявні у вигляді таблиць або графіків відомості записують у вигляді відповідних математичних виразів [9].

Математична модель в силу спрощення процесів, що відбуваються в елементах системи, не завжди повністю відповідає (адекватна) їй. Але навіть у цьому випадку кількісні дослідження математичної моделі дозволяють отримати якісний опис реальної системи [9].

2.4 Вибір методів дослідження

Для досягнення мети даного дослідження необхідно виконати наступні кроки:

– реалізувати програмну системи та спеціальний її модуль, що вимірює повний час завантаження веб-сторінки;

– вибрати найбільш навантажену сторінку за кількістю запитів до бази даних, а також складністю серверного та клієнтського коду;

– виміряти час завантаження сторінки без додаткових оптимізаційних змін системи, записати результат;

– вносячи різні комбінації оптимізаційних змін до коду або структури системи, які буде детально описано у наступних розділах, виміряти час завантаження веб-сторінки для кожної з комбінацій та зберегти результати кожного такого експерименту до бази даних;

– зібравши результати експериментів до таблиці, обчислити ступінь впливу кожного фактору оптимізації на загальний час завантаження сторінки за допомогою регресійного аналізу.

Таким чином, були обрані наступні методи дослідження: експеримент, вимірювання, порівняння та математичне моделювання.

3 АНАЛІЗ ПРОВЕДЕНИХ ТЕОРЕТИЧНИХ ДОСЛІДЖЕНЬ

3.1 Аналіз інструменту Google PageSpeed Insights

Одним із найпопулярніших інструментів перевірки швидкості завантаження веб-сторінки є Google PageSpeed Insights (далі – PSI). Цей безкоштовний онлайн-сервіс, розроблений компанією Google, дозволяє отримувати звіти про швидкість завантаження сторінок як на персональних комп'ютерах, так і на мобільних пристроях, а також поради, як цю швидкість збільшити [10].

Інструмент PSI за допомогою технології Lighthouse отримує різноманітні показники швидкості завантаження сторінки, що знаходиться за вказаним URL, такі як перша промальовка контенту, час завантаження достатньої кількості контенту, індекс швидкості завантаження, час закінчення роботи центрального процесору (ЦП), час завантаження для взаємодії та приблизний час затримки при вводі. Кожному показнику надається оцінка [10].

Перша промальовка контенту – це показник, що визначає інтервал часу між початком завантаження сторінки та появою першого зображення або блоку тексту.

Перша значна промальовка, або промальовка достатньої кількості контенту – показник, що визначає інтервал часу між початком завантаження сторінки та появою основного контенту.

Індекс швидкості завантаження відображає, наскільки швидко на сторінці з'являється контент.

Час закінчення роботи ЦП – час, коли на сторінці стає можливою обробка користувацького вводу.

Час завантаження для взаємодії – це час, протягом якого сторінка стає повністю готовою до взаємодії з користувачем.

PSI надає дані як про те, наскільки швидко сторінка завантажувалась у реальних користувачів, так і дані, отримані в результаті імітації процесу завантаження. Оскільки імітація виконується в керованих умовах, за її допомогою зручно виявляти та усувати проблеми зі швидкістю, але є ризик пропустити деякі з тих, що виникають у реальних умовах. Дані спостережень від реальних

користувачів відображають реальну ситуацію, але набір доступних показників є обмеженим [11].

Ураховуючи вищезгадані недоліки сервісу PSI, можна зробити висновок, що покладатися лише на нього протягом дослідження не є доцільним. Тому було вирішено інтегрувати у розроблену досліджувану систему власний інструмент, що обчислює час повного завантаження веб-сторінки, а за допомогою PSI порівняти показники найменш оптимального (без оптимізаційних змін) та найбільш оптимального (коли активні всі оптимізаційні зміни) випадків завантаження сторінки.

3.2 Виділення досліджуваних факторів

Швидкість завантаження веб-сторінки тісно пов’язана з етапами обробки сторінки у браузері. Спрощено етапи завантаження сторінки виглядають наступним чином:

– введення адреси сторінки у браузері та обробка DNS-запиту до сервера, на якому розміщено сайт;

– обробка HTTP-переадресацій під час завантаження сторінки;

– підключення браузера до серверу;

– передача даних від HTTP-сервера браузеру;

– обробка завантаженої інформації браузером та промальовка елементів;

– повне завантаження сторінки з усіма скриптами, зображеннями, стилями [2].

Проблеми з завантаженням сторінки можуть виникати на будь-якому з цих етапів. Проте у даному дослідженні не розглядається фактори потужності й конфігурації сервера та швидкості інтернет-з’єднання, оскільки вони не залежать безпосередньо від веб-розробника. Також, як вже зазначалося, не розглядається оптимізація структури бази даних (далі – БД). Вважається, що досліджується випадок, коли веб-система взаємодіє з готовою реляційною БД, на зміну структури якої у розробника немає прав, тому досліджується оптимізація саме програмного коду системи – як серверної частини, так і клієнтської.

Переважна більшість фреймворків серверних мов програмування мають готову реалізацію взаємодії з БД. Найчастіше розробнику не потрібно власноруч писати SQL запити, оскільки фреймворк реалізує технологію Object-Relational Mapping (ORM). ORM є технологією програмування, що пов'язує БД з концепціями об'єктно-орієнтованих мов програмування. Тобто це принцип, згідно з яким робота з БД здійснюється в коді на рівні об'єктів наступним чином:

– таблицям БД відповідають окремі класи;

– у класах описані властивості об'єктів, кожна з яких відповідає полю в таблиці;

– робота виконується з об'єктами таких класів, а кожен об'єкт відповідає одному запису в БД [12].

З одного боку, такий підхід має очевидні переваги, такі як незалежність від БД, наочне моделювання предметної області, зменшення кількості коду та зручність використання. А з іншого боку, реалізація ORM потребує додаткових витрат пам'яті, більшого навантаження на процесор сервера, а отже, і часу виконання серверного коду. Альтернативою ORM є використання прямих SQL запитів до БД, що можна виділити як перший фактор оптимізації веб-системи.

Хоча виконання серверного коду займає деякий час, саме виконання клієнтського коду з подальшим відображенням сторінки у браузері займає більшу частину часу повного завантаження веб-сторінки [13]. Варто зазначити, що на час виконання клієнтської частини впливає не тільки код, але й розмір підключених до сторінки файлів На сьогоднішній день важко уявити сайт без жодного зображення. Очевидно, що час завантаження та промальовки зображення є прямо пропорційним об'єму дискового простору, що воно займає на сервері. Для оптимізації завантаження зображень компанія Google розробила у 2010 році формат графіки WebP. На відміну від інших форматів, WebP підтримує стиснення зображень до 25% без втрат якості [14]. Тому як другий фактор оптимізації швидкості завантаження сторінки було обрано використання WebP зображень замість аналогічних у форматі JPG.

Сучасні сайти важко уявити без стилізації та інтерактивності, тому в кожній веб-системі завжди наявне підключення набору CSS-стилей та JS-скриптів. Коди CSS та JS є доволі громіздкими. Зробити ці файли менш об'ємним можливо шляхом видалення зайвих символів – пробілів, коментарів, переносів рядків. Цей підхід називається мініфікацією, а для його реалізації існують прості онлайн-сервіси, а також вбудований функціонал фреймворків. Отже, наступним фактором оптимізації було виділено мініфікація CSS та JS.

У продовження теми скриптів варто зазначити, що їх розташування при підключенні на сторінці також відіграє важливу роль у процесі завантаження. Громіздкі скрипти, що знаходяться у верхній частині сторінки, можуть блокувати її відображення, тому їх підключення краще перенести донизу, перед закриттям тегу <body>, якщо це дозволяє специфіка виконання коду сторінки. Підключення скриптів у нижній частині сторінки є ще одним фактором оптимізації.

Отже, було визначено такі фактори оптимізації часу завантаження веб-сторінки:

– використання прямих SQL запитів для роботи з БД замість взаємодії за допомогою ORM;

– використання стиснених без втрати якості зображень у форматі WebP замість JPG;

– мініфікація CSS та JS файлів;

– підключення скриптів перед закриваючим тегом </body> замість підключення перед закриваючим тегом </head> (у верхній частині розмітки сторінки).

3.3 Математична модель часу завантаження веб-сторінки

Виділення факторів оптимізації часу завантаження веб-сторінки надає змогу звести задачу до дослідження вагомості кожного з них, тобто знаходження вагового коефіцієнту кожного фактору. Таким чином, швидкість завантаження сторінки можна описати рівнянням лінійної множинної регресії.

Термін "регресія" був введений Френсісом Гальтон в кінці 19-го століття. Гальтон виявив, що діти батьків з високим або низьким ростом зазвичай не успадковують видатний зростання і назвав цей феномен "регресія до посередності". Спочатку цей термін використовувався виключно в біологічному сенсі. Після робіт Карла Пірсона цей термін стали використовувати і в статистиці [15].

Регресійний аналіз – це метод моделювання вимірюваних даних і вивчення їх властивостей. Дані складаються з пар значень залежної змінної (змінної відгуку) та незалежної змінної (пояснюючої змінної). Регресійна модель є функцією незалежної змінної та параметрів с доданням випадкової величини. Параметри моделі налаштовуються таким чином, що модель якнайкраще наближує дані [15].

Критерієм якості наближення (цільовою функцією) зазвичай є середньоквадратична похибка: сума квадратів різниці значень моделі і залежної змінної для всіх значень незалежної змінної в якості аргументу. Передбачається, що залежна змінна є сумою значень деякої моделі та випадкової величини. Відносно характеру розподілу цієї величини робляться припущення, що називаються гіпотезою про породження даних. Для підтвердження або спростування цієї гіпотези виконуються статистичні тести. При цьому передбачається, що незалежна змінна не містить помилок. Регресійний аналіз використовується для прогнозу, аналізу часових рядів, тестування гіпотез та виявлення прихованих взаємозв’язків у даних.[15]

Лінійна регресія – це метод відновлення залежності між двома змінними, при якому залежність описується лінійною функцією.

Множинною називають лінійну регресію, у моделі якої число незалежних змінних дві або більше. Це виражена у вигляді прямої залежність середнього значення залежної величини від двох або більше незалежних [15].

Завданням множинної лінійної регресії є побудова лінійної моделі зв'язку між набором безперервних предикторів і безперервною залежною змінною.

Найбільш часто використовуване рівняння регресії представлено формулою (1):

$$Y = \sum_{i=1}^{n} a_i x_i + b_0 \quad (1)$$

де Y – залежна величина,

a_i – вагові коефіцієнти,

x_i – фактори впливу,

b_0 – вільний член[16].

У контексті даної роботи залежною величиною є час завантаження веб-сторінки, x_i – це вищезазначені фактори оптимізації (прийматимуть значення 0 або 1, тобто використовується або ні), a_i – це вагові коефіцієнти, які необхідно знайти, b_0 – час, за який відбувається завантаження сторінки, якщо усі змінні дорівнюють нулю, тобто коли відсутні всі фактори оптимізації.

4 АНАЛІЗ РЕЗУЛЬТАТІВ ПРАКТИЧНИХ ДОСЛІДЖЕНЬ

4.1 Опис розробленої експериментальної системи

Серед веб-систем з достатнім рівнем навантаження найпопулярнішими є інтернет-магазини. Вони передбачають постійну роботу з БД з великою кількістю запитів, а також значну кількість елементів на сторінці. Тому дослідження було вирішено здійснити на фрагменті розробленого інтернет-магазину для продажу товарів ручної роботи (хендмейду), далі – MyHandmade.

Для реалізації системи було обрано наступні технології:

– серверна частина – PHP фреймворк Laravel 6;

– система керування базами даних (СКБД) – MySQL 5;

– CSS фреймворк – MaterializeCSS;

– JS бібліотеки – JQuery, Materialize, NoUISlider, wNumb.

Мова програмування PHP була обрана через високу поширеність у використанні для подібних рішень. Laravel – безкоштовний веб-фреймворк з відкритим вихідним кодом. Його описують як фреймворк з «виразним та елегантним синтаксисом» [17]. Він містить готові реалізації багатьох типових задач, таких як аутентифікація, маршрутизація та сесії. Основними перевагами Laravel є:

– велика екосистема зі швидким розгортанням своєї платформи;

– детальна якісна документація;

– власний шаблонізатор Blade;

– логічний та зрозумілий синтаксис;

– власна ORM Eloquent [17].

В основі архітектури фреймворка Laravel лежить архітектурний патерн Model-View-Controller (MVC). Цей патерн розподіляє работу веб-системи на три окремих функціональних ролі:

– модель даних (model) – забезпечує взаємодію з БД;

– користувацький інтерфейс (view) – отримує дані та відображає їх користувачеві;

– контролер (controller) – сповіщає модель про необхідність змін відповідно до дій користувача [17].

На рисунку 1 представлена схема архітектури MVC фреймворка Laravel.

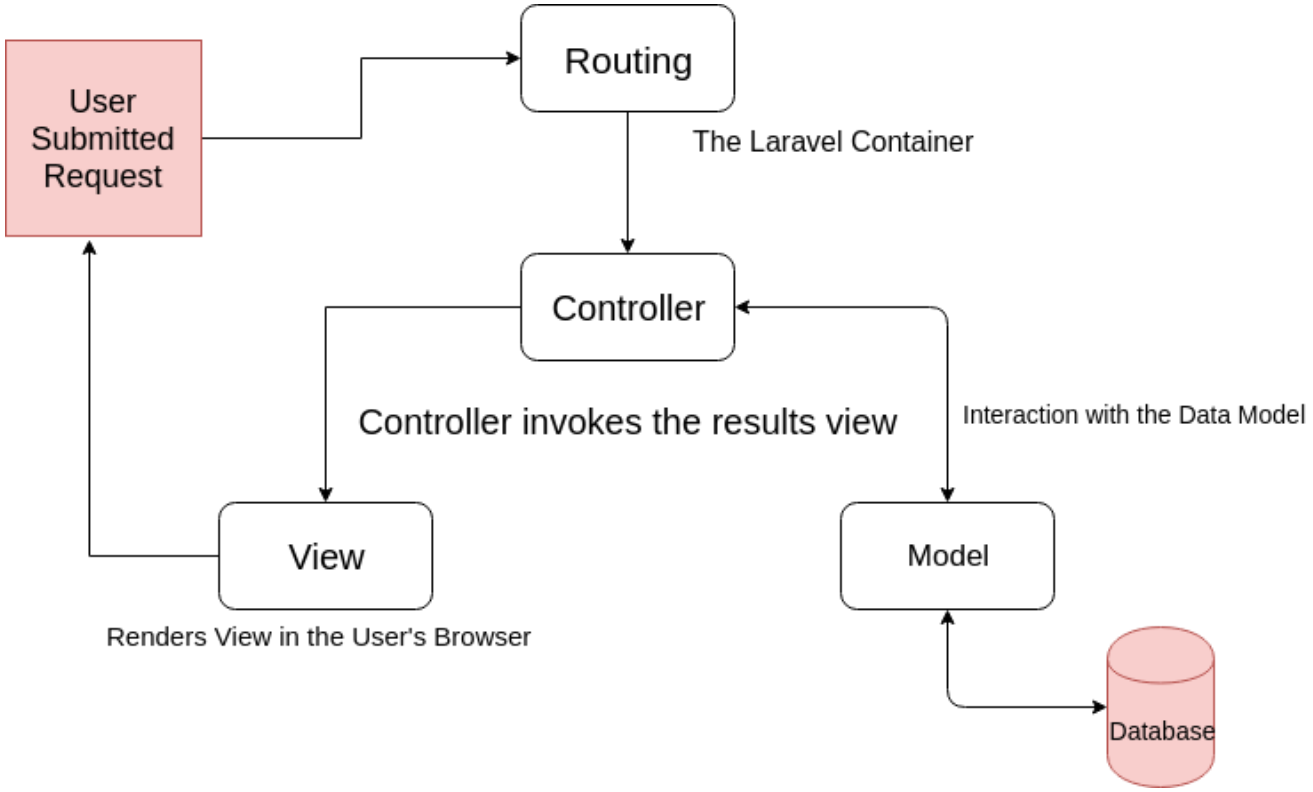

Рисунок 1 – Схема архітектури та взаємодії компонентів Laravel

Для дослідження була обрана найбільш навантажена сторінка системи, а саме – сторінка, на якій здійснюється виведення товарів певної категорії з інформацією про них, фільтрів із кількістю товарів за кожним з них та пагінації. На рисунку 2 показана схема БД даного фрагменту системи.

Основними сутностями даного фрагменту є категорія (наприклад, біжутерія), продукт, тип характеристики продукту (наприклад, матеріал), характеристика продукту (наприклад, бісер, нитки для в'язання для типу «матеріал»).

Таблиця charprods не є самостійною сутністю, а забезпечує зв'язок між продуктами та характеристиками. Також на схемі присутня таблиця experiments, у яку для зручності записуються результати вимірів часу завантаження сторінки.

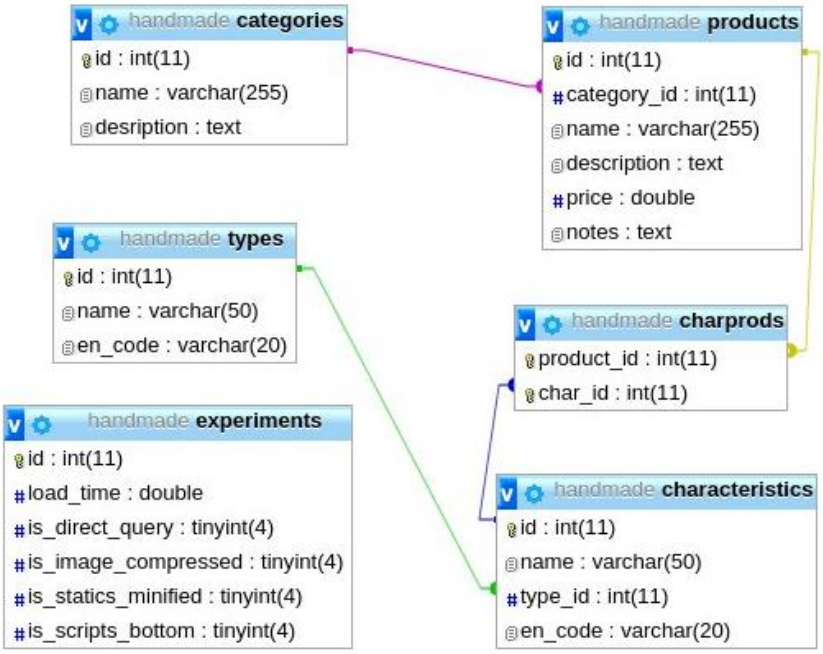

Рисунок 2 – Схема БД системи MyHandmade

Досліджувана сторінка показана на рисунку 3.

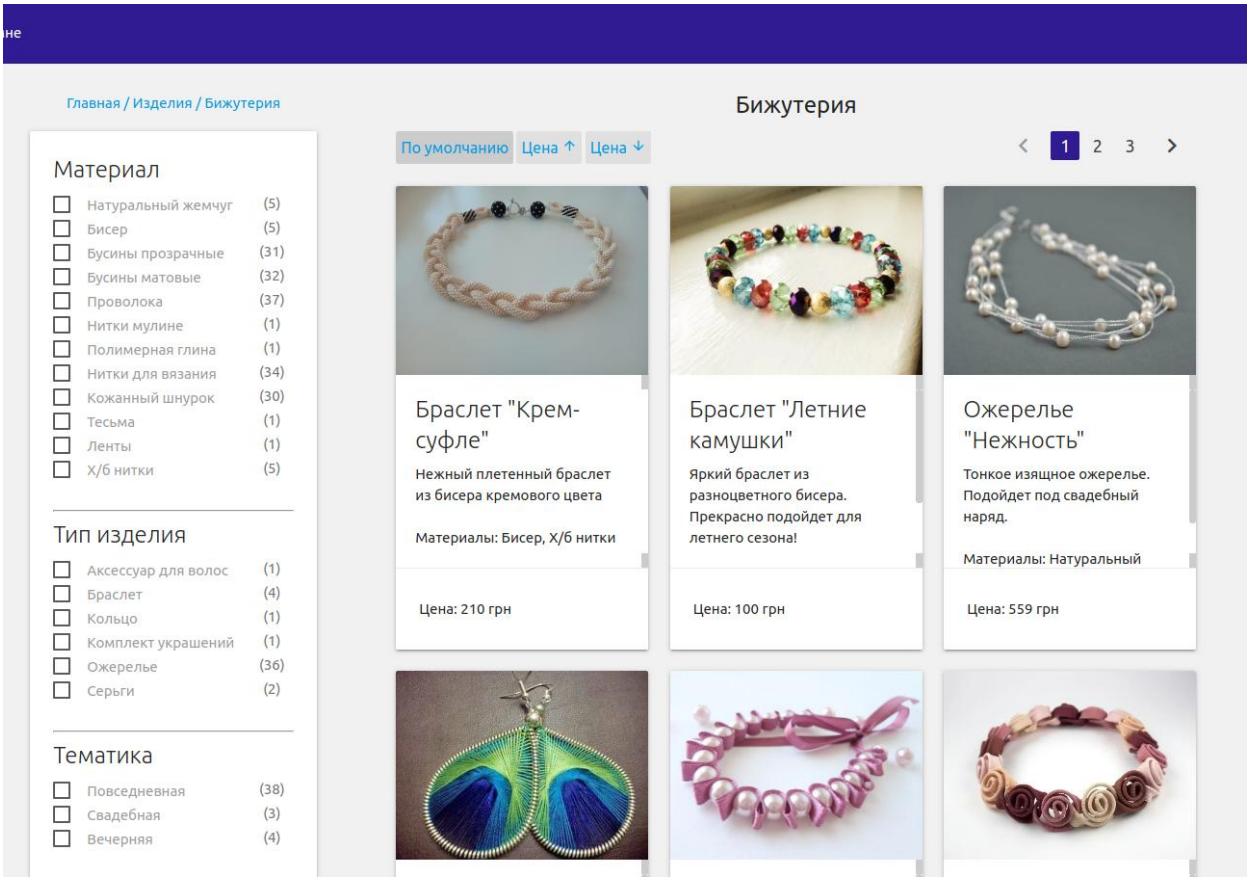

Рисунок 3 – Зовнішній вигляд досліджуваної сторінки

Після завантаження на сторінці виводиться 15 продуктів з їх характеристиками та зображеннями. Найскладнішими з точки зору часу виконання є запити на виведення фільтрів на товарів. Код, що відповідає за виведення фільтрів, наведений нижче.

```
private function getFilters($categoryId)
    {
        $filters = [];
        $characteristicsTypes = Type::all();

        foreach($characteristicsTypes as $type) {
            $tmp = [
                "title" => $type->name,
                "characteristics" => Characteristic::where('type_id',
$type->id)
                    ->join('charprods', 'characteristics.id', '=',
'charprods.char_id')
                    ->join('products', 'charprods.product_id', '=',
'products.id')
                    ->where('products.category_id', '=', $categoryId)
                    ->groupBy('characteristics.id')
                    ->selectRaw('count(distinct products.id) as
products_count, characteristics.id, characteristics.name,
characteristics.en_code')
                    ->get()
            ];

            $filters[] = $tmp;
        }

        return $filters;
    }
```

Як бачимо, запит, що виконується у циклі, має два оператори JOIN, що суттєво впливає на швидкість виконання. Надалі в ході експериментів даний код буде змінено для використання безпосередніх запитів до БД без ORM. Фрагмент оптимізованого коду наведено у додатку В.

Нижче наведено код методу, який збирає всю інформацію про товари разом з їх характеристиками.

```
public static function getProductsWithMaterials($categoryId)
    {
        $products = Product::where('category_id', $categoryId)->offset(0)-
>limit(self::PRODUCTS_PER_PAGE)->get();

        foreach($products as $product) {

            $materials = Type::where('types.name', '=', 'Материал')
```

```
                ->join('characteristics', 'characteristics.type_id', '=',
'types.id')
                ->join('charprods', 'characteristics.id', '=',
'charprods.char_id')
                ->where('charprods.product_id', '=', $product->id)
                ->selectRaw('GROUP_CONCAT(characteristics.name SEPARATOR ",
") as materials')
                ->first()->materials;

            $product->materials = $materials;
        }

        return $products;
    }
```

У даному коді також наявні два оператори JOIN у циклі. Вищезгадані методи дозволять продемонструвати помітну різницю у часі завантаження сторінки після переходу від ORM до безпосередніх SQL запитів до БД.

Для того, щоб відтворити реальні умови експерименту, було вирішено виконати розгортання системи на хостингу. Для цього було зареєстроване доменне ім’я myhandmade.site та куплений хостинг у системі «Хостинг Україна». Також придбання хостингу зробило можливими подальші тести з використанням сервісу Google PageSpeed Insights. Після проведення експериментів доступ до сайту був обмежений для запобігання появи реальних користувачів.

4.2 Опис розробленого інструменту

Як вже зазначалося, показники сервісу PSI не є надійними, оскільки він враховує обмежену кількість факторів. Для виміру часу повного завантаження сторінки був розроблений інструмент, що підраховує час виконання серверної частини й повного завантаження і виконання клієнтської, після чого знаходить їх суму.

Нижче наведений фрагмент коду, який додає на сторінку результат виміру часу виконання серверного коду.

```
$finish = microtime(true);
$loadTime = $finish - LARAVEL_START;

if($_SERVER['REQUEST_URI'] !== '/record')
{
    echo '<div id="loadTime" data-load="' . $loadTime . '"></div>';
    echo '<div id="backendFinish" data-load="' . $finish . '"></div>';
}
```

Протягом подальшого виконання клієнтського коду дані значення перехоплюються JS функцією, обчислюється час виконання клієнтської частини і знаходиться загальний час виконання.

Сторінка з активним інструментом наведена на рисунку 4.

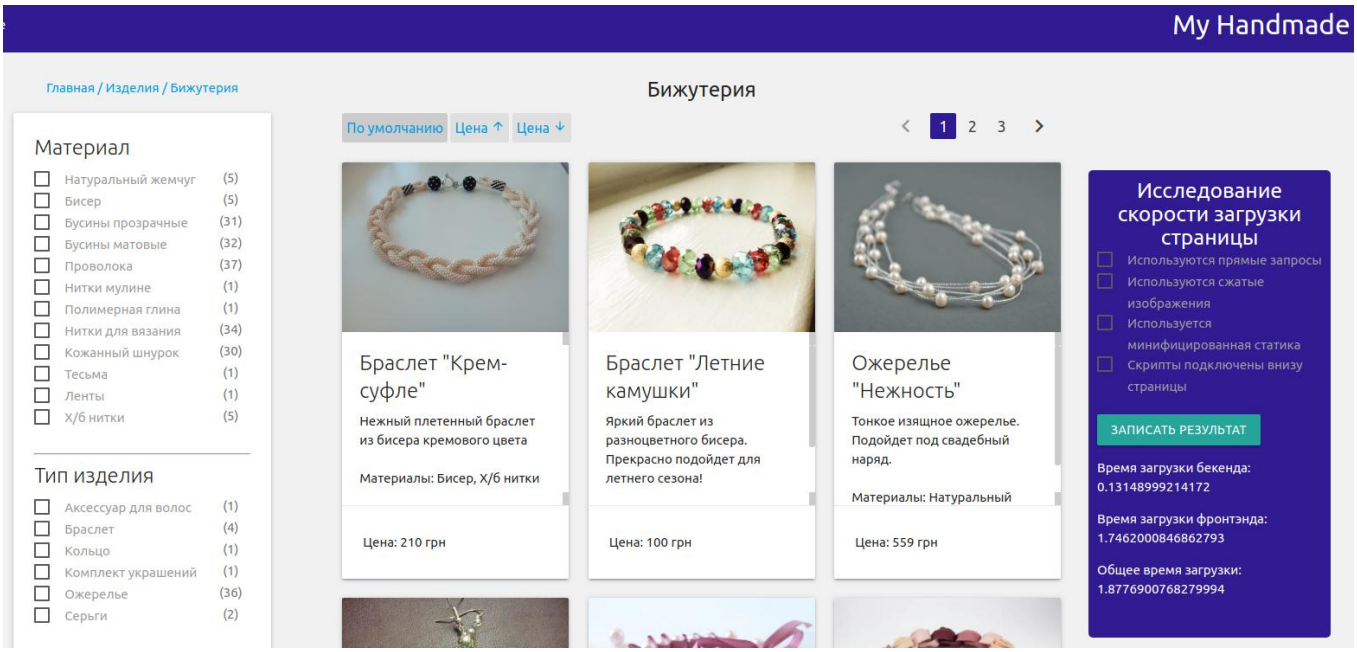

Рисунок 4 – Інструмент виміру часу на сторінці товарів

Як видно з рисунка, окрім результатів вимірювань, інструмент містить форму, у якій є можливість вибрати фактори оптимізації, що були застосовані під час експерименту. Натиснувши на кнопку, результати можна записати в БД до таблиці “experiments”. По мірі необхідності є можливість вивантажувати дані з цієї таблиці у зручному для подальшого аналізу вигляді, про що буде детально розказано далі.

4.3 Опис проведених експериментів

Спочатку необхідно визначитися з позначеннями та необхідною кількістю експериментів.

Було виділено 4 фактори оптимізації системи, кожен з яких може перебувати у стані «активний» або «неактивний». Позначимо ці стани 1 та 0 відповідно.

Щоб порахувати кількість експериментів, необхідно 2 стани факторів розмістити по 4 місцях в усіх можливих комбінаціях, у тому числі з повтореннями. Тобто маємо формулу (2) – формулу комбінаторики «розміщення з повтореннями»:

$$\tilde{A}_n^k = 2^4 = 16 \quad (2)$$

де $\tilde{A}$ – кількість експериментів;

n – кількість станів;

k – кількість факторів.

Отже, потрібно виконати 16 експериментів. Схематичне зображення стану факторів у кожному експерименті наведено у таблиці 1.

Таблиця 1 – Комбінації станів факторів для кожного експерименту

№ експерименту	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	0	0	0	0
2	0	0	0	1
3	0	0	1	0
4	0	0	1	1
5	0	1	0	0
6	0	1	0	1
7	0	1	1	0
8	0	1	1	1
9	1	0	0	0
10	1	0	0	1
11	1	0	1	0
12	1	0	1	1
13	1	1	0	0
14	1	1	0	1
15	1	1	1	0
16	1	1	1	1

Кожен експеримент було проведено у приватному вікні браузера Google Chrome, щоб виключити можливість кешування. Також у коді було налаштоване примусове завантаження усіх скриптів, стилів та зображень без можливості кешування.

Для кожного експерименту виконувалося по 10 вимірювань часу завантаження сторінки, після чого обчислювалось середнє значення. Це надало змогу зменшити ймовірні похибки, що обумовлюються швидкістю інтернет-з’єднання та навантаженням на хостинг.

Результати першого експерименту представлені у таблиці 2.

Таблиця 2 – Результати першого експерименту (0, 0, 0, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	2,079589844	0	0	0	0
2	2,228677034	0	0	0	0
3	2,232164145	0	0	0	0
4	2,093738079	0	0	0	0
5	2,191606998	0	0	0	0
6	2,484255075	0	0	0	0
7	2,382527113	0	0	0	0
8	2,573390245	0	0	0	0
9	2,093281984	0	0	0	0
10	2,076581955	0	0	0	0
	Середній:				
	2,243581247	0	0	0	0

Перший експеримент являє собою заміри часу на неоптимізованій системі, тобто:

– використовується ORM;

– зображення не стиснені;

– скрипти та стилі не мініфіковані;

– скрипти підключені вгорі розмітки сторінки.

Значення факторів оптимізації можна позначити як 0, 0, 0, 0, оскільки вони не були застосовані.

На рисунку 5 показаний вигляд інструменту під час одного з вимірів.

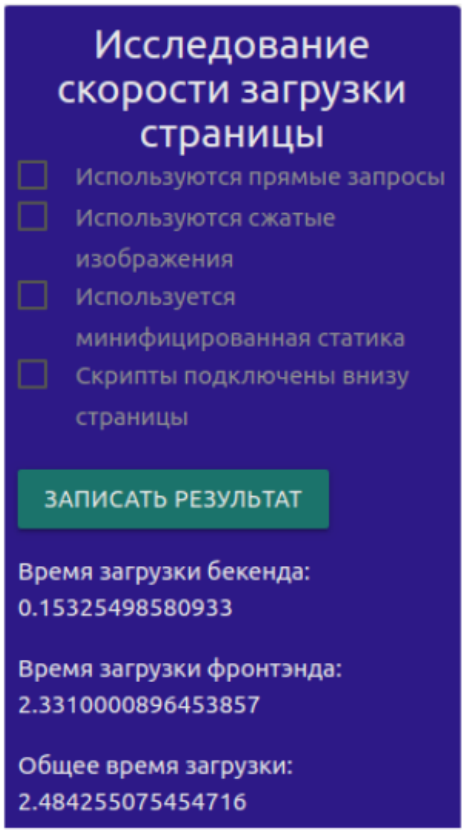

Рисунок 5 – Вимір під час першого експерименту

Нижче наведений код, який обчислює та виводить результати вимірювання часу.

```
        window.onload = function() {
            var currentTimestamp = Date.now();
            currentTimestamp = currentTimestamp / 1000;
            var backendFinished =
parseFloat(document.getElementById("backendFinish").getAttribute("data-
load"));
            var frontLoadTime = currentTimestamp - backendFinished;

            var backendLoadTime =
parseFloat(document.getElementById("loadTime").getAttribute("data-load"));
            var fullLoadTime = backendLoadTime + frontLoadTime;
            var investigationBlock = $("#investigation");
            investigationBlock.append('<input type="hidden"
id="fullLoadTime" value="' + fullLoadTime + '"/>');
            investigationBlock.append('<p>Время загрузки бекенда: ' +
backendLoadTime + '</p>');
```

```
            investigationBlock.append('<p>Время загрузки фронтэнда: ' +
frontLoadTime + '</p>');
            investigationBlock.append('<p>Общее время загрузки: ' +
fullLoadTime + '</p>');
```

Для неоптимізованої системи був проведений аналіз за допомогою сервісу PSI. Отримані результати наведено на рисунку 6.

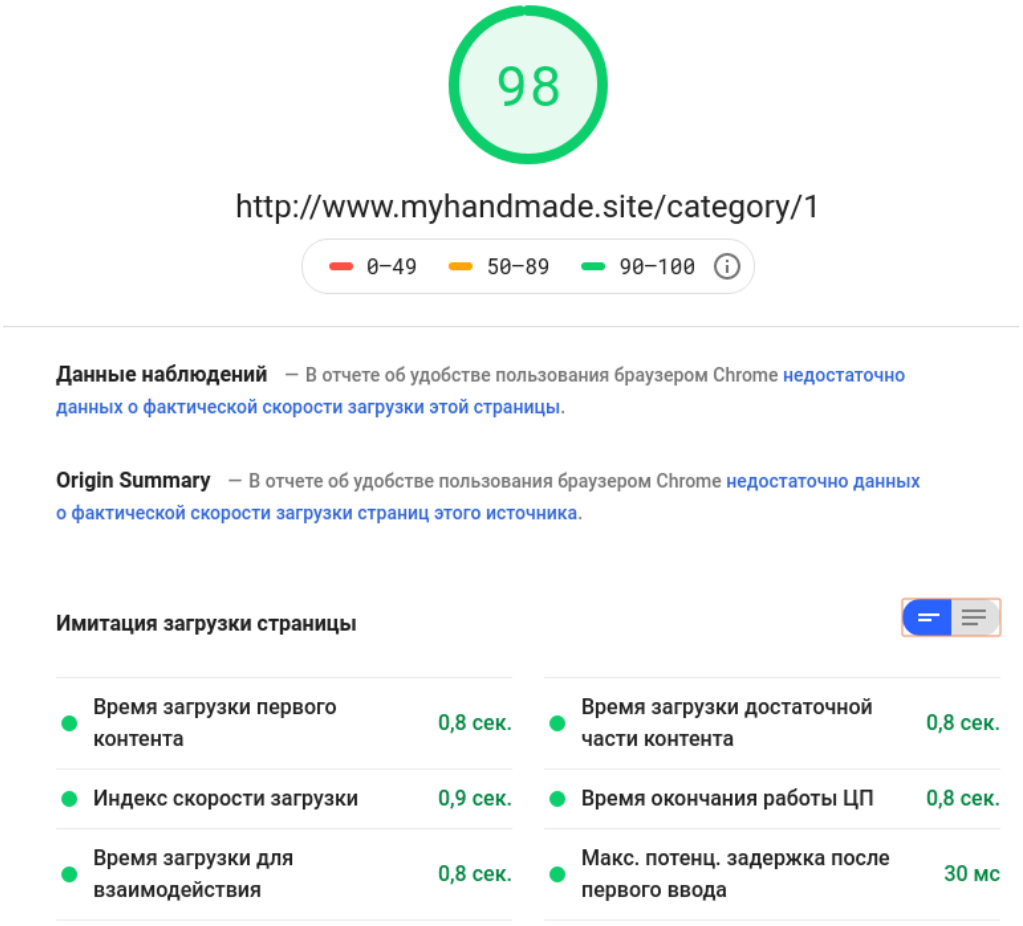

Рисунок 6 – Результати аналізу не оптимізованої системи у PSI

Як видно з рисунку, отримані значення не співпадають з результатами вимірів за допомогою самописного інструменту, але це зумовлено тим, що PSI не враховує швидкість з’єднання, а також має свої особливі алгоритми розрахунків, як зазначалось раніше. Отримана оцінка є досить високою.

Далі були проведені експерименти 2-15 (нумерацію див. у таблиці 1). Для них не проводились вимірювання за допомогою сервісу PSI, оскільки результати не були б досить точними. Результати експериментів, проведених за допомогою самописного інструменту вимірювання часу, наведені у таблицях 3-16.

Таблиця 3 – Результати другого експерименту (0, 0, 0, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,994728804	0	0	0	1
2	1,92519784	0	0	0	1
3	2,091004848	0	0	0	1
4	1,822798014	0	0	0	1
5	2,301831961	0	0	0	1
6	2,012765884	0	0	0	1
7	1,93995595	0	0	0	1
8	2,337400913	0	0	0	1
9	1,881169796	0	0	0	1
10	1,919710875	0	0	0	1
	Середній				
	2,022656488	0	0	0	1

Таблиця 4 – Результати третього експерименту (0, 0, 1, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	2,390100002	0	0	1	0
2	2,079747915	0	0	1	0
3	2,156044245	0	0	1	0
4	2,152264833	0	0	1	0
5	2,398879051	0	0	1	0
6	2,190287828	0	0	1	0
7	2,248314142	0	0	1	0
8	1,875099182	0	0	1	0
9	1,914840937	0	0	1	0
10	2,054647207	0	0	1	0
	Середній				
	2,146022534	0	0	1	0

Таблиця 5 – Результати четвертого експерименту (0, 0, 1, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,909723043	0	0	1	1
2	1,876940012	0	0	1	1
3	1,910982847	0	0	1	1
4	1,973463058	0	0	1	1
5	1,837249041	0	0	1	1
6	2,042140961	0	0	1	1
7	1,922727823	0	0	1	1
8	1,93827486	0	0	1	1
9	1,902523041	0	0	1	1
10	1,917464018	0	0	1	1
	Середній				
	1,92314887				

Таблиця 6 – Результати п’ятого експерименту (0, 1, 0, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	2,097400904	0	1	0	0
2	2,168391943	0	1	0	0
3	2,015087843	0	1	0	0
4	2,066659212	0	1	0	0
5	2,011646986	0	1	0	0
6	2,023306131	0	1	0	0
7	2,214337826	0	1	0	0
8	2,081648827	0	1	0	0
9	1,906606913	0	1	0	0
10	2,013796091	0	1	0	0
	Середній				
	2,059888268	0	1	0	0

Таблиця 7 – Результати шостого експерименту (0, 1, 0, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,967872143	0	1	0	1
2	2,193325043	0	1	0	1
3	2,07466507	0	1	0	1
4	1,987890959	0	1	0	1
5	2,079749107	0	1	0	1
6	2,111679316	0	1	0	1
7	1,994282007	0	1	0	1
8	2,093892097	0	1	0	1
9	2,097757816	0	1	0	1
10	1,85756588	0	1	0	1
	Середній				
	2,045867944	0	1	0	1

Таблиця 8 – Результати сьомого експерименту (0, 1, 1, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,836182833	0	1	1	0
2	1,898257971	0	1	1	0
3	1,841228008	0	1	1	0
4	2,047595978	0	1	1	0
5	1,868808031	0	1	1	0
6	2,011519909	0	1	1	0
7	1,890366077	0	1	1	0
8	2,031462908	0	1	1	0
9	1,922749758	0	1	1	0
10	1,960041046	0	1	1	0
	Середній				
	1,930821252	0	1	1	0

Таблиця 9 – Результати восьмого експерименту (0, 1, 1, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,81593895	0	1	1	1
2	1,963927984	0	1	1	1
3	1,87665081	0	1	1	1
4	1,920342922	0	1	1	1
5	1,933310032	0	1	1	1
6	1,876349211	0	1	1	1
7	1,857350826	0	1	1	1
8	1,979985237	0	1	1	1
9	1,907285929	0	1	1	1
10	1,835536003	0	1	1	1
	Середній				
	1,89666779	0	1	1	1

Таблиця 10 – Результати дев’ятого експерименту (1, 0, 0, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	2,016444683	1	0	0	0
2	1,931794882	1	0	0	0
3	1,940884829	1	0	0	0
4	1,916116238	1	0	0	0
5	1,885405064	1	0	0	0
6	2,068695068	1	0	0	0
7	2,04676199	1	0	0	0
8	1,973317146	1	0	0	0
9	1,921426058	1	0	0	0
10	2,028602123	1	0	0	0
	Середній:				
	1,972944808	1	0	0	0

Таблиця 11 – Результати десятого експерименту (1, 0, 0, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,849751949	1	0	0	1
2	1,884717941	1	0	0	1
3	1,974166155	1	0	0	1
4	1,859055996	1	0	0	1
5	1,825587273	1	0	0	1
6	1,875868082	1	0	0	1
7	1,834658146	1	0	0	1
8	1,938117981	1	0	0	1
9	1,987975121	1	0	0	1
10	1,943126917	1	0	0	1
	Середній				
	1,897302556	1	0	0	1

Таблиця 12 – Результати одинадцятого експерименту (1, 0, 1, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,948875904	1	0	1	0
2	1,876355171	1	0	1	0
3	1,858311892	1	0	1	0
4	1,909178972	1	0	1	0
5	1,958450317	1	0	1	0
6	2,019886017	1	0	1	0
7	1,897324085	1	0	1	0
8	1,860609293	1	0	1	0
9	1,895932913	1	0	1	0
10	1,962309122	1	0	1	0
	Середній				
	1,918723369	1	0	1	0

Таблиця 13 – Результати дванадцятого експерименту (1, 0, 1, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,872246027	1	0	1	1
2	1,809081793	1	0	1	1
3	1,86041379	1	0	1	1
4	1,83079195	1	0	1	1
5	1,92461586	1	0	1	1
6	1,864613056	1	0	1	1
7	1,839675345	1	0	1	1
8	1,84567968	1	0	1	1
9	1,854564677	1	0	1	1
10	1,905445666	1	0	1	1
	Середній				
	1,860712784	1	0	1	1

Таблиця 14 – Результати тринадцятого експерименту (1, 1, 0, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,907480955	1	1	0	0
2	1,918442011	1	1	0	0
3	1,836977959	1	1	0	0
4	1,881803989	1	1	0	0
5	1,981238127	1	1	0	0
6	1,815653801	1	1	0	0
7	1,864235163	1	1	0	0
8	1,891139746	1	1	0	0
9	1,91265893	1	1	0	0
10	1,918356895	1	1	0	0
	Середній:				
	1,892798758	1	1	0	0

Таблиця 15 – Результати чотирнадцятого експерименту (1, 1, 0, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,879319191	1	1	0	1
2	1,878573895	1	1	0	1
3	1,953794003	1	1	0	1
4	1,929349184	1	1	0	1
5	1,903537035	1	1	0	1
6	1,925505877	1	1	0	1
7	1,809844971	1	1	0	1
8	1,850981951	1	1	0	1
9	1,817567825	1	1	0	1
10	1,971791983	1	1	0	1
	Середній				
	1,892026591	1	1	0	1

Таблиця 16 – Результати п’ятнадцятого експерименту (1, 1, 1, 0)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,940821886	1	1	1	0
2	1,92638588	1	1	1	0
3	1,807682037	1	1	1	0
4	1,858352184	1	1	1	0
5	1,893385172	1	1	1	0
6	1,870264769	1	1	1	0
7	1,842620134	1	1	1	0
8	1,941364765	1	1	1	0
9	1,760087252	1	1	1	0
10	1,867398977	1	1	1	0
	Середній:				
	1,870836306	1	1	1	0

В останньому експерименті активними є усі фактори:

– використовуються безпосередні SQL запити до БД;

– використовуються стиснені зображення у форматі WebP;

– скрипти та стилі мініфіковані;

– скрипти підключені перед закриваючим тегом </body>, а не перед </head>.

Результати експерименту 16 продемонстровані у таблиці 17.

Таблиця 17 – Результати шістнадцятого експерименту (1, 1, 1, 1)

№	Час завантаження	Прямі запити	Стиснені зображення	Мініфікація	Скрипти внизу
1	1,750344276	1	1	1	1
2	1,885823011	1	1	1	1
3	1,878547192	1	1	1	1
4	1,938464642	1	1	1	1
5	1,81799984	1	1	1	1
6	1,817178011	1	1	1	1
7	1,865087032	1	1	1	1
8	1,841116905	1	1	1	1
9	1,93601799	1	1	1	1
10	1,802643299	1	1	1	1
	Середній:				
	1,85332222	1	1	1	1

Як вже помітно з результатів експериментів, виділені фактори суттєво впливають на швидкість завантаження веб-сторінки, а у випадку активації усіх факторів оптимізації вдалось отримати найменше значення середнього часу. Різниця між середнім часом завантаження сторінки оптимізованої та не оптимізованої системи склала приблизно 0,39 секунд. Також підтвердженням успішної оптимізації слугує аналіз сторінки за допомогою PSI, результати якого показані на рисунку 7. Як видно за показником часу завантаження сторінки для взаємодії, вдалось скоротити цей час з 0,8 до 0,5 секунд. Також загальна оцінка зросла до 99 балів із 100.

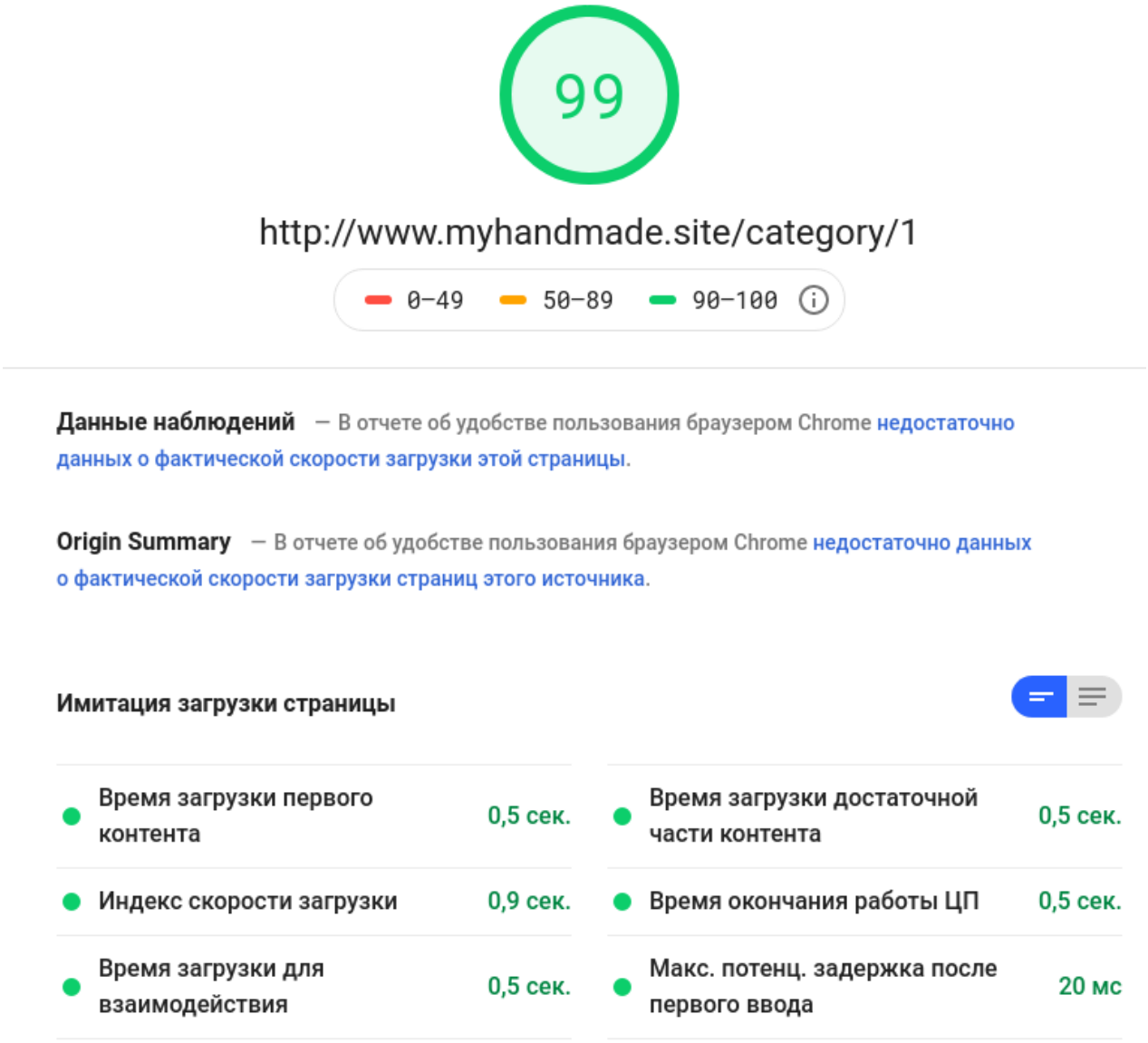

Рисунок 7 – Результати аналізу в PSI з усіма активними факторами

Для визначення вагомості кожного з факторів оптимізації середній час завантаження сторінки, отриманий в результаті кожного з експериментів, було внесено в окрему таблицю для подальшого дослідження за допомогою регресійного аналізу.

Нехай Y – це середній час завантаження сторінки, X_1 – використання прямих запитів замість ORM, X_2 – використання стиснених без втрати якості зображень, X_3 – мініфікація скриптів та стилів, X_4 – підключення скриптів унизу розмітки сторінки замість підключення вгорі. У формулі (3) наведено загальний вигляд рівняння часу завантаження сторінки.

$$Y = Y_0 + aX_1 + bX_2 + cX_3 + dX_4 \quad (3)$$

де Y_0 – загальний час завантаження сторінки;

Y_0 – час завантаження сторінки за відсутності факторів оптимізації;

a, b, c, d – коефіцієнти, що відповідають вагомості кожного з факторів;

X_1, X_1, X_1, X_1 – фактори впливу.

Задача полягає в знаходженні коефіцієнтів a, b, c, d.

У таблиці 18 наведено підготовлені для регресійного аналізу дані. Наявність факторів оптимізації позначається значенням 1, а відсутність – значенням 0.

Таблиця 18 – Вхідні дані для регресійного аналізу

Y	X1	X2	X3	X4
2,2436	0	0	0	0
2,0227	0	0	0	1
2,146	0	0	1	0
1,9231	0	0	1	1
2,0599	0	1	0	0
2,0459	0	1	0	1
1,9308	0	1	1	0
1,8967	0	1	1	1
1,9729	1	0	0	0
1,8973	1	0	0	1
1,9187	1	0	1	0
1,8607	1	0	1	1
1,8928	1	1	0	0
1,892	1	1	0	1
1,8708	1	1	1	0
1,8533	1	1	1	1

Аналіз було виконано за допомогою пакету аналізу даних MS Excel. Для цього необхідно вибрати «Дані» – «Аналіз даних» – «Регресія» та зазначити діапазони комірок таблиці, у яких знаходяться дані, як показано на рисунку 8.

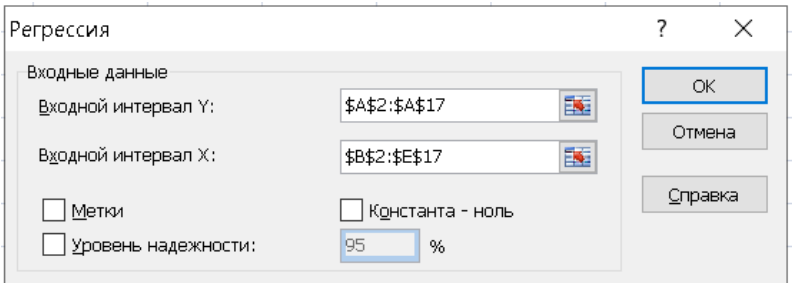

Рисунок 8 – Вибір вхідних даних для регресії в Excel

Результати регресійного аналізу наведено на рисунку 9.

Регресійна статистика	
Множинний R	0,884782578
R-квадрат	0,782840211
Нормований R-квадрат	0,703873015
Стандартна похибка	0,060656016
Спостереження	16
	Коефіцієнт
Y-перетин	2,146930947
Змінна X 1	-0,138748375
Змінна X 2	-0,067857941
Змінна X 3	-0,078351442
Змінна X 4	-0,080488912

Рисунок 9 – Результати регресійного аналізу в Excel

R-квадрат називають коефіцієнтом детермінації. Це значення, яке відображає адекватність моделі, тобто те, наскільки обрана модель пояснює залежність між параметрами, що досліджуються. Високоякісною вважається модель з коефіцієнтом детермінації 0,8 або вище. У моделі, що розглядається, коефіцієнт дорівнює 0,78, що свідчить про достатню якісність моделі.

Y-перетин показує, яке значення приймає Y, коли значення незалежних змінних дорівнюють 0. Це значення незначно відхиляється від результату експерименту без факторів оптимізації, але є адекватним, враховуючи можливу похибку.

Змінні X1, X2, X3 та X4 є шуканими ваговими коефіцієнтами досліджуваних факторів. Знак «-» вказує на від'ємний вплив, що у даному дослідженні є справедливим – за наявності фактора час завантаження сторінки зменшується. Отже, кінцеве рівняння часу завантаження веб-сторінки має вигляд:

$$Y = 2{,}1469 - 0{,}1387X_1 - 0{,}0679X_2 - 0{,}0784X_3 - 0{,}0804X_4 \qquad (4)$$

де Y – час завантаження сторінки;

X_1, X_2, X_3, X_4 – фактори оптимізації.

З отриманих результатів можна зробити висновок, що для розглянутої веб-сторінки найбільш вагомим фактором оптимізації виявилося використання прямих запитів до БД замість використання ORM. Варто зазначити, що такий підхід є дієвим та оптимальним, коли передбачено багато вибірок даних за допомогою оператора SELECT. Для випадків, коли треба записати або редагувати невелику кількість даних, зручніше з точки зору коду використовувати засоби ORM, оскільки тоді час виконання не є критичним.

Найменш вагомим фактором виявилося використання стиснених без втрати якості зображень. Теоретично формат WebP дозволяє виконувати більш жорстке стиснення, але через втрату якості найчастіше воно не має сенсу. У розглянутому випадку на сторінку виводилось лише 15 зображень, при наявності більшої кількості результат міг би відрізнятися.

Другим за вагомістю фактором виявилося підключення скриптів у нижній частині розмітки сторінки. Проте слід зауважити, що такий підхід не завжди є можливим – виникає необхідність підключити їх у тезі <head>, якщо потрібно виконати певні дії під час початку завантаження розмітки сторінки. Але найчастіше даний підхід може бути використаний на практиці.

Мініфікація файлів скриптів та стилів виявилась третім за вагомістю фактором. Показник впливу може бути збільшений у разі великої кількості підключених файлів. У даному випадку було підключено три файли стилів та чотири файли скриптів (бібліотеки).

ВИСНОВКИ

Швидкість завантаження веб-сторінки є однією з найважливіших характеристик, що обумовлюють популярність, кореляцію та пошукове ранжування веб-сайту.

Для дослідження вагомості факторів оптимізації швидкості завантаження веб-сторінки було використано емпіричні та емпірико-теоретичні методи дослідження, а саме: вимірювання, експеримент, порівняння та математичне моделювання. Також було розроблено експериментальну програмну систему, яка є типовою моделлю інтернет-магазину та передбачає можливість оптимізації.

Було виділено чотири фактори оптимізації програмного коду та структури системи, що впливають на швидкість завантаження веб-сторінки: використання прямих запитів до бази даних замість готової реалізації ORM, стиснення зображень без втрат якості, мініфікація файлів стилів та скриптів, підключення файлів скриптів перед закриваючим тегом тіла розмітки сторінки.

Експериментально доведено, що кожен з факторів зменшує час завантаження веб-сторінки, та досліджено їх вагомість. Найбільший вплив на час завантаження здійснює використання прямих запитів до бази даних, найменший – використання стиснених зображень. У випадку реальних веб-систем вагомість кожного з факторів може відрізнятися залежно від особливостей їх структури, проте наведені фактори є досить універсальними та можуть бути застосовані для більшості веб-систем.

Результати дослідження можуть бути використані на практиці у сфері розробки веб-систем та є найбільш корисними для веб-розробників як серверної, так і клієнтської частин.

У подальшому можуть буди досліджені методи оптимізації веб-систем на рівні структури бази даних, реляційної або об’єктно-орієнтованої.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Правда о том, насколько важна скорость загрузки сайта [Електронний ресурс] – Режим доступу: https://vc.ru/flood/34484-pravda-o-tom-naskolko-vazhna-skorost-zagruzki-sayta – 29.10.2019 р. – Назва з екрану.
2. Скорость загрузки страницы: метрики, инструменты и способы повышения [Електронний ресурс] – Режим доступу: https://blog.cybermarketing.ru/skorost-zagruzki-stranicy-metriki-instrumenty-i-sposoby-povysheniya/ – 29.10.2019 р. – Назва з екрану.
3. Как увеличить скорость загрузки сайта, чтобы не терять клиентов [Електронний ресурс] – Режим доступу: https://www.owox.ru/blog/use-cases/page-speed/ – 29.10.2019 р. – Назва з екрану.
4. Как скорость загрузки сайтов влияет на конверсию трафика? [Електронний ресурс] – Режим доступу: https://ppc.world/articles/kak-skorost-zagruzki-saytov-vliyaet-na-konversiyu-trafika-issledovanie-rynka-nedvizhimosti/ – 30.10.2019 р. – Назва з екрану.
5. Скорость загрузки сайта [Електронний ресурс] – Режим доступу: https://www.ashmanov.com/education/articles/skorost-zagruzki-sajta/ – 30.10.2019 р. – Назва з екрану.
6. Грабченко А.І., Федорович В.О., Гаращенко Я.М. Методи наукових досліджень: Навч. посібник. – Х.: НТУ "ХПІ", 2009. – 142 с.
7. Методы научного познания [Електронний ресурс] – Режим доступу: https://gtmarket.ru/concepts/6874 – 05.11.2019 р. – Назва з екрану.
8. Модель [Електронний ресурс] – Режим доступу: https://gtmarket.ru/concepts/7024 – 05.11.2019 р. – Назва з екрану.
9. Маценко В.Г. Математичне моделювання: навчальний посібник - Чернівці: Чернівецький національний університет, 2014.–519 с.
10. PageSpeed Insights API [Електронний ресурс] – Режим доступу: https://developers.google.com/speed/docs/insights/v5/about – 10.11.2019 р. – Назва з екрану.

11. How To Think About Speed Tools [Електронний ресурс] – Режим доступу: https://developers.google.com/web/fundamentals/performance/speed-tools/ – 10.11.2019 р. – Назва з екрану.
12. ORM - Object Relational Mapping [Електронний ресурс] – Режим доступу: https://webshake.ru/oop-v-php-prodvinutyj-kurs/object-relational-mapping-orm-v-php – 15.11.2019 р. – Назва з екрану.
13. Мациевский Н.С. Реактивные веб-сайты. Клиентская оптимизация в алгоритмах и примерах: Учебное пособие / Н.С. Мациевский, Е.В. Степанищев, Г.И. Кондратенко — М.: Інтернет-університет інформаційних технологій: БИНОМ. Лабораторія знань, 2010. – 336 с.
14. WebP сегодня: зачем и как? [Електронний ресурс] – Режим доступу: https://medium.com/web-standards/webp-сегодня-для-чего-и-как-4f64d4330f8d – 15.11.2019 р. – Назва з екрану.
15. Регрессионный анализ [Електронний ресурс] – Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Регрессионный_анализ – 15.11.2019 р. – Назва з екрану.
16. Множественная линейная регрессия [Електронний ресурс] – Режим доступу: http://statistica.ru/theory/mnozhestvennaya-lineynaya-regressiya/– 15.11.2019 р. – Назва з екрану.
17. Laravel — лидер среди PHP фреймворков, одобренный разработчиками [Електронний ресурс] – Режим доступу: https://webformyself.com/laravel-lider-sredi-php-frejmvorkov-odobrennyj-razrabotchikami/ – 15.11.2019 р. – Назва з екрану.
18. Єрохін А. Л., Бєшлєй М. В. Дослідження методів оптимізації веб-сайтів підприємців на локальних ринках //Системи обробки інформації. – 2014. – №. 5. – С. 150-151.