МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему:

«Інформаційна система оцінювання швидкості завантаження веб-додатків»

Завідувач випускаючої кафедри — **Довбиш А. С.**

Керівник роботи — **Москаленко В. В.**

Студент групи ІН.мз – 92с — **Шепелєв Д. О.**

СУМИ 2021

Сумський державний університет
(назва вузу)

Факультет *ЕлІТ* Кафедра *Комп ютерних наук*
Спеціальність *«Інформатика»*

Затверджую:
зав.кафедрою _____
_____
“_____” _____ 20___р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТОВІ

Шепелєва Дмитра Олександровича
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи) *Інформаційна система оцінювання швидкості завантаження веб-додатків*

затверджую наказом по інституту від “____” _____ 20 ___ р. № _____

2. Термін здачі студентом закінченого проекту (роботи) _____

3. Вхідні данні до проекту (роботи) _____
_____
_____
_____
_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

1) Огляд та аналіз проблеми швидкості завантаження веб-додатків; 2) Постановка задачі; 3) Огляд існуючих технологій та вибір програмних засобів; 4) Огляд методів аналізу; 5) Програмна реалізація алгоритму аналізу та веб-інтерфейсу

_____
_____

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
_____
_____
_____
_____

6. Консультанти до проекту (роботи), із значенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту (роботи)	Термін виконання проекту (роботи)	Примітка
1.			
2.			
3.			
4.			
5.			

Студент – дипломник _____
(підпис)

Керівник проекту _____
(підпис)

РЕФЕРАТ

Записка: 43 стор., 21 рис., 1 додаток, 15 джерел.

Об’єкт дослідження – інформаційна система оцінювання швидкості завантаження веб-додатків.

Мета роботи — проаналізувати проблему швидкого завантаження веб-додатків. Створити інформаційну систему у вигляді веб-додатку для аналізу швидкості завантаження веб-додатків.

Результати — описані методи та алгоритм аналізу швидкості завантаження веб-додатків. Проведені порівняння існуючих методів аналізу. Проведено аналіз існуючих сервісів для аналізу швидкості завантаження веб-додатків. Описаний процес створення інформаційної системи. Результатом роботи стала інформаційна система у вигляді веб-додатку за допомогою сучасних технологій програмування.

REACT, NODE, EXPRESS, RECHARTS, API, PUPPETEER,
ВЕБ-ДОДАТОК

ЗМІСТ

ВСТУП

Інформаційні системи є невід'ємними частинами нашого життя. Зараз навіть важко уявити повсякденність без мобільного інтернету та доступу у Інтернет. Важливим питанням постає зручність використання веб-додатків. Швидкість Інтернету завдає великого впливу на цей процес, але є багато факторів зі сторони розробки додатків. Гарним прикладом може бути досвід використання веб-додатків з мобільних пристроїв чи з комп'ютерів, використання різних браузерів, час використання, наприклад, застосування білої теми вдень та темної теми веб-додатку вночі. Користувачі дають перевагу швидким веб-додаткам, які миттєво завантажуються, які не потребують багато ресурсів від пристрою. Саме у 2020 році в умовах пандемії це питання стало найбільш актуальним. Користувачі звертають увагу на швидкість завантаження веб-додатку, адже швидкість дає впевненість, що даний сайт є надійним. Наприклад, онлайн-магазини втрачають користувачів, якщо останні не бажають чекати через швидкість завантаження. Метою цієї роботи є створення інформаційної системи, що дозволить користувачам та розробникам зрозуміти швидкість завантаження веб-додатку.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Інформаційна системи та веб-додатки

Інформаційна система — комунікаційна система, що забезпечує збирання, пошук, оброблення та пересилання інформації [1]. Варіантом інформаційної системи є веб-додаток. Веб-додатки дуже популярні, та є одним з варіантів отримання інформації у мережі Інтернет.

Спочатку всі веб-сайти носили лише інформаційний характер. Вони складалися з простого тексту без форматування та гіперпосилань. З часом додалася графічна інформація. Сьогодні існує стільки різних типів веб-сайтів, що часом важко віднести певний сайт до певної категорії. Нижче наведені деякі з них.

Персональний сайт демонструє особистість власника веб-додатку та рекламує його, сферу його інтересів, його цілі, вміння та досягнення, іноді включає портфоліо. Зазвичай він створюється власником для того, щоб заявити про себе, знайти друзів, однодумців, людей з подібними поглядами тощо.

Особистий блог - це веб-сайт із регулярними публікаціями, що містять текст, зображення та коментарі. Зазвичай особисті блоги містять кілька сторінок. Їх кількість залежить від активності автора, і іноді блоги можуть бути досить великими. Структура блогу завжди проста і зрозуміла.

Сторінка-вітрина (лендінг) – це веб-сайт, основна мета якого - це спонукати користувача купувати товар. На цільовій сторінці зазвичай відображається товари, які є логічним продовженням результату пошуку або посилання. Цільові сторінки використовуються для генерації потенційних покупців. Дії, які відвідувач здійснює на сторінці-вітрині, визначають коефіцієнт конверсії рекламодавця з боку веб-додатку.

Розважальні веб-додатки – це сайт, який дозволяю користувачу насолодитись контентом, наприклад, перегляд кінофільму, браузерні ігри, платформи для прослуховування музики, комедійні портали і тд.

Сайт новин - це впливова інформаційна система, яка присвячена новинам з різних сфер життя. Основна мета сайту - якомога швидше і детальніше надавати користувачеві інформацію. Новини на сайтах найчастіше дуже швидко оновлюються наживо.

1.2 Технології створення веб-додатків

Правильний набір технологій значною мірою є запорукою успіху веб-додатку, тоді як неправильний вибір технологій розробки веб-програм може бути причиною невдачі. Вибір відповідного стеку технологій особливо важливий для малого бізнесу та приватних осіб, оскільки вони, як правило, мають обмежений бюджет і, отже, їм потрібен стек технологій, який забезпечує найбільший виграш для того, щоб їхні проекти були успішними. Великим компаніям слід ретельніше досліджувати стек на можливість впоратися з великою кількістю користувачів.

У веб-розробки є дві сторони: клієнтська та серверна. Клієнтська сторона також називається інтерфейсом. Програмування на стороні сервера включає програму, базу даних і сервер.

Клієнтська веб-розробка включає все, що користувачі бачать на своїх екранах. Деякі основні компоненти інтерфейсу:

- мова розмітки гіпертексту (HTML) та каскадні таблиці стилів (CSS). HTML повідомляє браузеру, як відображати вміст веб-сторінок, тоді як CSS надає стилістичне відображення цього вмісту;
- JavaScript (JS). JS робить веб-сторінки інтерактивними. Існує безліч бібліотек JavaScript та фреймворків для швидшої та простішої веб-розробки. Гарними прикладами є jQuery, React.js, Angular.js, Vue.js.

З серверною стороною взаємодіють за допомогою запитів, найчастіше серверна сторона не видна для кінцевого користувача, лише

тільки через API, коли така можливість існує. Зараз велика кількість ферймворків та програмних мов доступні для реалізації серверної частини. Прикладами могуть послугувати Ruby on Rails (Ruby), Django (Python), Spring (Java), Nest.js (Node.js) і тд.

Спочатку JavaScript був створений, щоб зробити веб-сторінки «живими». Програми на цій мові називаються скриптами. Вони можуть вбудовуватися в HTML і виконуватися автоматично при завантаженні веб-сторінки. Скрипти поширюються і виконуються, як простий текст. Їм не потрібна спеціальна підготовка або компіляція для запуску. Сьогодні JavaScript може виконуватися не тільки в браузері, а й на сервері або на будь-якому іншому пристрої, який має спеціальну програму, що називається «двигун» JavaScript. У браузера є власний двигун, який іноді називають «віртуальна машина JavaScript». Різні движки мають різні імена, наприклад, V8 - в Chrome і Opera, SpiderMonkey - в Firefox. [2]

На сьогодні використання JavaScript вже недостатньо, з’являються нові стандарти та нові виклики у створенні веб-додатків. Підтримувати JavaScript у великих командах стає з кожним роком важче, адже веб-додатки стають складніше та розширюють можливості користувача. JavaScript - це мова програмування з динамічною типізацію. Саме тому команда Microsoft презентувала нову мову програмування над рівнем JavaScript.

TypeScript - це набір JavaScript, який поєднує перевірку типу та статичний аналіз, явні інтерфейси та найкращі практики в єдину мову та компілятор. Спираючись на JavaScript, TypeScript тримає розробника поруч із середовищем виконання, на яке він націлюється, додаючи лише синтаксичний цукор, необхідний для підтримки великих програм та великих команд. [3]

1.3 Швидкість веб-додатків

Перші враження важливі в Інтернеті. Користувачі та клієнти за швидкістю завантаження роблять висновки про якість послуг на сайті. Якщо веб-сайт завантажується швидко, то користувач відразу помітить це та отримає гарне перше враження. Це перший крок для продажу послуг чи товарів на сайті. Якщо він завантажується швидко, то користувач чи клієнт реагує на це позитивно, тобто за замовчуванням очікується швидке завантаження веб-додатку, але якщо сайт завантажується повільно, то це відразу викликає негативні відчуття.

Користувачі вважають швидкі веб-сайти професійними та надійними. З іншого боку, повільний веб-сайт змушує нас думати, що він небезпечний та ненадійний. Важко повернути це негативне перше враження. Існуючі веб-сайти встановлює високу планку, коли мова заходить про швидкість сайту. Підтвердженням цим словам може послугувати статистика від Google. 53% користувачів закривають сайт, якщо мобільному сайту потрібно більше 3 секунд для завантаження [4]. Ось ще статистика від Sean Work та компанії Kissmetrics [5]:

- 73% користувачів мобільного Інтернету кажуть, що вони стикалися з веб-сайтом, який завантажувався занадто повільно;
- 51% користувачів мобільного Інтернету кажуть, що вони стикалися з веб-сайтом, який аварійно завершив роботу, завис або отримав помилку;
- 38% користувачів мобільного Інтернету кажуть, що вони стикалися з веб-сайтом, який був недоступний;
- 47% споживачів очікують, що веб-сторінка завантажиться за 2 секунди або менше;
- 40% людей залишають веб-сайт, який завантажується більше 3 секунд;

- затримка відповіді сторінки на 1 секунду може призвести до зменшення конверсій на 7%;
- якщо сайт електронної комерції заробляє 100 000 доларів на день, затримка сторінки на 1 секунду може потенційно коштувати 2,5 мільйона доларів втрачених продажів щороку.

Згідно даним Econsultancy.com [6] конверсія падає якщо сайт завантажується довше 1-2 секунд, саме про це повідомляє наступний графік:

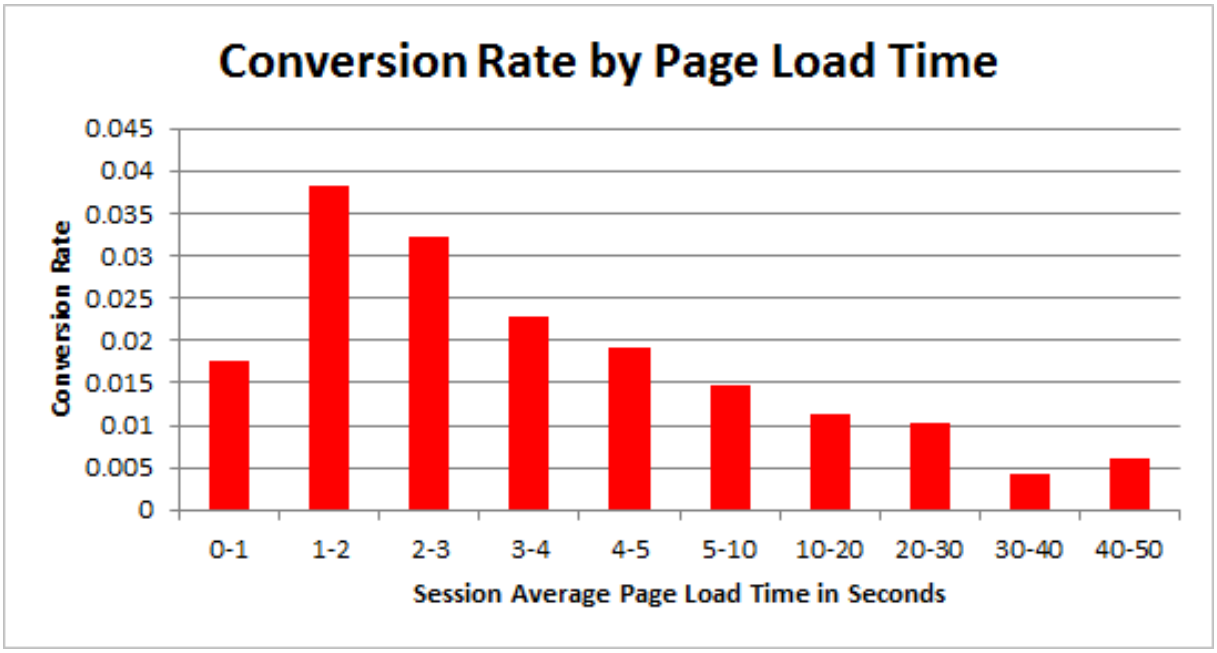

Рисунок 1.1 – Графік залежності конверсії від часу завантаження сайту

Згідно даним MarketingSherpa [7] конверсія більше ніж 5%, якщо веб-сайт завантажується зі швидкістю 2,4 секунди. Також можна зробити висновок з графіку, що більшість сесій, тобто користувачів завантажують веб-додаток зі швидкість 2,4 – 3,6 секунд, що цілком співпадає з попередніми даними.

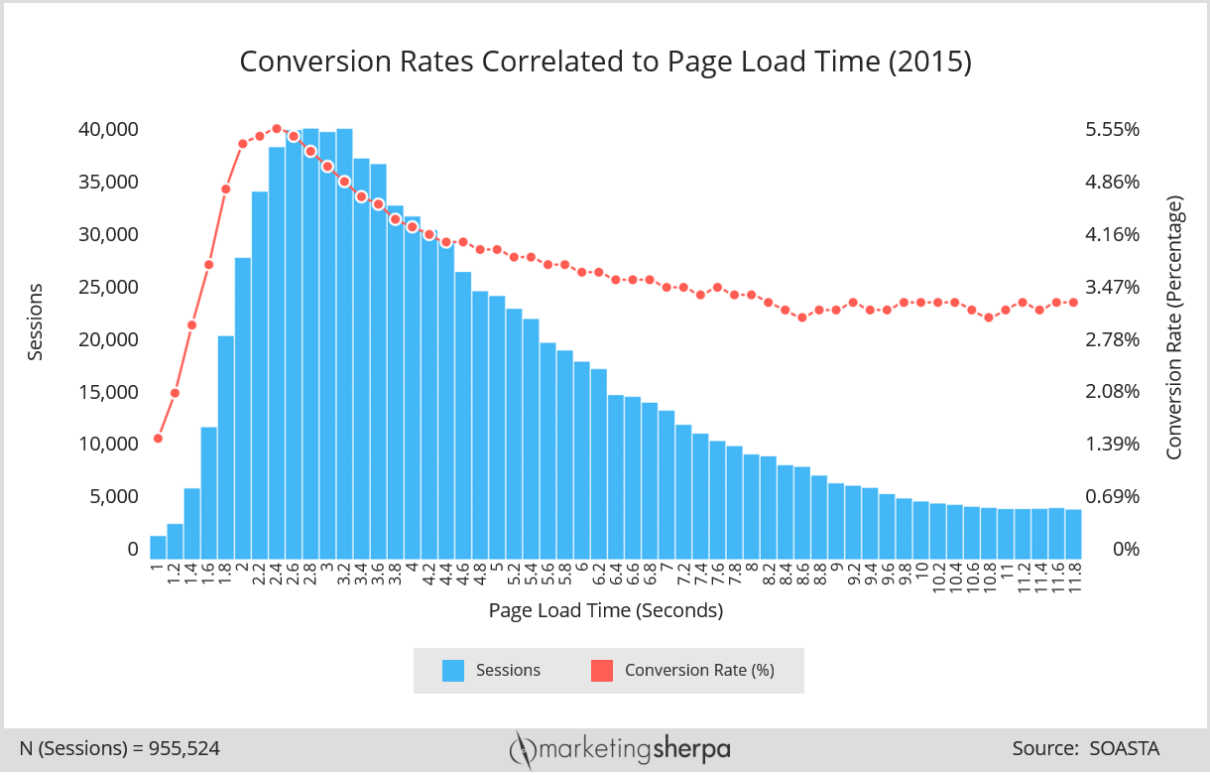

Рисунок 1.2 – Графік залежності сесій та конверсій від швидкості завантаження сайту

При завантаженні сайту зі швидкістю 5 секунд конверсія падає до 3,5%. Це говорить про те, що з 100 000 користувачів у випадку 5 секундного завантаження 1 500 користувачів закривають сайт.

Згідно статистики можна зрозуміти, що швидкість завантаження веб-додатку дуже важлива. Саме тому з'явилися додатки для аналізу швидкості завантаження.

1.4 Веб-додатки для заміру швидкості сайтів

Існують платформи аналізу швидкості завантаження веб-сайтів. Потрібно розглянути та порівняти деякі з них на прикладі сайту Сумського державного університету https://sumdu.edu.ua/uk:

- https://developers.google.com/speed/pagespeed/insights/;
- https://tools.pingdom.com;
- https://www.webpagetest.org.

1.4.1 Google PageSpeed Insights

Першим слід розглянути веб-додаток Google Page Speed Insights. Ось що повідомляє про себе цей сайт: «PageSpeed Insights (PSI) повідомляє про ефективність сторінки як на мобільних пристроях, так і на персональних комп'ютерах, а також пропонує поради щодо покращення цієї сторінки. PSI надає як лабораторні, так і польові дані про сторінку. Лабораторні дані корисні для налагодження проблем продуктивності, оскільки вони збираються в контрольованому середовищі. Однак це може не враховувати реальні вузькі місця. Польові дані корисні для відображення справжнього реального досвіду користувача, але мають більш обмежений набір показників.»[8]

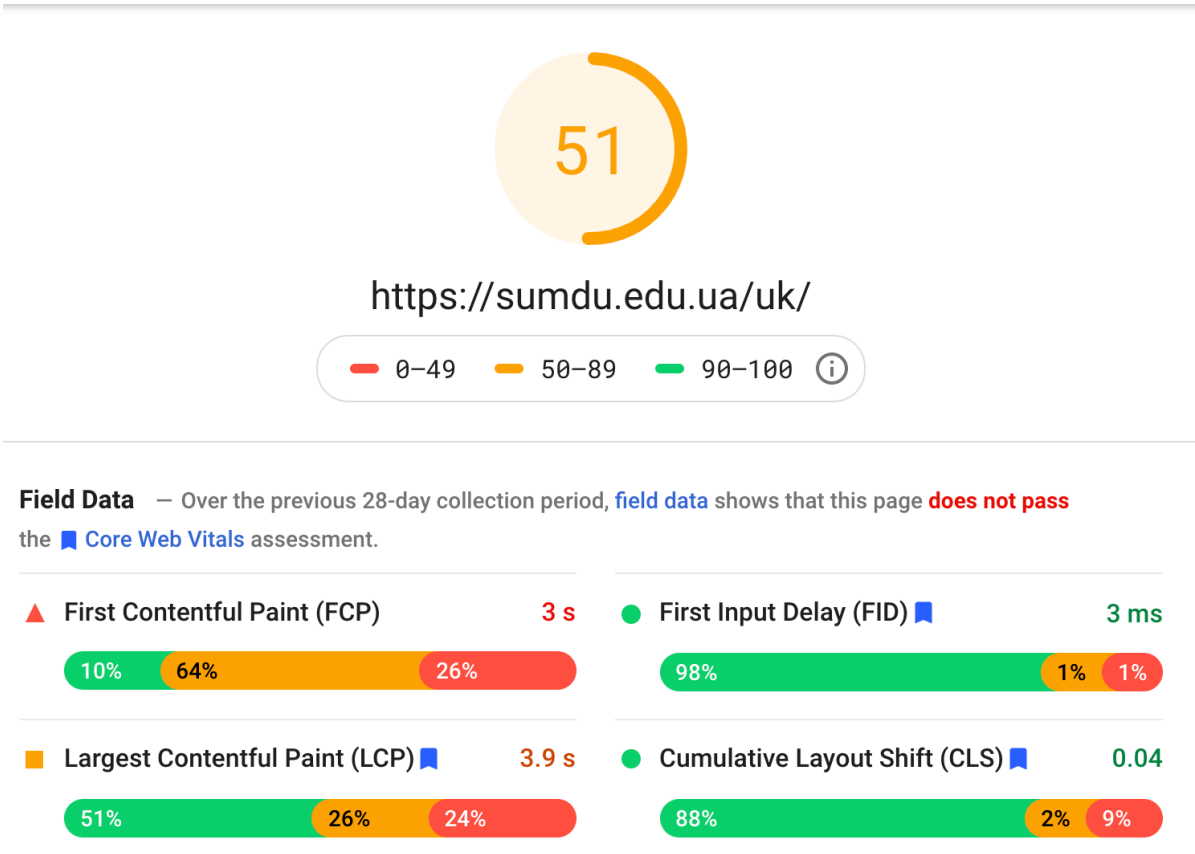

Рисунок 1.3 – Аналіз головної сторінки СумДУ сервісом PageSpeed Insights, відображеної на ПК

Згідно результату аналізу можна побачити, що веб-додаток отримав 51 бал зі 100. Перший зміст відобразився за 3 секунди, весь контент

відобразився за 3,9 секунди. Згідно попереднім графікам залежності конверсії та часу відображення, можна зробити висновок, що сайт на ПК втрачає конверсію.

Потрібно проаналізувати результати відображення сайту на мобільному пристрої, використовуючи той самий веб-додаток від Google. Для цього в налаштування потрібно обрати Mobile.

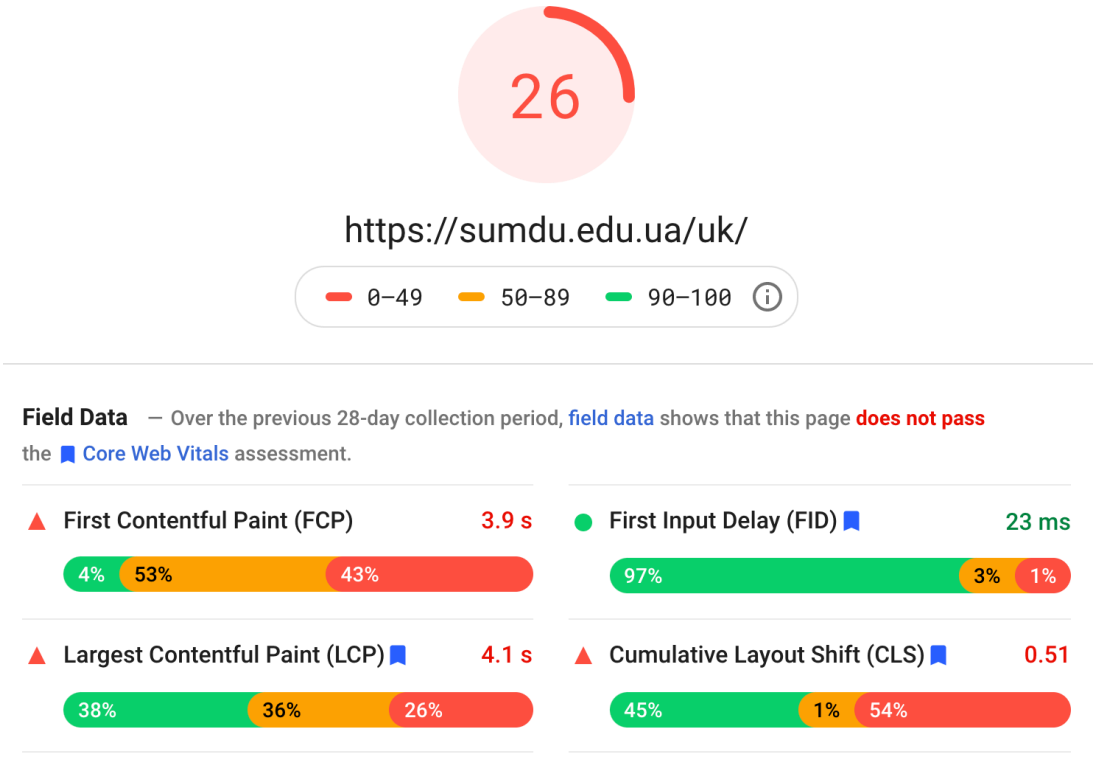

Рисунок 1.4 – Аналіз головної сторінки СумДУ сервісом PageSpeed Insights, відображеної на мобільному пристрої

Згідно результату аналізу можна побачити, що веб-додаток отримав 26 балів зі 100. Перший зміст відобразився за 3,9 секунди, весь контент відобразився за 4,1 секунди. Згідно попереднім графікам залежності конверсії та часу відображення, можна зробити висновок, що сайт на мобільних пристроях втрачає конверсію.

Можна зробити висновок, що PageSpeed Insights використовує лише один алгоритм для аналізу швидкості завантаження сайту.

1.4.2 Pingdom Website Speed Test

Засіб тестування швидкості веб-сайту Pingdom синтетично перевіряє швидкість веб-сайту та повідомляє про його ефективність. Pingdom дозволяє протестувати веб-сайт із семи місць на планеті, включаючи такі [9]:

- Європа (Франкфурт, Німеччина);
- Європа (Лондон, Великобританія);
- Азія (Токіо, Японія);
- Океанія (Сідней, Австралія);
- Північна Америка (Вашингтон, округ Колумбія, США);
- Північна Америка (Сан-Франциско, США);
- Південна Америка (Сан-Паулу, Бразилія).

Це дає змогу чітко зрозуміти, як відрізняється ефективність веб-сайту в залежності від місцезнаходження користувача. Якщо цільова аудиторія базується на певній географічній області, завжди найкраще проаналізувати дані із сервера, що знаходиться найближче до їхнього місцезнаходження, оскільки це дасть найбільш точний аналіз їхнього користувацького досвіду. [9]

Тест швидкості веб-сайту Pingdom простий у використанні, але забезпечує корисні, поглиблені результати. Ці результати показують не тільки те, наскільки швидко ваш сайт завантажується, але чому сайт може бути неефективним, що дозволяє робити стратегічні вдосконалення. [9]

Наступне зображення показує результати аналізу головної сторінки сайту СумДУ.

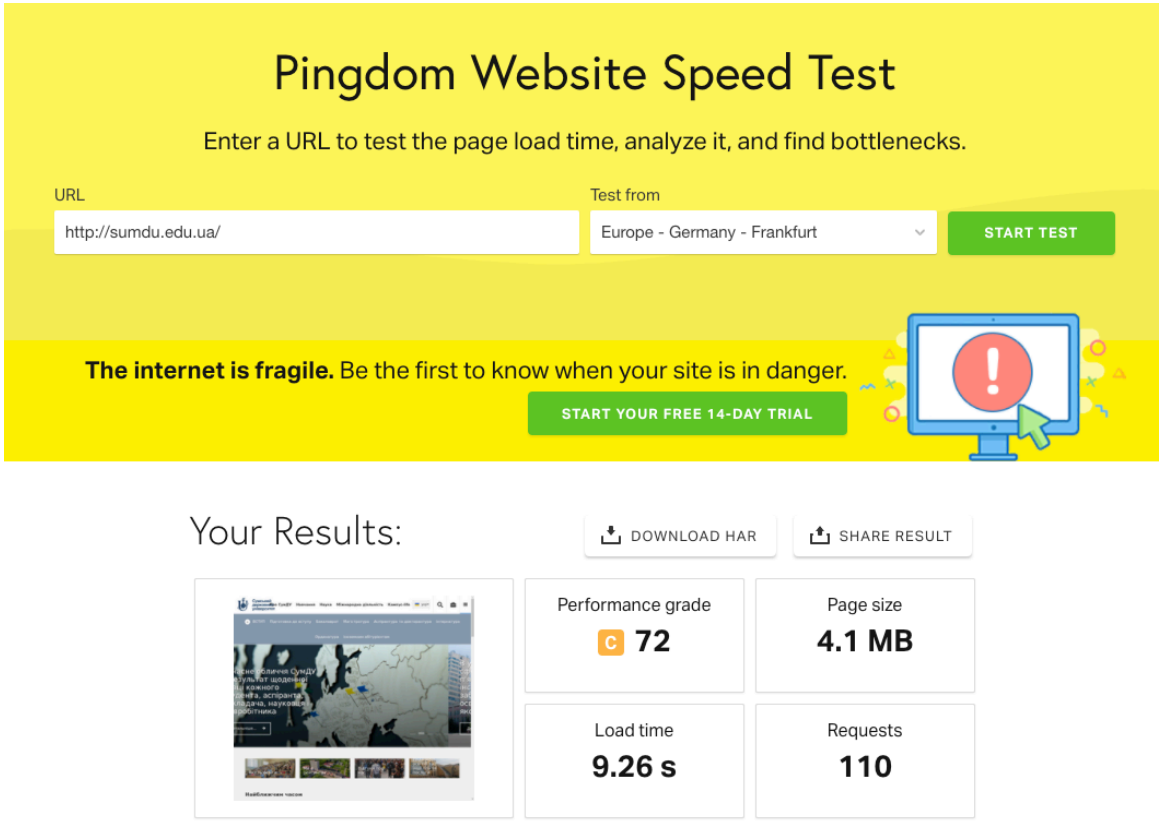

Рисунок 1.5 – Аналіз головної сторінки СумДУ сервісом Pingdom

Згідно результату аналізу, можна зробити висновок, що сторінка повністю завантажується за 9,26 секунд. Сторінка надіслає 110 запитів та важить 4,1 мегабайти. Аналіз дав результат в 72 бали зі 10 та отримав результат С. Згідно попереднім графікам можна зробити висновок, що сайт втрачає конверсію.

1.4.3 Webagetest.org

WebPagetest - це інструмент, який спочатку був розроблений AOL для внутрішнього використання та був відкритим у 2008 році за ліцензією BSD. Платформа активно розробляється на GitHub, а також періодично упаковується і доступна для завантаження, якщо хтось хочете запустити власний екземпляр. [10]

Інтернет-версія на веб-сайті www.webpagetest.org працює на користь спільноти виконавців, де кілька компаній та приватних осіб забезпечують інфраструктуру тестування по всьому світу. [10]

В обмін на проведення місця тестування партнери отримують свій логотип, пов’язаний з місцем розташування, та банер на сайті. Розміщення тестового місця відкрите для кожного. [10]

Веб-додаток дозволяю користувачу змінити багато налаштувань, наприклад, звідки зробити запит, браузер, якість інтернет з’єднання.

Наступне зображення показує результати аналізу головної сторінки сайту СумДУ з наступними налаштуваннями:

- місце запиту Варшава, Польща;
- браузер Google Chrome;
- Інтернет з’єднання LTE (12 Мб/с за секунду).

Результат показує, що більшість запитів зі сторінки відсилаються на зображення (57,8%), на стилі (18,3%) та на JavaScript (16,5%).

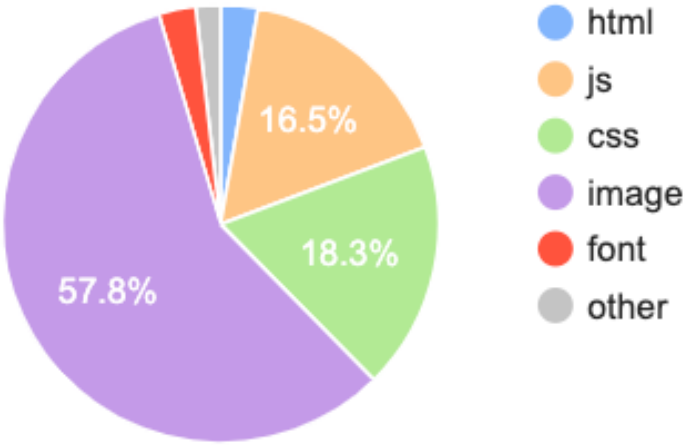

Рисунок 1.6 – Діаграма відсоткового відношення запитів по типу до загальної кількості запитів

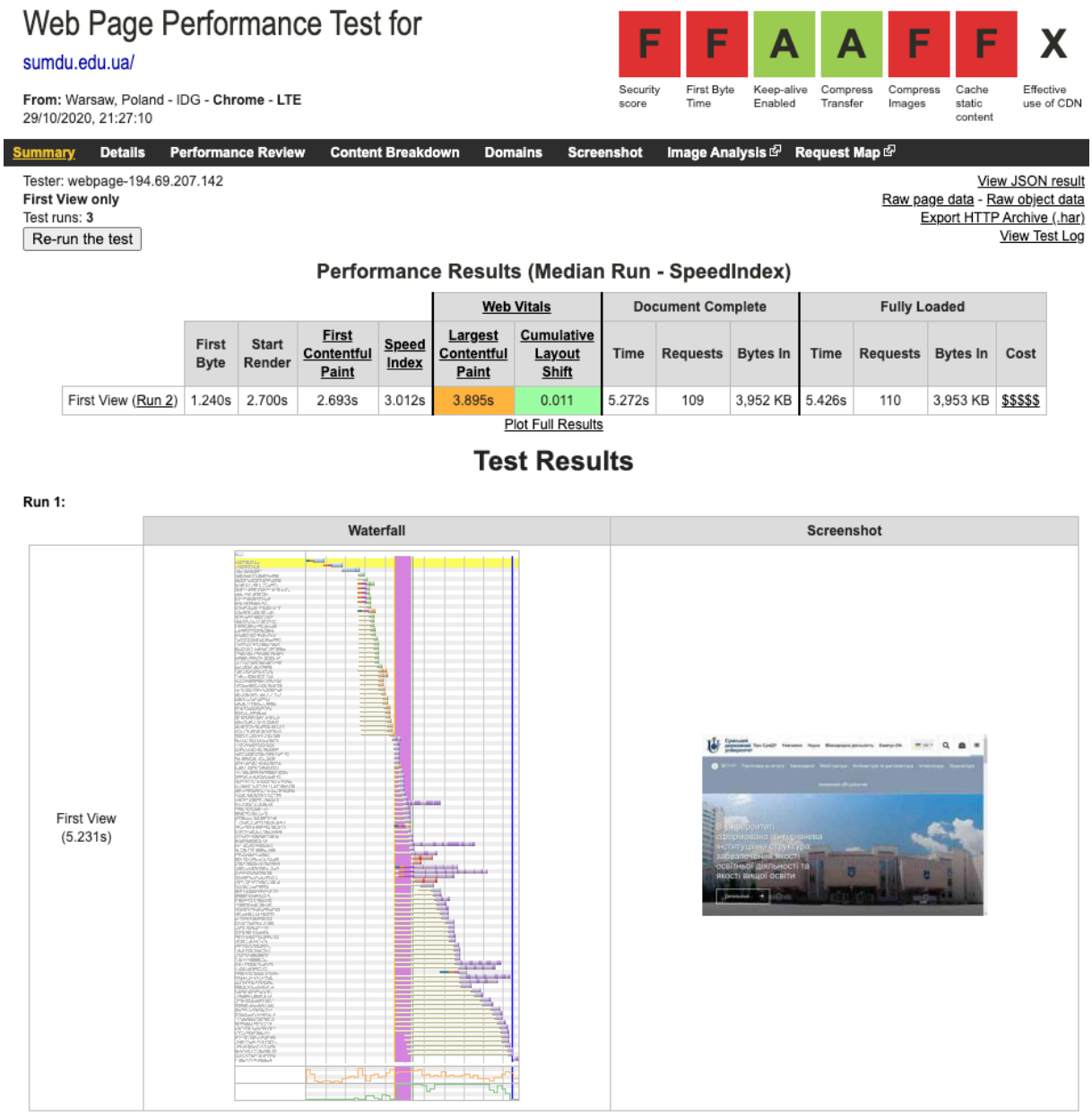

Рисунок 1.7 – Аналіз головної сторінки СумДУ сервісом WebPagetest.org

Згідно результату можна свідчити, що перший контент відобразився за 2,693 секунди. Повне відображення за 5,426 секунди. Було надіслано 110 запитів зі сторінки. Веб-додаток використовує наступну шкалу:

- A+;
- A;
- B;
- C;
- D;

- E;
- F.

Згідно даній шкалі A+ найкраща оцінка, а F – найгірша. Згідно результатам аналізу головна сторінка СумДУ отримала наступні оцінки:

- безпека – F;
- відображення першого байту змісту F;
- функція «тримати живим» - A;
- надсилання стискуваного змісту – A;
- стискування зображень – F;
- кешування статичного контенту – F.

1.5 Постановка задачі

Створити веб-додаток, який буде:

- аналізувати швидкість відображення іншого веб-додатку за посиланням;
- мати високу швидкість завантаження змісту;
- очікувано працювати у сучасних браузерах, таких як:
  - Google Chrome;
  - Firefox;
  - Safari.
- мати можливість налаштувати швидкість з’єднання;
- мати можливість відображати запити надіслані від веб-додатку, що аналізується;
- мати можливість аналізувати різними методами;
- мати можливість давати інформацію відображення на комп’ютері та на мобільному додатку.

2 МЕТОДИ ВИРІШЕННЯ ПОСТАВЛЕНИХ ЗАДАЧ

2.1 Веб-розробка

Веб-розробка - це процес програмування веб-додатків та програм, до яких користувачі можуть отримати доступ через Інтернет. Інтернет - це мережа, що включає безліч різних комп'ютерів, які взаємодіють між собою. Ці комп'ютери можна розділити на дві різні групи: сервери, що зберігають і надають його іншим та клієнти, які запитують цей вміст.

Процес перегляду основної веб-сторінки, наприклад, основний сайт СумДУ виглядає наступним чином:

1. Користувач вводить у рядок URL-адреси посилання https://sumdu.edu.ua
2. Браузер посилає запит до DNS.
3. DNS робить запит до Root Server.
4. Root Server повертає IP адресу TLD Server.
5. DNS робить запит до TLD Server.
6. TLD Server повертає IP адресу Authoritative Name Server.
7. DNS робить запит до Authoritative Name Server.
8. Authoritative Name Server повертає IP адресу сервера веб-додатка.
9. Браузер робить запит до IP адресу сервера веб-додатка.
10. Сервер веб-додатку повертає контент для користувача.

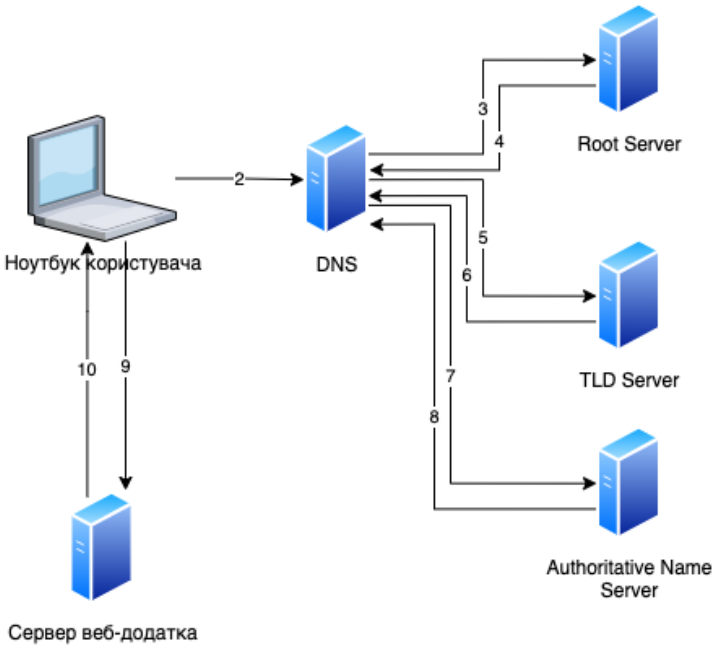

Рисунок 2.1 – Ланцюг перегляду веб-сторінки основного веб-додатку СумДУ

На кожному з цих етапів відбувається кешування результату, тобто не кожен запит до веб-додатку відбувається навіть через Root Server, бо DNS або браузер зберігають локально IP адресу сервера веб-додатка.

Root Server – це сервер у ланцюгу DNS, який відповідає за роботу кореневої зони DNS, тобто відповідні за трансляцію IP адресів доменів верхнього рівня: а саме UA, COM, ORG і так далі. Згідно з даними Root Server Technical Operations Assn [14] в Україні знаходиться 10 Root серверів: 7 у Києві, 1 у Одесі, 1 у Харкові, 1 у Луганську.

TLD Server – це сервер найвищого рівня доменних імен (наприклад UA).

Authoritative Name Server – це сервер який безпосередньо повертає IP адресу веб-додатку, що запитує користувач.

```
→  ~ host -t ns sumdu.edu.ua
sumdu.edu.ua name server ns1-09.azure-dns.com.
sumdu.edu.ua name server ns4-09.azure-dns.info.
sumdu.edu.ua name server ns3-09.azure-dns.org.
sumdu.edu.ua name server ns2-09.azure-dns.net.
```

Рисунок 2.2 – Список Authoritative Name Server сторінки Сумського Державного Університету

```
→  ~ host sumdu.edu.ua
sumdu.edu.ua has address 40.68.5.128
sumdu.edu.ua mail is handled by 10 mail.sumdu.edu.ua.
```

Рисунок 2.3 – IP-адреса сторінки Сумського Державного Університету

2.2 Клієнтська частина

На сьогодення існує велика кількість фреймворків та бібліотек для програмування клієнтської частини веб-додатку.

React.js - це бібліотека JavaScript з відкритим кодом, яка використовується для побудови веб та мобільних інтерфейсів. Застосовується для обробки всіх запитів програми для будь-якого веб або мобільного додатку. Він також використовується для повторного використання компонентів інтерфейсу. React дозволяє розробникам створювати веб-програми, які можуть змінювати дані на екрані користувача без перезавантаження сторінки веб-додатку, таким чином клієнту чи користувачу приємніше та зручніше користуватися сайтом. Головні переваги полягають в тому, що ця бібліотека легко масштабується, проста у використанні та швидка у дії. Ось декілька прикладів веб-додатків, які побудовано за допомогою React: facebook.com, dropbox.com, Airbnb.com, Netflix.com, reddit.com, bbc.com.

Create React App - це інструмент, створений розробниками у Facebook, щоб допомогти створювати веб-додатки, використовуючи React. Це позбавляє розробника від трудомісткого налаштування. Розробник

може просто запускати одну команду, яка створює додаток для редагування, налаштовує та створює елементи, які потребує розробник.

Додаток Create React App не обробляє бізнес логіку, не використовує бази даних, він лише створює шаблон React додатка, тому розробник може використовувати його з будь-якою серверною базою. Create React app використовує Babel та Webpack. [11]

Babel - це набір інструментів, який в основному використовується для перетворення коду ECMAScript 2015+ у зворотну сумісну версію JavaScript у поточних та старих браузерах або середовищах. [12]

Webpack - це конструктор модулів. Це важливо розуміти, оскільки Webpack не запускається під час веб-додатку, він працює під час розробки. [13] Webpack - це інструмент, за допомогою якого розробник використовує конфігурацію, щоб налаштувати, як завантажувати конкретні модулі чи файли. Розробник описує налаштування Webpack, як завантажувати файли JavaScript, або як повинені завантажуватися CSS тощо. Потім, коли розробник запускає його, він переходить у точку входу і проходить майбутній веб-додаток вгору-вниз і з’ясовує, що саме потрібно користувачу зараз, в якому порядку він це потребує, і від чого залежить кожна частина коду чи стилів. Потім Webpack створює пакети - якомога менше, максимально оптимізованих, які він включає як сценарії у веб-додаток.

2.3 Серверна частина

Node.js - надзвичайно потужна платформа на основі JavaScript, яка використовується для розробки додатків для онлайн-чату, веб-сайтів для потокового відео, односторінкових програм та багатьох інших веб-програм. Побудований на JavaScript V8 Engine від Google Chrome, він використовується великими та відомими компаніями (Netflix, Paypal, NASA та Walmart).

Node.js є відкритим і повністю безкоштовним, ним користуються тисячі розробників по всьому світу. Це приносить цьому інструменту безліч переваг, що робить його кращим вибором, ніж інші платформи на стороні сервера, такі як Java або PHP.

Усі API бібліотеки Node.js є асинхронними (тобто не блокуються), тому сервер на базі Node.js не чекає, поки API поверне дані. Сервер викликає API, і якщо дані не повертаються, сервер переходить до наступного API, модуль подій Node.js допомагає серверу отримати відповідь від попереднього виклику API. Сервер Node.js реагує неблокуючим способом, що робить його дуже масштабованим на відміну від традиційних серверів, які створюють обмежені потоки для обробки запитів.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Створення серверної частини

Щоб почати роботу потрібно створити проект за допомогою команди npm init та вказати відповідні налаштування. Список бібліотек, що використовувалися:

- express - це мінімальна та гнучка бібліотека для веб-додатків Node.js, яка забезпечує надійний набір функціоналу для опрацювання запитів за протоколом HTTP;
- lodash - це бібліотека функцій JavaScript, яка допомагає програмістам писати більш зручний код, який легше підтримувати за рахунок винесення найбільш використовуваних функцій у бібліотеку;
- moment – це бібліотека для зручної маніпуляції часу та дати, що дозволяє розробникам писати легший код для розуміння при взаємодії с часом та датами;
- puppeteer - це бібліотека API з протоколом DevTools для управління Chrome або Chromium. Зазвичай браузер без графічного відображення, але його можна налаштувати на роботу Chrome або Chromium з повним графічним відображення. Бібліотека використовуємо Chrome для імплементації, але розробники використовують JavaScript для взаємодії. Puppeteer – це офіційна бібліотека команди Google Chrome для взаємодії з Chrome без графічного відображення. Дана бібліотека може бути не самою швидкісною, оскільки запускає екземпляр Chrome при роботі з даною бібліотекою. Це один з кращих способів для тестування веб-додатків автоматизовано та один з доступних методів запуску браузера без графічного відображення;
- cors – це бібліотека для використання механізму CORS (механізм, який дозволяє запитувати ресурси, які знаходяться на сервері

веб-додатку з іншого домену за межами домену, з якого подано перший запит);

- @types/moment, @types/puppeteer – це бібліотеки типів для вищезгаданих бібліотек Moment та Puppeteer. Бібліотеки типів дозволяють розробнику швидше та зручніше програмувати.

```
→  diploma-master-api git:(master) cat package.json
{
  "name": "diploma-master-api",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node ./index.js"
  },
  "author": "Dmytro Shepeliev",
  "license": "ISC",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.17.1",
    "lodash": "^4.17.20",
    "moment": "^2.29.1",
    "puppeteer": "^5.5.0"
  },
  "devDependencies": {
    "@types/moment": "^2.13.0",
    "@types/puppeteer": "^5.4.0"
  }
}
```

Рисунок 3.1 – Вміст файлу package.json

Для запуска сервера використовується бібліотека Express.js. На першому рядку імпортується бібліотека express за допомогою команди require, на другому рядку створюється сервер, на третьому рядку імпортується бібліотека cors, на п’ятому рядку імпортується так званий route – тобто функція, яка буде виконуватися на певний URL сервера. Далі

на сьомому рядку до створеного раніше сервера додаються заголовки CORS. На дев'ятому рядку можна побачити створення URL за протоколом HTTP GET /api/get-page-load-time. Дана функція приймає Request та Response. У об'єкті Request зберігаються данні користувача такі як: HTTP заголовки, IP-адреса та параметри. Об'єкт Response відповідний за відповідь користувачу, а саме встановлює HTTP заголовки відповіді. Даний URL відповідає форматом JSON, повертаючи результат аналізу швидкості веб додатку. У самому кінці файлу можна побачити запуск сервера на 4000 порту.

```
const express = require("express");
const app = express();
const cors = require("cors");

const { getPageLoadTime } = require("./routes/getPageLoadTime");

app.use(cors());

app.get("/api/get-page-load-time", async (req, res) => {
  const { url, preset } = req.query;

  const results = await getPageLoadTime(url, preset);

  return res.json({ results });
});

app.listen( port: 4000,  callback: () => {
  console.log("[API] Listening on port 4000");
});
```

Рисунок 3.2 – Зображення індексного файлу сервера

3.2 Алгоритми аналізу

Алгоритми аналізу базуються на подіях JavaScript або на кількості HTTP запитів, які знаходяться в активному стані. Алгоритми аналізу

знаходяться у функції getPageLoadTime. Функція використовує наступні методи тестування:

- domcontentloaded;
- load;
- networkidle2;
- networkidle0.

3.2.1 Метод domcontentloaded

Метод domcontentloaded вважає завантаження завершеним, коли запускається подія DOMContentLoaded. Подія DOMContentLoaded – це подія, яка запускається, коли початковий документ HTML був повністю завантажений та проаналізований браузером. Важливо помітити, що подія не чекає завантаження стилів, зображень та дочірніх iframes.

3.2.2 Метод load

Метод load вважає завантаження завершеним, коли запускається подія load. Подія load – це подія, яка запускається, коли завантажується вся сторінка, включаючи всі залежні ресурси, такі як таблиці стилів та зображення, дочірні iframes тощо.

3.2.3 Метод networkidle2

Метод networkidle2 вважає завантаження завершеним, коли веб-додаток має у активних запитах не більше ніж 2 запити.

3.2.4 Метод networkidle0

Метод networkidle0 вважає завантаження завершеним, коли веб-додаток не має активних запитів.

3.2.5 Порівняльна таблиця методів

Назва метода	Базується на запитах	Базується на подіях JavaScript	Гарантує можливість використовувати веб-додаток
Метод domcontentloaded	НІ	ТАК	НІ
Метод load	НІ	ТАК	ТАК

Метод networkidle2	ТАК	НІ	ТАК
Метод networkidle0	ТАК	НІ	ТАК

Таблиця 3.1 – Порівняльна таблиця методів аналізу

Згідно з таблиці можна зробити висновок, що методи, на які потрібно базуватися є load, networkidle2, networkidle0, адже саме вони гарантують можливість використовувати веб-додаток. Саме в такому порядку вони гарантують завантаження, адже це залежить від деяких факторів, наприклад, характер веб-сайту, якщо це лендінг-сторінка тоді найкращим буде метод load або networkidle2, а якщо це високонавантажений веб-додаток з великою кількістю можливих взаємодій з користувачем, тобто HTTP запити є важливими для функціоналу, а не лише відображенням зображення, то єдиним методом є networkidle0.

Можна розглянути наступні приклади:

- Для головної сторінки СумДУ найкращим методом послугує load, адже більшість HTTP запитів є стилями та зображеннями.
- Для використання пошти Google найкращим методом послугує networkidle0, адже більшість HTTP запитів повертає данні (повідомлення, листи, деталі календаря, данні користувачів, які надсилають листи, фільтри користувача і так далі). В даному випадку зображення та стилі завантажуються швидше, ніж дані, тому метод load та networkidle2 спрацює швидше, ніж деякі з HTTP запитів повернуть данні, наприклад, список електронних листів, тобто веб-додатком буде неможливо користуватися з функціональної точки зору.

Додатковим висновком є те, що кожен сайт перед аналізом різними методами потрібно віднести до певної категорії взаємодій: інформаційної (наприклад, лендінг-сторінка) – веб-додаток, де користувачі отримують статичну інформацію, та функціональної – веб-додаток, де користувачі отримують динамічну інформацію, яка залежить від факторів (наприклад, сайт новин). Зрозумівши, до якої категорії відноситься певний веб-додаток – потрібно зробити акцент на один з методів.

3.3 Обмеження швидкості завантаження

Однією з вимог до даною інформаційної є можливість аналізувати додаток при різній швидкості Інтернет. Обрано 4 стандарти швидкості інтернет: 100мб/с (WiFi), 4G, 3G, 2G.

```
const NETWORK_PRESETS = {
  "2G": {
    offline: false,
    downloadThroughput: 12500,
    uploadThroughput: 12500,
    latency: 300
  },
  "3G": {
    offline: false,
    downloadThroughput: 187500,
    uploadThroughput: 187500,
    latency: 100
  },
  "4G": {
    offline: false,
    downloadThroughput: 18750000,
    uploadThroughput: 18750000,
    latency: 20
  }
};
```

Рисунок 3.3 – Зображення індексного файлу сервера

На Рисунку 3.3 можна побачити налаштування швидкості для 2G, 3G та 4G протоколів, а саме:

- швидкість 2G – 0.1 Мб/с;
- швидкість 3G – 1.5 Мб/с;
- швидкість 4G – 45 Мб/с;

- швидкість WiFi – 100 Мб/с.

Згідно зі специфікацією протоколів максимальна швидкість вища, але це залежить від покриття зв’язку та від багатьох інших факторів, тому були взяті більш реальні данні згідно зі статтею «Download Speeds: What Do 2G, 3G, 4G & 5G Actually Mean?» [15].

Реальні данні у оператора Vodafone UA у місті Суми навіть нижчі, на рисунку знизу наведені результати заміру швидкості сервісом speedtest.net. Результатом є 31.19 Мб/с.

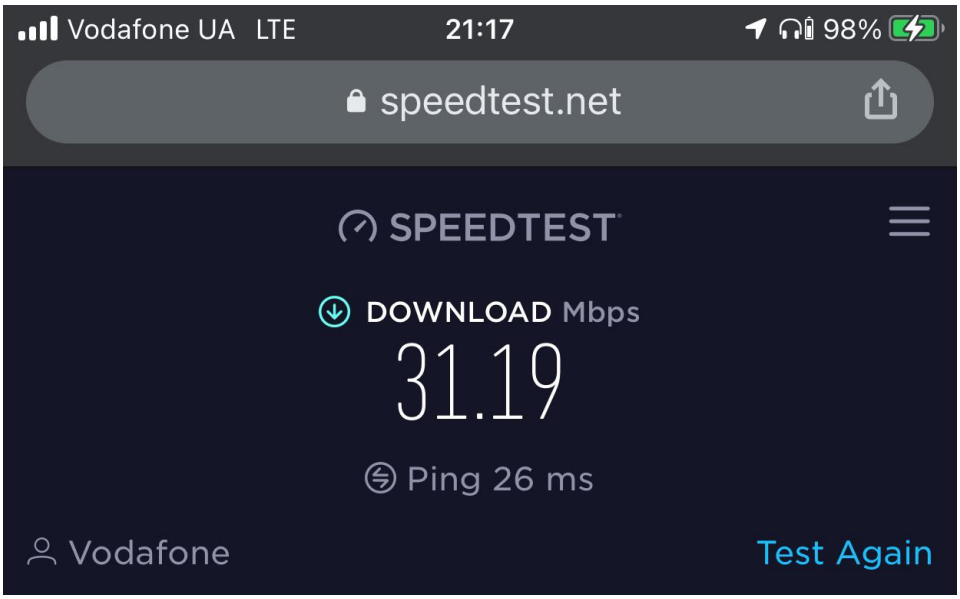

Рисунок 3.4 – Результат заміру швидкості сервісом speedtest.net у місті Суми, Україна. Оператор Vodafone UA

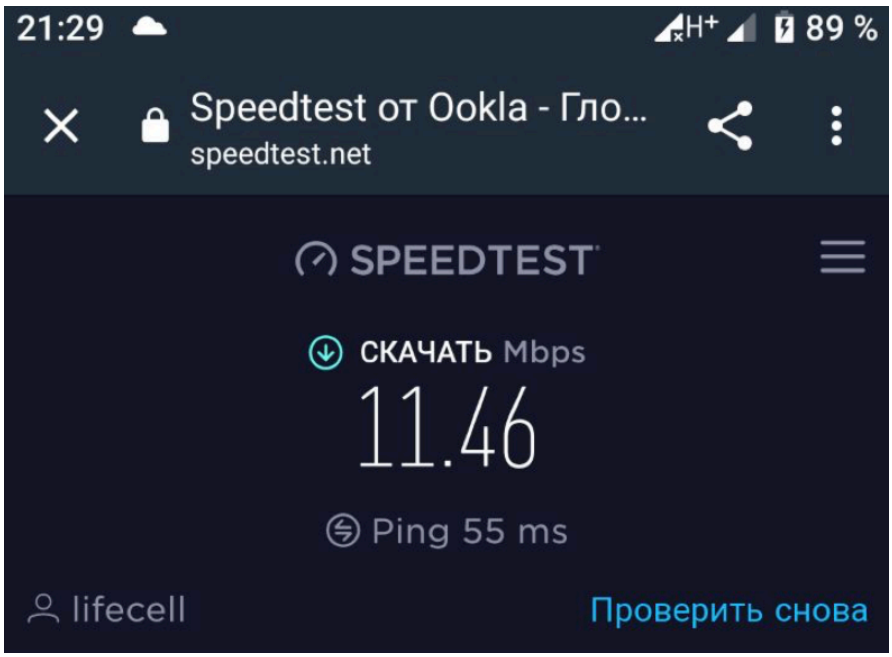

Рисунок 3.5 – Результат заміру швидкості сервісом speedtest.net у місті Суми, Україна. Оператор lifecell

Рисунок 3.6 – Результат заміру швидкості сервісом speedtest.net у місті Суми, Україна. Оператор Kyivstar

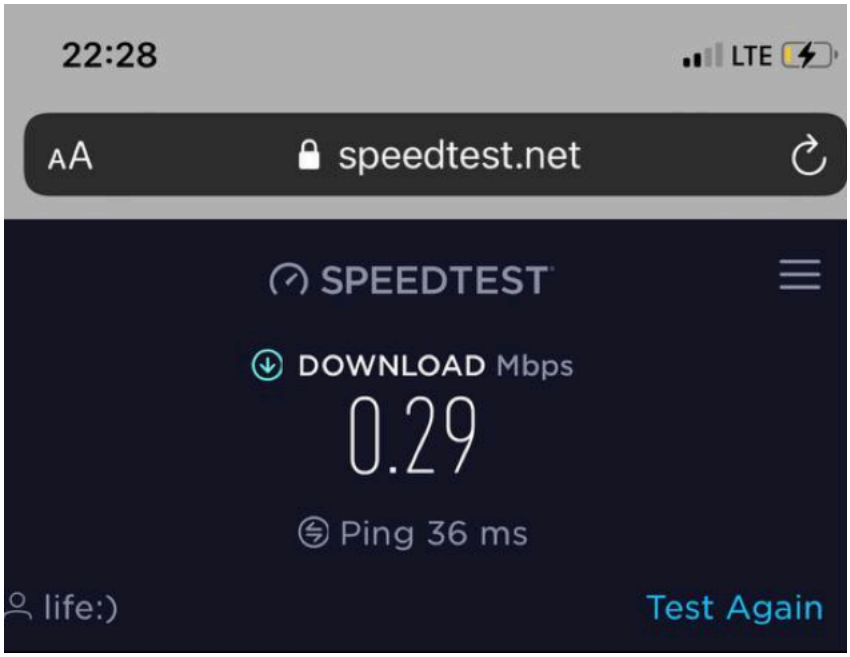

Рисунок 3.7 – Результат заміру швидкості сервісом speedtest.net у 10 кілометрах від міста Мінськ, Білорусь. Оператор life:)

Отримані данні:

- 31,19 Мб/с (Vodafone, місто Суми, Україна);
- 11,46 Мб/с (Lifecell, місто Суми, Україна);
- 2,44 Мб/с (Kyivstar, місто Суми, Україна);
- 0,29 Мб/с (life:, 10км від міста Мінськ, Білорусь).

Дані тестування показують, що навіть мінімальні значення не досягаються, тому важко зробити висновок реальних даних, адже за результатом дослідження можливі значення для швидкості 4G можуть не досягати стандарту 3G.

3.4 Віртуальні пристрої для аналізу швидкості веб-додатків

Бібліотека Puppeteer дозволяє відкривати браузер без графічної частини для деяких мобільних пристроїв та планшетів. Список пристроїв у Додатку А. Також браузер відкриває веб-додатків з характеристиками по-замовчування 800 пікселей висота та 600 пікселей широта. Для даного дослідження було також обрані налаштування мобільного пристрою iPhone X з наступними налаштуваннями:

- заголовок HTTP User Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 11_0 like Mac OS X) AppleWebKit/604.1.38 (KHTML, like Gecko) Version/11.0 Mobile/15A372 Safari/604.1;
- висота 812 пікселей;
- широта 375 пікселей.

3.5 Алгоритм виконання програми аналізу

Програма запускає асинхронно 2 задачі: аналіз на мобільному пристрої та на комп'ютері. Кожна з цих задач розподіляється на 4 підзадачі, а саме: кожен з методів аналізу використовує алгоритм. Алгоритм полягає в наступному:

1. Запустити браузер.
2. Включити режим перехвату HTTP запитів.
3. Включити або вимкнути обмеження швидкості.
4. Симуляція мобільного пристрою, якщо потрібно.
5. Налаштування перехвату HTTP запитів.
6. Запам'ятати час до відкривання сторінки.
7. Встановлення метода аналізу.
8. Очікування відкриття сторінки.
9. Запам'ятати час відкриття сторінки.
10. Закрити браузер.

Вищенаведений алгоритм виконується 2 х 4 = 8 разів, тобто 2 рази на пристрів та 4 метода на кожен метод. Вхідними параметрами до цього алгоритму є URL веб-додатку, що аналізується та обмеження швидкості (2G, 3G, 4G або WiFi). Результатом виконаного алгоритму є час за який виконується певний метод та список запитів, який надсилає веб-додаток.

3.6 Створення клієнтської частини

Для створення проекту за допомогою react-creact-app потрібно запустити команду: npx create-react-app diploma-master-front.

Список бібліотек, що використовувалися:

Рисунок 3.8 – Вміст файлу package.json

- @material-ui/core – це бібліотека стилів для використання Material UI набору компонентів для користувачів. Специфікація для даних компонентів розроблена компанією Google;
- @material-ui/icons – це бібліотека Material UI, які містять іконки;
- @testing-library/jest-dom, @testing-library/react, @testing-library/user-event – це набір бібліотек для тестування React, DOM елементів та подій користувача (симуляція вводу тексту і так далі);
- axios – це бібліотека для можливості надсилати HTTP запити;

- lodash - це бібліотека функцій JavaScript, яка допомагає програмістам писати більш зручний код, який легше підтримувати за рахунок винесення найбільш використовуваних функцій у бібліотеку;
- React, React DOM – це бібліотека JavaScript з відкритим кодом, яка використовується для побудови веб та мобільних інтерфейсів;
- react-scripts – це бібліотека, що налаштовує середовище розробки та запускає сервер, а також реалізує так зване Hot Module Reloading (технологія, що дозволяє оновлювати код у веб-додатку без перезапуску сервера);
- recharts – це бібліотека, що дозволяє швидко та легше створювати графіки, графічно відображуючи їх за допомогою SVG;
- web-vitals – це бібліотека, що реалізує ініціативу Google для створення звітів щодо характеристик веб-додатків, таких як швидкість відображення і так далі.

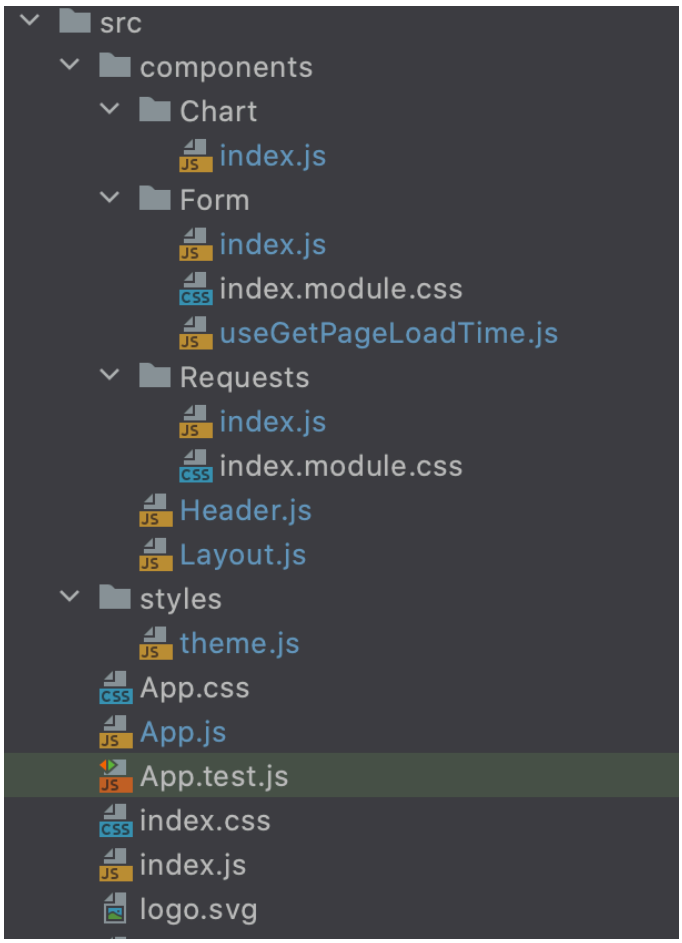

Рисунок 3.9 – Структура клієнтської частини

На Рисунку 3.9 зображена структура клієнтської частини, а саме:

- index.js – це файл, який відповідає за старт веб-додатку, додаючи дерево React до DOM;
- index.css – це файл, що містить глобальні стили, які використовуються для всього проекту, наприклад, розмір та тип шрифту;
- logo.svg – це зображення логотипу сайту;
- App.js – це основний компонент веб-додатку, який містить у собі всі інші компоненти;
- App.css – це файл стилів для компоненту App;
- styles/theme.js – це файл, що містить налаштування для теми для бібліотеки Material UI;

- components/Layout.js – це файл, який містить у собі компонент макету, який містить у собі заголовок та форму;
- components/Header.js – це файл, який містить у собі компонент заголовку;
- components/Form/index.js – це файл, який містить у собі текстове поле для вводу URL веб-додатку, що аналізується, випадаючий список з обмежуванням швидкості з'єднання, компонент, що відображає прогрес аналізу, графіки результату тестування та список запитів, що надсилає веб-додаток, що аналізується;
- components/Form/index.module.css – це файл, який містить у собі стилі для компоненту форми;
- components/Form/useGetPageLoadTime.js – це файл, який містить у собі логіку HTTP запитів та дані результатів аналізу;
- components/Requests/index.js – це файл, який містить у собі компонент, який відображає список HTTP запитів, які надсилає веб-додаток, що аналізується;
- components/Requests/index.css – це файл, який містить у собі стилі для компоненту Requests.

3.7 Приклад використання

Налаштування для аналізу головної сторінки сайту Сумського державного університету наступні: швидкість інтернету 4G, симуляція аналізу мобільної платформи та комп'ютера.

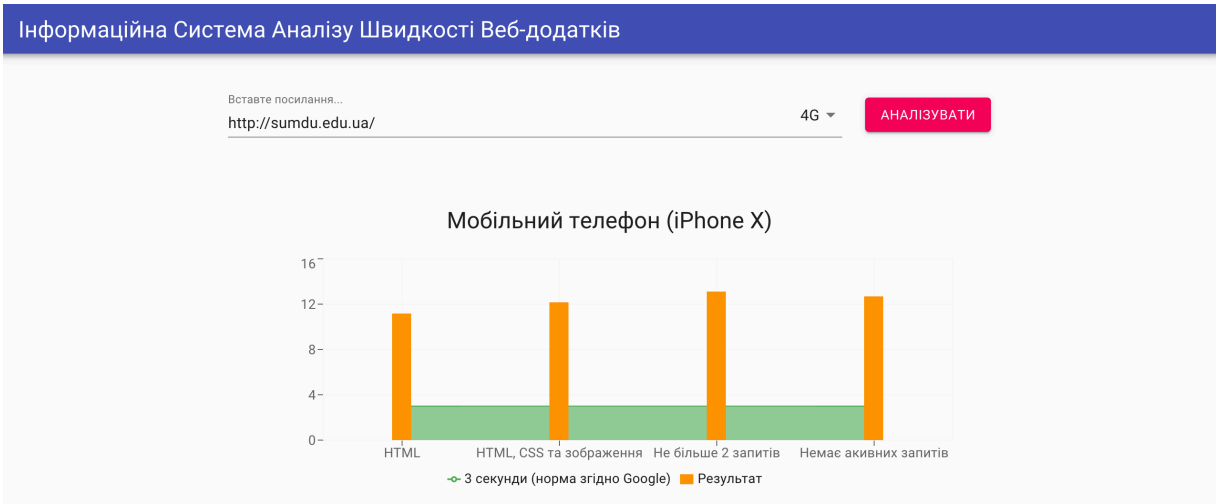

Рисунок 3.10 – Аналіз головної сторінки СумДУ, симулюючи мобільний телефон

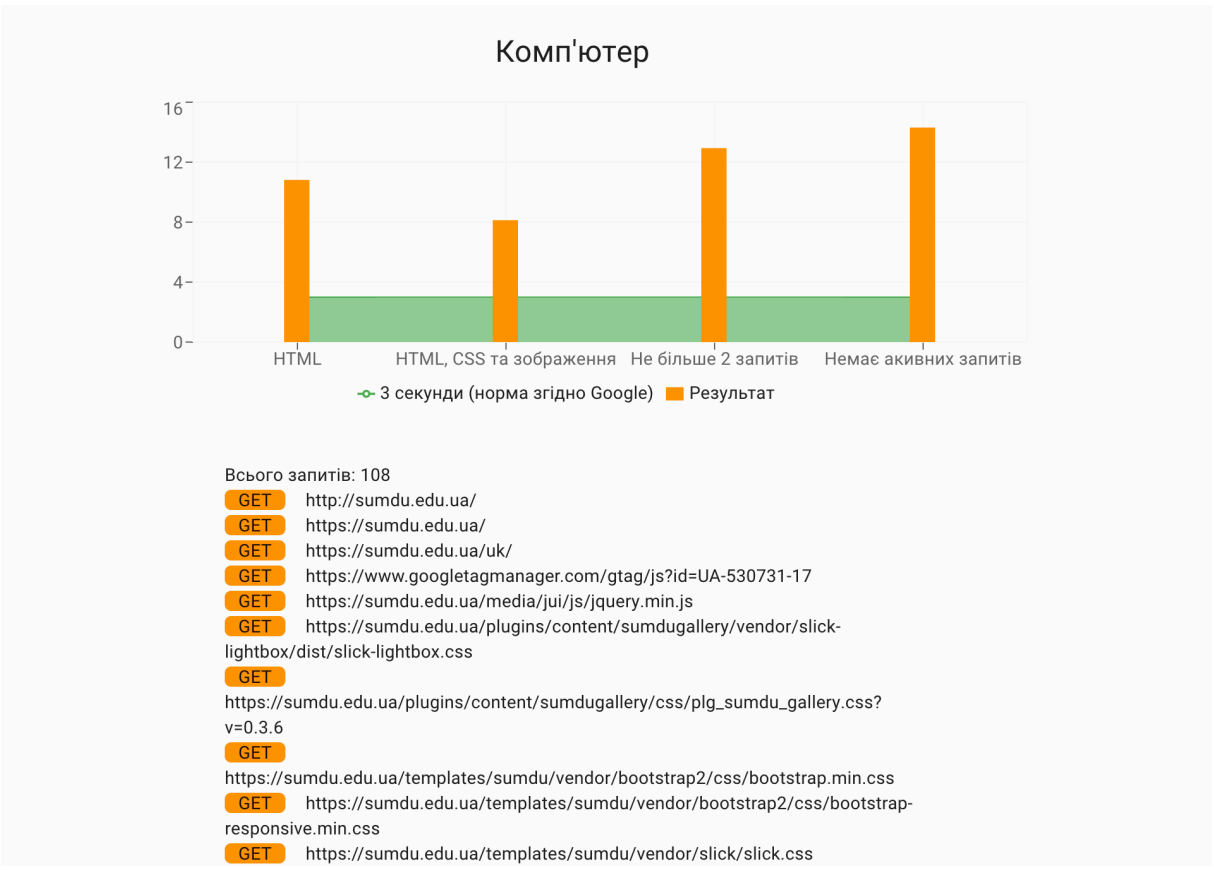

Рисунок 3.11 – Аналіз головної сторінки СумДУ, симулюючи комп’ютер та список запитів

Можна зробити наступні висновки:

- швидкість завантаження на мобільному телефоні:
    - метод domcontentloaded – 11,169 секунд;
    - метод load – 12,161 секунд;
    - метод networkidle2 – 13,108 секунд;
    - метод networkidle0 – 12,682 секунд;
- швидкість завантаження на комп’ютері:
    - метод domcontentloaded – 10,817 секунд;
    - метод load – 8,121 секунд;
    - метод networkidle2 – 12,937 секунд;
    - метод networkidle0 – 14,308 секунд.

Веб-додаток, що аналізується, відправив 108 запитів.

3.8 Відгук про використання веб-додатку

Відгук про використання веб-додатку, що аналізує залишив Альберт Фрімпонг (Технічний директор компанії Sourcify): «Завдяки цьому веб-додатку ми змогли проаналізувати при різних обмежувачах швидкостей результат відображення нашої головної сторінки. Цей результат дуже важливий для нас, адже головна сторінка нашої платформи пропонує певний сервіс. У списку HTTP запитів ми побачили запит стороннього програмного забезпечення, який вже не використовуємо. Завдяки цьому швидкість відображення веб-додатку зросла на половину секунди».

ВИСНОВКИ

Проаналізовано важливість швидкого відображення змісту веб-додатку. Проаналізовані існуючі сервіси та так звані сайти-конкуренти для аналізу швидкості веб-додатків. Розглянуто приклади аналізу веб-додатку на сучасних платформах аналізу будь-якого сайту. Проаналізовано переваги технологічного стеку для створення веб-додатку, що буде аналізувати сайти. Обрані технології для програмування інформаційної системи. Створена інформаційної система аналізу веб-додатків. Розроблено алгоритм аналізу веб-додатків з використанням 4 методів. Проаналізована швидкість мобільного інтернету у місті Суми. Проаналізована головну сторінку Сумського державного університету. Інформаційна система запропонована для використання та отримано відгук. Було проаналізовано становище веб-додатків для аналізу інших веб-додатків, та було поставлено задачу покращити інструменти, які вирішують дану проблему. Досліди та експерименти показали, що створена інформаційна система може з легкістю замінити існуючі та бути інструментом, який пропонує більше можливостей (такі як обмеження швидкості і так далі).

СПИСОК ЛІТЕРАТУРИ

1. ДСТУ 2392-94 Інформація та документація. Базові поняття

2. Современный учебник JavaScript [Electronic Resource] – Access Mode: https://learn.javascript.ru/intro.

3. TypeScript: JavaScript Development at Application Scale [Electronic Resource] – Access Mode: https://web.archive.org/web/20160303224243/http://blogs.msdn.com/b/somasegar/archive/2012/10/01/typescript-javascript-development-at-application-scale.aspx.

4. Insights. Ideas. Inspiration [Electronic Resource] – Access Mode: https://www.thinkwithgoogle.com/consumer-insights/consumer-trends/mobile-site-load-time-statistics/.

5. How Loading Time Affects Your Bottom Line [Electronic Resource] – Access Mode: https://neilpatel.com/blog/loading-time/.

6. Site speed: case studies, tips and tools for improving your conversion rate [Electronic Resource] – Access Mode: https://econsultancy.com/site-speed-case-studies-tips-and-tools-for-improving-your-conversion-rate/.

7. Ecommerce Charts: How conversion rates correlate to page load times [Electronic Resource] – Access Mode: https://www.marketingsherpa.com/article/chart/conversion-correlate-page-load-time.

8. About PageSpeed Insights [Electronic Resource] – Access Mode: https://developers.google.com/speed/docs/insights/v5/about.

9. How to Analyze Website Speed Test Results [Electronic Resource] – Access Mode: https://www.pingdom.com/blog/how-to-analyze-website-speed-test-results.

10. About WebPagetest.org [Electronic Resource] – Access Mode: https://www.webpagetest.org/about.

11. Create a New React App [Electronic Resource] – Access Mode: https://reactjs.org/docs/create-a-new-react-app.html.

12. What is Babel? [Electronic Resource] – Access Mode: https://babeljs.io/docs/en/.

13. What is Webpack and why should I care? [Part 1][Introduction] [Electronic Resource] – Access Mode: https://medium.com/the-self-taught-programmer/what-is-webpack-and-why-should-i-care-part-1-introduction-ca4da7d0d8dc.

14. Root Servers [Electronic Resource] – Access Mode: Root Server Technical Operations Assn. www.root-servers.org.

15. Download Speeds: What Do 2G, 3G, 4G & 5G Actually Mean? [Electronic Resource] – Access Mode: https://kenstechtips.com/index.php/download-speeds-2g-3g-and-4g-actual-meaning.

ДОДАТОК А

Blackberry PlayBook, Blackberry PlayBook landscape, BlackBerry Z30, BlackBerry Z30 landscape, Galaxy Note 3, Galaxy Note 3 landscape, Galaxy Note II, Galaxy Note II landscape, Galaxy S III, Galaxy S III landscape, Galaxy S5, Galaxy S5 landscape, iPad, iPad landscape, iPad Mini, iPad Mini landscape, iPad Pro, iPad Pro landscape, iPhone 4, iPhone 4 landscape, iPhone 5, iPhone 5 landscape, iPhone 6, iPhone 6 landscape, iPhone 6 Plus, iPhone 6 Plus landscape, iPhone 7, iPhone 7 landscape, iPhone 7 Plus, iPhone 7 Plus landscape, iPhone 8, iPhone 8 landscape, iPhone 8 Plus, iPhone 8 Plus landscape, iPhone SE, iPhone SE landscape, iPhone X, iPhone X landscape, iPhone XR, iPhone XR landscape, JioPhone 2, JioPhone 2 landscape, Kindle Fire HDX, Kindle Fire HDX landscape, LG Optimus L70, LG Optimus L70 landscape, Microsoft Lumia 550, Microsoft Lumia 950, Microsoft Lumia 950 landscape, Nexus 10, Nexus 10 landscape, Nexus 4, Nexus 4 landscape, Nexus 5, Nexus 5 landscape, Nexus 5X, Nexus 5X landscape, Nexus 6, Nexus 6 landscape, Nexus 6P, Nexus 6P landscape, Nexus 7, Nexus 7 landscape, Nokia Lumia 520, Nokia Lumia 520 landscape, Nokia N9, Nokia N9 landscape, Pixel 2, Pixel 2 landscape, Pixel 2 XL, Pixel 2 XL landscape.