МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
червня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 - Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Інформаційна система моніторингу статистичних характеристик вебсайтів структурних підрозділів Сумського державного університету»
здобувача групи ІН – 94-1 Богатиренка Дмитра Віталійовича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Дмитро БОГАТИРЕНКО
(підпис)

Керівник,
кандидат технічних наук, доцент Віктор АВРАМЕНКО _____
(підпис)

Суми – 2023

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН – 94-1 Богатиренка Дмитра Віталійовича

1. Тема роботи: «Інформаційна система моніторингу статистичних характеристик вебсайтів структурних підрозділів Сумського державного університету»
затверджую наказом по СумДУ від_____
2. Термін здачі здобувачем кваліфікаційної роботи *до 09 червня 2023 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Постановка й формування завдань дослідження.2) Огляд технологій, що використовуються для моніторингу статистичних характеристик вебсайтів. 3) Розробка інформаційної системи з моніторингу статистичних характеристик вебсайтів 4) Аналіз результатів.*
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 «____» _____ 20 ___ р.
Завдання прийняв до виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Постановка й формування завдань дослідження*		
2	*Огляд технологій, що використовуються для моніторингу статистичних характеристик вебсайтів*		
3	*Розробка інформаційної системи з моніторингу статистичних характеристик вебсайтів*		
4	*Аналіз отриманих результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 46 стр., 8 таблиць, 2 додатки, 18 використаних джерел.

Обґрунтування актуальності теми роботи – Тема кваліфікаційної роботи є актуальною, оскільки присвячена розв’язанню важливих практичних задач отримання, обробки і аналізу даних з вебсторінок. Через неоднорідність даних та їх надлишок існує потреба в ефективних і автоматизованих методах аналізу вебсторінок.

Об’єкт дослідження — система вебсайтів Сумського державного університету.

Мета роботи — розробка інформаційної системи для моніторингу статистичних характеристик вебсайтів структурних підрозділів Сумського державного університету і оцінки ефективності їхньої роботи.

Методи дослідження — статистичні методи аналізу та обробки даних, методи розпізнавання текстової інформації.

Результати — розроблено інформаційну систему, яка отримує статистичні характеристики вебсайтів структурних підрозділів Сумського державного університету, для аналізу та дослідження ефективність їхньої роботи. Статистичні характеристики отримані по реальним даним під час роботи вебсайтів структурних підрозділів Сумського державного університету.

ІНФОРМАЦІЙНА СИСТЕМА, СТАТИСТИЧНІ ХАРАКТЕРИСТИКИ, C#, .NET, ВЕБСАЙТИ, МАТЕМАТИЧНЕ ОЧІКУВАННЯ, СЕРЕДНЄ КВАДРАТИЧНЕ ВІДХИЛЕННЯ, КОЕФІЦІЄНТ НАПОВНЮВАНОСТІ САЙТУ

ЗМІСТ

ВСТУП

У сучасному світі Інтернет виконує важливу роль як джерело інформації для різних галузей життя, включаючи бізнес, науку, медіа та фінанси. Завдяки широкому доступу до вебсторінок і розмаїттю інформації, потреба у зборі та аналізу даних з ресурсів мережі Інтернет стає все більш актуальною.

Однак, збір і обробка великих обсягів даних з вебсторінок може бути складним завданням, оскільки вони можуть мати різну структуру і форматування. Крім того, потенційні зміни в структурі вебсторінки можуть ускладнити процес автоматизації збору і аналізу даних.

Для ефективного збору і аналізу даних з вебсторінок існують різні методи та інструменти. Один із таких методів - автоматичний збір даних з вебсторінок, що дозволяє автоматично отримувати необхідні дані. Для ефективного автоматичного отримання даних використовуються спеціалізовані алгоритми та інструменти.

У цьому контексті програми для збору і аналізу даних з вебсторінок мають велике значення. Вони дозволяють автоматизувати процеси збору даних, що раніше вимагали багато часу та зусиль. Це особливо актуально для ситуацій, коли необхідно взаємодіяти з великим числом вебсторінок або виконувати регулярне оновлення інформації.

В результаті проведеної роботи розроблено програму, яка забезпечує збір даних з вебсторінок і їх подальший аналіз, з метою отримання статистичних характеристик, коефіцієнту наповненості вебсатів, середнє квадратичне відповіді від серверу. Програма використовує спеціалізовані алгоритми та методи для ефективного і точного отримання даних з вебсторінок, забезпечуючи швидкий та надійний доступ до інформації.

Для досягнення цієї мети використовуються різні методи дослідження, враховуються такі фактори, як ефективність, точність та швидкість збору і аналізу даних.

Дана робота складається зі вступу, інформаційного огляду, постановки задачі, вибір методу розв’язання поставленої задачі, опису програмного забезпечення інформаційної системи, висновків, списку використаних джерел та додатків.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Загальна інформація про вебсайти

HTML (HyperText Markup Language) є основною мовою, що використовується для створення вебсторінок. Вона передає структуру та семантику контенту, який потім браузери можуть правильно відображати. HTML використовується для створення різних елементів вебсторінки, таких як заголовки, параграфи, списки, зображення, таблиці, форми та багато іншого[1].

Структура HTML-документа має певну конвенцію. Зазвичай, вона складається з елементів, таких як <html>, <head> та <body>.

Елемент <html> є корневим елементом, що охоплює всі інші елементи документа. Елемент <head> містить інформацію про документ, таку як заголовок сторінки, метатеги та посилання на зовнішні ресурси.

Елемент <body> містить основний контент сторінки, який буде відображатися користувачам.

Одним із основних аспектів HTML є використання тегів. Теги визначають різні елементи на сторінці. Вони мають початковий тег <назва> та закриваючий тег </назва>. Наприклад, тег <h1> визначає найвищий рівень заголовка, а тег <p> використовується для створення параграфу тексту[2].

Крім основних тегів, HTML також має багато атрибутів, які надають додаткову інформацію або властивості елементам. Атрибути вказуються всередині відкриваючого тега і можуть використовуватися для задання стилів, властивостей або інших значень. Наприклад, атрибут href використовується для визначення посилання на сторінку у тезі Текст посилання.

HTML також підтримує вкладеність елементів, що означає, що елементи можуть бути вкладені один в одного. Наприклад, елемент може містити в

собі елементи <p>, <img> та інші. Це дозволяє розміщувати елементи у відповідних блоках та організовувати структуру сторінки.

Окрім структури та контенту, HTML також використовується у поєднанні з CSS (Cascading Style Sheets) для оформлення та стилізації вебсторінок. CSS дозволяє змінювати зовнішній вигляд елементів, таких як кольори, шрифти, розташування тощо, що робить сторінку більш привабливою та зручною для користувача[3].

CSS (Cascading Style Sheets) є мовою стилізації, що дозволяє задавати зовнішній вигляд вебсторінок, створених за допомогою HTML. Завдяки CSS, розробники можуть змінювати кольори, розміри, шрифти, розташування елементів та багато іншого, щоб досягти бажаного дизайну сторінки.

CSS використовує селектори для вибору елементів, які потребують стилізації. Селектори можуть бути назвами елементів (наприклад, h1, p, div), класами (наприклад, .клас), ідентифікаторами (наприклад, #ідентифікатор) або іншими атрибутами елементів[4].

Кожен селектор має блок властивостей, де вказуються конкретні стилі, що мають бути застосовані до вибраних елементів. Властивості CSS визначають, який аспект елемента потрібно змінити, наприклад, color для кольору тексту, font-size для розміру шрифту, background для фону елемента.

Кожна властивість має значення, які вказуються разом з властивістю. Значення можуть бути числовими, рядками або ключовими словами, залежно від властивості. Наприклад, для властивості font-family значення може бути 'Arial', "Helvetica", sans-serif, де sans-serif є ключовим словом.

CSS також підтримує спадкування стилів, що означає, що стилі, застосовані до батьківського елемента, можуть спадати на його дочірні елементи. Це допомагає зменшити кількість потрібних стилів та полегшує керування дизайном сторінки.

CSS також підтримує різні техніки та функції, такі як анімації, переходи, трансформації, гнучкі розташування (flexbox) та сітки (grid). Ці можливості дозволяють створювати динамічні та інтерактивні ефекти на вебсторінках.

Для використання CSS у вебсторінках, стилі можуть бути вкладені безпосередньо у HTML-документ у блоках <style> або зберігатися в окремих зовнішніх файлів CSS з розширенням .css, які потім підключаються до HTML-документу.

JavaScript (JS) є мовою програмування, яка використовується для надання інтерактивності та динамічності вебсторінкам[5]. В поєднанні з HTML та CSS, JavaScript дозволяє створювати вебдодатки, анімації, валідацію форм, взаємодію з користувачем та багато іншого.

JavaScript використовується для виконання різноманітних дій на вебсторінках. Це можуть бути такі речі, як зміна вмісту елементів на сторінці, обробка подій, взаємодія з сервером для отримання або відправлення даних, анімація елементів, перевірка та валідація форм, маніпуляція зображеннями та багато іншого.

Завдяки JavaScript можна здійснювати маніпуляції з елементами на сторінці, наприклад, змінювати їх вміст, стилі, розміри, показувати або ховати елементи, додавати анімацію та переходи між сторінками. Також можна створювати функції, що викликаються при виникненні певних подій, таких як клік мишею, натискання клавіші, завантаження сторінки тощо.

JavaScript також має можливість взаємодіяти з сервером за допомогою AJAX (асинхронний JavaScript і XML). Це дозволяє отримувати або відправляти дані на сервер без перезавантаження сторінки, що робить вебдодатки більш динамічними та ефективними.

Крім того, JavaScript підтримує роботу з кукі (cookies) та локальним сховищем (localStorage, sessionStorage), що дозволяє зберігати дані на боці

клієнта та використовувати їх при наступних відвідуваннях сторінок.

JavaScript є потужним і гнучким інструментом для веброзробки, що дозволяє створювати багатофункціональні та інтерактивні вебсторінки. Він широко використовується розробниками та має велику кількість ресурсів та бібліотек, які допомагають спростити процес розробки та розширити можливості JavaScript.

1.2 Клієнт-серверна архітектура

Клієнт-серверна архітектура є широко використовуваним підходом до розробки додатків та систем. Вона передбачає розділення функціональності між клієнтом (клієнтська частина) та сервером (серверна частина) та використання механізмів обміну даними між ними[6].

Клієнт - це програма або пристрій, який виконується на боці користувача. Він надає інтерфейс для взаємодії з додатком чи системою. Клієнт може бути веббраузером, мобільним додатком, настільною програмою або іншим програмним забезпеченням. Клієнтська частина відповідає за зборку та відображення даних, взаємодію з користувачем та передачу запитів до сервера.

Сервер - це комп'ютер або пристрій, який надає послуги, оброблює запити та постачає дані клієнтам. Серверна частина зазвичай відповідає за обробку бізнес-логіки, зберігання та керування даними. Вона відповідає за виконання запитів клієнтів, обробку даних та надання відповідей.

Обмін даними між клієнтом та сервером зазвичай здійснюється за допомогою протоколів передачі даних, таких як HTTP (Hypertext Transfer Protocol). Клієнт може ініціювати запит до сервера, відправляючи HTTP-запит, який містить необхідні параметри та дані. Сервер обробляє цей запит, виконує необхідні дії та надсилає відповідь клієнту, яка може містити запитані дані або повідомлення про статус операції.

Клієнт-серверна архітектура може мати різні варіації та рівні складності,

включаючи однорівневу, дворівневу та багаторівневу архітектури. У багаторівневій архітектурі можуть бути використані додаткові компоненти, такі як проксі-сервери, кеш-сервери та бази даних.

Клієнт-серверна архітектура дозволяє розподілити навантаження та виконувати функціональність на різних рівнях, забезпечуючи більшу масштабованість, безпеку та швидкість обробки даних. Вона є основою багатьох вебдодатків та систем[7].

Клієнт-серверна архітектура використовує різні методи для обміну даними між клієнтом та сервером. Основні методи, які використовуються в HTTP-протоколі, включають GET, POST, PUT, DELETE та PATCH[8]. Порівняння основних методів наведено в таблиця 1.1.

Таблиця 1.1 Основні методи

Метод	**Опис**
GET	Використовується для отримання даних з сервера.
POST	Використовується для відправки даних на сервер для обробки.
PUT	Використовується для оновлення існуючих даних на сервері.
DELETE	Використовується для видалення даних на сервері.
PATCH	Використовується для часткового оновлення даних на сервері.

Кожен з цих методів має свою власну призначену роль і використовується в різних ситуаціях. Основні відмінності між ними полягають у способі взаємодії з сервером та впливі на дані на сервері.

GET (отримання): GET-запит використовується для отримання даних з сервера. Він передає параметри запиту через URL-адресу. GET-запити повинні бути безпечними та не мають повинні мати побічних ефектів, тобто не змінювати

стан сервера. Вони часто використовуються для отримання вмісту сторінок, запитів до бази даних, викликів API тощо.

POST (відправлення): POST-запит використовується для відправки даних на сервер для обробки. Він передає дані у тілі запиту, яке може бути у форматі JSON, XML або іншому. POST-запити можуть мати побічні ефекти, змінюючи стан сервера або додаючи нові дані. Вони часто використовуються для створення нових ресурсів, відправлення форм, виконання операцій, які змінюють дані на сервері.

PUT (оновлення): PUT-запит використовується для оновлення (заміни) існуючих даних на сервері. Він використовується для внесення змін до вже існуючих ресурсів. При використанні PUT-запиту, повний ресурс повинен бути переданий в тілі запиту.

DELETE (видалення): DELETE-запит використовується для видалення даних на сервері. Він вказує серверу видалити вказаний ресурс. DELETE-запити повинні бути обережно використовувані, оскільки вони безповоротно видаляють дані.

PATCH (часткове оновлення): PATCH-запит використовується для часткового оновлення даних на сервері. Він передає тільки необхідні зміни до ресурсу, не вимагаючи відправки повного ресурсу. Цей метод дозволяє внести мінімальні зміни до існуючих даних.

Кожен з цих методів має свою роль і використовується в залежності від конкретних потреб додатку або системи. Важливо правильно вибирати метод відповідно до типу операції та його впливу на дані на сервері.

SSL (Secure Sockets Layer) є протоколом захищеної комунікації, що забезпечує шифрування та аутентифікацію даних між клієнтом та сервером в інтернеті[9]. Використовуючи SSL, можна забезпечити безпеку під час передачі конфіденційної інформації, такої як паролі, особисті дані, фінансова інформація

та інше.

Одним з головних аспектів SSL є шифрування даних. Коли клієнт встановлює з'єднання з сервером, SSL використовує криптографічні алгоритми для шифрування даних, що передаються між ними. Це означає, що навіть якщо зловмисник зможе перехопити ці дані, вони будуть зашифровані і не можуть бути прочитані без використання відповідного ключа.

SSL також забезпечує аутентифікацію, що дозволяє клієнту та серверу перевірити свою взаємну ідентичність. Це досягається за допомогою цифрових сертифікатів, які видані довіреними центрами сертифікації. Клієнт може перевірити валідність сертифіката сервера, що підтверджує, що він спілкується з вірним сервером, а не з підробкою. Це дозволяє уникнути атак "помилкового сервера" та забезпечує довіру між клієнтом та сервером.

Крім того, SSL забезпечує цілісність даних. Використовуючи хеш-функції, SSL гарантує, що дані під час передачі не були змінені зловмисниками. Якщо будь-які зміни виявляються під час процесу передачі, отримувач буде повідомлений про це.

Зараз SSL заміщується більш безпечним TLS (Transport Layer Security), який є його наступником. TLS зберігає всі основні принципи SSL, але працює на більш сучасних і безпечних алгоритмах шифрування та аутентифікації.

Використання SSL (або TLS) є важливим для забезпечення безпеки під час передачі даних через інтернет. Вебсайти, які використовують SSL, можна впізнати за наявністю протоколу "https://" у URL-адресі та іконки замка в браузері.

1.3 Статистичне моделювання

Статистичне моделювання використовується для розуміння та аналізу статистичних даних[10]. Воно базується на використанні математичних моделей, які допомагають описати залежності між змінними та зробити прогнози.

Основною метою статистичного моделювання є знаходження підтвердження гіпотез, виявлення взаємозв'язків між факторами та розробка моделей, які допомагають пояснити та передбачити явища.

Статистичне моделювання є важливою галуззю статистики, яка використовується для вивчення і прогнозування залежностей та взаємозв'язків між змінними у наборі даних. Воно дозволяє дослідникам і аналітикам використовувати математичні моделі для опису, аналізу та прогнозування різних явищ, які спостерігаються в реальному світі.

Одним з ключових показників у статистичному моделюванні є середнє квадратичне відхилення (standard deviation). Воно вимірює розсіювання або розкид значень навколо середнього значення[11]. Середньо-квадратичне відхилення є показником різноманітності або варіабельності даних. Чим більше значення середнього квадратичного відхилення, тим більша розсіяність даних.

Середнє квадратичне відхилення можна обчислити за допомогою наступної формули:

$$\bar{o} = \text{sqrt}((1/(N-1)) * \Sigma(x_i - \mu)^2)$$

позначення:

- $\bar{o}$ - середнє квадратичне відхилення,
- x_i - значення змінної,
- μ - середнє значення змінної,
- N - кількість значень змінної.

Статистичні моделі можуть бути простими або складними, в залежності від складності явища, яке досліджується. Вони можуть включати різні типи залежностей, такі як лінійні, нелінійні, просторові та інші. Для побудови статистичних моделей використовуються статистичні методи, які дозволяють оцінити параметри моделі на основі доступних даних.

Середнє квадратичне відхилення є мірою розсіювання або розкиду значень відносно середнього значення[12]. Воно використовується для вимірювання варіабельності даних та оцінки точності моделей.

Формула для обчислення середнього квадратичного відхилення вже була наведена в попередній відповіді. Важливо зазначити, що середнє квадратичне відхилення виражається в тих же одиницях, що й вихідні дані.

Середнє квадратичне відхилення є важливим інструментом для аналізу розподілу даних. Воно дозволяє оцінити, наскільки точно значення розподілені навколо середнього значення. Чим менше середнє квадратичне відхилення, тим ближче значення до середнього, і навпаки.

Середнє квадратичне відхилення також використовується для порівняння різних груп даних або моделей. Воно дозволяє визначити, яка група або модель має меншу варіабельність і, отже, є більш стабільною та передбачуваною.

В цілому, статистичне моделювання та середнє квадратичне відхилення допомагають розкрити складні залежності та зробити аналіз даних більш точним та об'єктивним. Вони є невід'ємними інструментами у сфері статистики та науки про дані для прийняття обґрунтованих рішень та прогнозування майбутніх подій.

1.4 Постановка задачі

Розробити алгоритм та комп’ютену програму для моніторингу вебсайтів структурних підрозділів Сумського державного університету з метою оцінки їхньої наповненості та отримання статистичних характеристик.

Для цього необхідно:

- вибір середовище розробки;
- провести аналіз актуальності розробки;
- обрати інструменти реалізації та середовище для розробки;
- обрати метод зчитування даних;
- обрати метод аналізу даних;

- обрати метод вимірювання тривалості обробки запиту;
- розглянути варіант запису результатів програми.

Для асинхронного зчитування даних з файлу повинно бути у форматі JSON. Зчитані дані повинні містити інформацію про шаблони пошуку (search values), дані для аналізу (data array) та посилання для перевірки (search links).

Після зчитування даних, програма повинна виконувати аналіз вебсторінок згідно з визначеними шаблонами пошуку та визначати наявність або відсутність заданих елементів на сторінках.

Для аналізу вебсторінок використовувати платформу .NET із використанням CSS селекторів.

Програма повинна вимірювати час відгуку вебсторінок, тобто час, який затрачається на їх завантаження та аналіз.

Після аналізу кожної вебсторінки, програма повинна зберігати результати у файлі формату HTML. Записані результати повинні включати інформацію про вебсторінку, час відгуку, наявність або відсутність заданих елементів. Крім того для кожного сайту повинні бути значені коефіцієнти наповнюваності, середнє квадратичне, математичне відхилення тривалості обробки запитів.

2 ВИБІР ПРОГРАМНИХ ЗАСОБІВ

2.1 Вибір мови програмування

Мова програмування C# є однією з основних мов, що використовуються для розробки програм на платформі .NET. Вона пропонує високий рівень абстракції, синтаксис, схожий до мови C++, та багатий набір функціональних можливостей[13]. C# є потужним інструментом для створення розширених програмних додатків і має велику спільноту розробників, що сприяє обміну знаннями та розвитку.

C# також підтримує широкий спектр функціональності, включаючи обробку винятків, делегати, події, лямбда-вирази та багато іншого. Вона надає вбудовану підтримку для маніпулювання рядками, колекціями, файлами, мережами і базами даних, що дозволяє розробникам створювати різноманітні програми з різними функціональними можливостями.

Крім того, C# має розширені можливості щодо безпеки програмного забезпечення, зокрема підтримку криптографічних алгоритмів, автентифікацію і авторизацію.

Ще однією важливою особливістю C# є його мультиплатформеність. За допомогою .NET Core розробники можуть створювати крос-платформні програми, які можуть працювати на різних операційних системах, включаючи Windows, macOS і Linux[14]. Це дозволяє розробникам створювати крос-платформенні рішення, що працюють на різних пристроях і платформах.

C# входить до сім'ї мов програмування .NET і використовується для розробки різноманітних типів додатків, включаючи веб-додатки, настільні програми та мобільні додатки. Він пропонує багатий набір функцій, які спрощують процес розробки, такі як синтаксичний цукор (syntactic sugar),

делегати, події, LINQ (Language-Integrated Query) та інші. Крім того, C# підтримує об'єктно-орієнтовану парадигму програмування, що дозволяє створювати модульні, і легко зрозумілі програми.

2.2 Середовище розробки Visual Studio

Visual Studio - це інтегроване середовище розробки (IDE), яке надає широкий набір інструментів для створення програм на платформі .NET. Воно включає в себе редактор коду з підсвічуванням синтаксису, автодоповненням та інструментами для налагодження програм[15]. Visual Studio також надає доступ до різних шаблонів проектів та бібліотек, що спрощує розробку і забезпечує продуктивність розробника.

Visual Studio також надає можливість створювати графічні інтерфейси для вебдодатків за допомогою Windows Presentation Foundation (WPF). WPF є одним з ключових фреймворків для створення розширених інтерфейсів користувача з використанням XAML (Extensible Application Markup Language) і C#[16]. Він надає розробникам багатофункціональність, можливості стилізації та анімації, що дозволяє створювати динамічні і привабливі вебдодатки.

Visual Studio підтримує розробку ASP.NET вебдодатків з використанням серверної сторони мови C#. За допомогою ASP.NET, розробники можуть створювати потужні вебдодатки, які взаємодіють з базами даних, забезпечують автентифікацію та авторизацію користувачів, обробляють форми та надають різноманітні функціональні можливості для веб-сторінок[17].

Visual Studio разом з мовою C# та платформою .NET надають розробникам потужні інструменти для створення веб-додатків з високою продуктивністю, функціональністю та забезпеченням безпеки. Використання Visual Studio спрощує процес розробки, дозволяючи розробникам швидко створювати і редагувати код, відлагоджувати програми та побудовувати потужні вебдодатки на платформі .NET.

Visual Studio є ідеальним середовищем розробки для платформи .NET, оскільки воно надає повну підтримку для розробки додатків на основі .NET Framework або .NET Core[18]. Ви можете легко створювати .NET-проекти, використовувати бібліотеки класів .NET, налагоджувати ваш код та використовувати інші інструменти, специфічні для .NET.

В цілому, Visual Studio є потужним та розширюваним середовищем розробки, яке надає розробникам широкий набір інструментів і підтримку для різних мов та платформ. Вибір середовища розробки залежить від ваших потреб та вподобань, але Visual Studio є одним з найпопулярніших варіантів на ринку.

2.3 Вибір методу розв’язання задачі

Перш за все, користувач повинен мати можливість встановлювати та редагувати параметри пошуку, які визначають критерії аналізу вебсторінок. Це дозволяє задавати специфічні параметри, за якими програма буде визначати та витягувати необхідну інформацію з веб-документів. Наприклад, користувач може встановити шаблон для вилучення заголовків новин, дат публікацій або інших важливих елементів.

Для вибору елементів, що аналізуються, користувач також повинен мати можливість встановлювати CSS селектори. Це дозволяє точно вказати, які елементи сторінки мають бути проаналізовані. CSS селектори дозволяють вибирати елементи за їх класами, ідентифікаторами, тегами або іншими атрибутами, забезпечуючи гнучкість та точність аналізу.

Для забезпечення ефективного аналізу вебсторінок програма повинна використовувати механізми асинхронного пошуку. Це дозволить здійснювати паралельне завантаження та аналіз багатьох сторінок одночасно, що покращить продуктивність програми та скоротить час виконання.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Розробка інформаційної системи

Розробка інформаційної системи включає в себе процес отримання, зчитування, перевірки та запису даних з вебсайтів.

Програма використовує HTTP-запити для отримання даних з вебсайту. Вона встановлює з'єднання з сервером за допомогою протоколу HTTP або HTTPS. Для цього використовуються бібліотеки, такі як HttpClient у мові програмування C#.

Програма створює запит до сервера, вказуючи URL-адресу сторінки, з якої потрібно отримати дані. Запит може включати параметри, які використовуються для фільтрації або сортування результатів. Після відправки запиту програма очікує на відповідь від сервера.

Після отримання відповіді вебсайту, програма зчитує дані з отриманої відповіді. Для цього використовуються різні методи, залежно від формату даних. Наприклад, якщо дані представлені у форматі HTML, можна використовувати бібліотеки, такі як HtmlAgilityPack для парсингу сторінки та витягування необхідних даних.

Обробка та аналіз статистичних даних є важливою складовою розробки інформаційних систем. Під час цього процесу, статистичні дані, які можуть бути отримані з вебсайтів, піддаються обробці для отримання корисної інформації.

У контексті розробки інформаційної системи, для отримання статистичних даних з вебсайтів використовуються спеціальні програми або скрипти. Зазвичай ці програми використовують HTTP-запити для з'єднання з сервером та отримання даних з вебсторінок.

Отримані дані можуть бути перевірені на правильність та оброблені, щоб отримати необхідну інформацію.

Після обробки та перевірки отриманих даних програма може зберігати ці дані у базу даних або файлову систему. Для збереження даних можуть використовуватися різні технології.

3.2 Опис основних класів та методів додатку

Клас Program є головним класом програми. У ньому знаходиться метод Main, який є точкою входу програми. Клас виконує наступні дії: зчитує дані з файлу JSON, створює новий контекст перегляду, ініціалізує об'єкт аналізатора даних, передає йому зчитані дані та контекст перегляду, і запускає процес аналізу даних. Результати аналізу записуються у файл HTML.

Клас DataReader відповідає за зчитування даних з файлу JSON. Він має метод ReadDataFromFile, який приймає шлях до файлу JSON і повертає об'єкт Data, що містить розібрані дані. Клас використовує бібліотеку Newtonsoft.Json для розбору даних з файлу JSON та створення об'єкта Data.

Клас Data представляє розібрані дані з файлу JSON. Він містить властивості, такі як SearchValue, DataArray та SearchLinks, які відображають значення пошуку, масив даних та посилання для пошуку, що були витягнуті з файлу JSON. Цей клас використовується для передачі даних між DataReader та DataAnalyzer.

Клас DataAnalyzer відповідає за аналіз даних. Він має метод AnalyzeData, який приймає об'єкт Data та контекст перегляду і виконує аналіз даних. Клас використовує контекст перегляду з бібліотеки AngleSharp для відкриття веб-сторінок та пошуку певних елементів на сторінках за допомогою селекторів CSS. Він перевіряє наявність певних значень у вмісті сторінки та перевіряє наявність певних посилань. Результати аналізу зберігаються в об'єкті AnalysisResult.

Клас FileWriter має метод WriteResultsToFile, який приймає результати аналізу та шлях до файлу HTML, і записує результати вказаного аналізу у файл HTML. Для створення та редагування HTML-файлів використовується бібліотека HtmlAgilityPack. Клас генерує HTML-структуру, додає результати аналізу до структури та зберігає у файлі HTML.

Ці класи працюють разом для зчитування даних з файлу JSON, аналізу цих даних, запису результатів аналізу у файл HTML. Кожен клас виконує свою власну функцію та має відповідальність у процесі виконання програми. Взаємодія між класами дозволяє забезпечити ефективний та структурований аналіз даних та збереження результатів.

3.3 Інструкція користувача

Завантажити скомпільований виконуваний файл програми (наприклад, "Program.exe").

Переконатися, що встановлений .NET Framework на комп'ютері. Якщо немає, то завантажити та встановити його з офіційного вебсайту Microsoft.

Створити папку на комп'ютері, в яку розмістити файли програми.

Розпакувати завантажений виконуваний файл програми ("Program.exe") у створену папку.

Створити вхідний файл даних у форматі JSON. Назвати його "data.json" і помістити у папку, де знаходиться виконуваний файл програми. Дотримуватись структури даних, яка вказана у коді програми в розділі "class Data".

Запустити наступну програму: Program.exe

Програма запуститься і почне обробку даних з вхідного файлу. В процесі виконання відображатиметься вивід інформації у командному рядку або терміналі.

Після завершення програма збереже результати аналізу у файл з назвою "index.html" у папці, де знаходиться виконуваний файл програми.

Відкрити файл "index.html" у веббраузері, щоб переглянути результати аналізу.

3.4 Контрольні приклади

Контрольні приклади отримані в результаті роботи програми. В таблиці 3.1 приведені відповіді сайтів на конкретний запит. А також тривалість відповіді. З нього видно, що по частині запитів були відповіді «Позитивна», а по іншим - «Негативна». Ці результати використані для обчислення коефіцієнта наповнюваності для кожного сайту. Ці коефіцієнти приведені на таблиці 3.2.

Таблиця 3.1 Відповіді сайтів на конкретний запит

Saites		Success
https://kvp.sumdu.edu.ua/		4,00%
https://teset.sumdu.edu.ua/uk/		64,00%
Стратегічний план розвитку підрозділу	https://teset.sumdu.edu.ua/uk/	Негативна
Корпоративна культура і академічна доброчесність	https://teset.sumdu.edu.ua/uk/	Позитивна
Інформація про керівника підрозділу, у т.ч. контактна інформація	https://teset.sumdu.edu.ua/uk/	Негативна
Інформація про заступників за напрямами діяльності, у т.ч. контактна інформація	https://teset.sumdu.edu.ua/uk/	Негативна
Інформація про кураторів груп	https://teset.sumdu.edu.ua/uk/	Позитивна
Відгуки випускників, роботодавців, студентів	https://teset.sumdu.edu.ua/uk/	Негативна
Відділ студентських справ	https://teset.sumdu.edu.ua/uk/	Позитивна
Студентське самоврядування інституту/факультету	https://teset.sumdu.edu.ua/uk/	Позитивна
Інформація про Асоціацію випускників інституту/факультету	https://teset.sumdu.edu.ua/uk/	Негативна
Загальна контактна інформація підрозділу (інтерактивні номери телефонів, корпоративні e-mail тощо та лінк на телефонний довідник університету)	https://teset.sumdu.edu.ua/uk/	Позитивна
Повна інформація про ОП	https://teset.sumdu.edu.ua/uk/	Негативна
Повна інформація про викладачів із зазначенням ключових дисциплін, які вони викладають із перелінковкою на відповідні силабуси та персональна контактна інформації (як мінімум, корпоративний e-mail)	https://teset.sumdu.edu.ua/uk/	Негативна
Каталог дисциплін вільного вибору циклу загальної підготовки на поточний навчальний рік	https://teset.sumdu.edu.ua/uk/	Негативна
Каталог та силабуси дисциплін вільного вибору циклу фахової підготовки зі спеціальності (в розрізі спеціальностей) на поточний навчальний рік	https://teset.sumdu.edu.ua/uk/	Негативна
Каталог та силабуси дисциплін вільного вибору циклу фахової підготовки за освітніми програмами (в розрізі програм) на поточний навчальний рік	https://teset.sumdu.edu.ua/uk/	Негативна
Аспірантура та докторантура інституту/факультету	https://teset.sumdu.edu.ua/uk/	Позитивна
On-line (дистанційне) навчання	https://teset.sumdu.edu.ua/uk/	Позитивна
Графіки навчального процесу	https://teset.sumdu.edu.ua/uk/	Позитивна
Рейтинги студентів на отримання стипендії	https://teset.sumdu.edu.ua/uk/	Позитивна
Інформація щодо можливостей та організації проходження практики	https://teset.sumdu.edu.ua/uk/	Позитивна
Інформація про працевлаштування випускників	https://teset.sumdu.edu.ua/uk/	Позитивна
Перелік потенційних роботодавців та наявних стратегічних партнерств з визначенням форм співпраці, даними підтверджувальних документів тощо	https://teset.sumdu.edu.ua/uk/	Негативна
Інформація про міжнародні проєкти	https://teset.sumdu.edu.ua/uk/	Позитивна
Інформація про можливості міжнародної академічної мобільності студентів та аспірантів, включаючи програми подвійних дипломів	https://teset.sumdu.edu.ua/uk/	Негативна
Рейтинги на поселення до гуртожитку	https://teset.sumdu.edu.ua/uk/	Позитивна
Стартова сторінка головного вебсайту СумДУ	//sumdu.edu.ua	Позитивна
Бібліотека	https://library.sumdu.edu.ua/	Позитивна
Розклад занять	http://schedule.sumdu.edu.ua	Позитивна
Реєстр установчих (дозвільних) документів з організації діяльності університету	Посилання не знайдено	Негативна
https://ttf.teset.sumdu.edu.ua/		36,00%

Таблиця 3.2 Коефіцієнти наповнюваності сайтів

Saites	Success
https://kvp.sumdu.edu.ua/	4,00%
https://teset.sumdu.edu.ua/uk/	64,00%
https://ttf.teset.sumdu.edu.ua/	36,00%
https://ecolog.sumdu.edu.ua/uk/	16,00%
https://pgm.sumdu.edu.ua/uk/	8,00%
https://ce.sumdu.edu.ua/uk/	60,0%
https://pmitkm.teset.sumdu.edu.ua/uk/	64,00%
https://tmvi.sumdu.edu.ua/uk/	60,0%
https://chem.teset.sumdu.edu.ua/uk/	72,00%
https://zmdm.teset.sumdu.edu.ua/ua/	60,0%
https://elit.sumdu.edu.ua/	8,00%
https://cs.sumdu.edu.ua/	8,00%
https://pom.sumdu.edu.ua/	12,00%
https://ezpf.elit.sumdu.edu.ua/	16,00%
https://ekt.elit.sumdu.edu.ua/	8,00%
https://nano.elit.sumdu.edu.ua/	0%
https://etech.sumdu.edu.ua/	8,00%
https://maimo.elit.sumdu.edu.ua/	0%
https://cybersecurity.sumdu.edu.ua/	16,00%
https://itp.elit.sumdu.edu.ua/	12,00%
https://ksu.sumdu.edu.ua/	20,0%
https://biem.sumdu.edu.ua/	28,00%
https://econ.biem.sumdu.edu.ua/	16,00%
https://mev.biem.sumdu.edu.ua/	40,0%
https://ek.biem.sumdu.edu.ua/	32,00%
https://management.biem.sumdu.edu.ua/	28,00%
https://fintech.biem.sumdu.edu.ua/	16,00%
https://dom.biem.sumdu.edu.ua/	20,0%
https://op.biem.sumdu.edu.ua/	28,00%
https://ifsk.sumdu.edu.ua/	28,00%
https://kim.sumdu.edu.ua/	32,00%
https://journ.sumdu.edu.ua/	20,0%
https://gf.sumdu.edu.ua/	28,00%
https://ppst.sumdu.edu.ua/	12,00%
https://lang.sumdu.edu.ua/	16,00%
https://law.sumdu.edu.ua/	12,00%
https://agpfeb.law.sumdu.edu.ua/	52,00%

В таблицях 3.3, 3.4, 3.5, 3.6, 3.7 приведені результати статистичного моделювання відповідно для 10, 20, 30, 40, 50 запитів.

Таблиця 3.3 Результат роботи статистичного моделювання для 10 запитів

Номер сайту	URL	Оцінка математичного очікування 10 запитів	Середньо-квадратичне відхилення 10 запитів
1	https://kvp.sumdu.edu.ua/	1826,755	277,2442028
2	https://teset.sumdu.edu.ua/uk/	1193,791	116,2779341
3	https://ttf.teset.sumdu.edu.ua/	514,929	59,15635599
4	https://ecolog.sumdu.edu.ua/uk/	972,799	106,3948829
5	https://pgm.sumdu.edu.ua/uk/	553,147	177,2574821
6	https://ce.sumdu.edu.ua/uk/	301,734	77,46028605
7	https://pmitkm.teset.sumdu.edu.ua/uk/	1024,636	50,29013427
8	https://tmvi.sumdu.edu.ua/uk/	553,403	41,36643649
9	https://chem.teset.sumdu.edu.ua/uk/	1014,138	110,5892316
10	https://zmdm.teset.sumdu.edu.ua/ua/	369,893	15,1655304
11	https://elit.sumdu.edu.ua/	601,455	66,43746053
12	https://cs.sumdu.edu.ua/	662,378	46,11005911
13	https://pom.sumdu.edu.ua/	774,994	79,73300811
14	https://ezpf.elit.sumdu.edu.ua/	1141,388	40,98517667
15	https://ekt.elit.sumdu.edu.ua/	669,571	36,91029259
16	https://nano.elit.sumdu.edu.ua/	610,786	20,07798584
17	https://etech.sumdu.edu.ua/	147,315	32,71314695
18	https://maimo.elit.sumdu.edu.ua/	703,625	136,9092641
19	https://cybersecurity.sumdu.edu.ua/	3669,234	471,6474628
20	https://itp.elit.sumdu.edu.ua/	678,722	60,42819445
21	https://ksu.sumdu.edu.ua/	586,841	59,82492688
22	https://biem.sumdu.edu.ua/	2596,632	516,4777636
23	https://econ.biem.sumdu.edu.ua/	1821,656	111,5620778
24	https://mev.biem.sumdu.edu.ua/	751,478	63,48389206
25	https://ek.biem.sumdu.edu.ua/	4374,17	159,3723937
26	https://management.biem.sumdu.edu.ua/	906,452	39,35183345
27	https://fintech.biem.sumdu.edu.ua/	799,358	25,25631837
28	https://dom.biem.sumdu.edu.ua/	1043,146	27,13437557
29	https://op.biem.sumdu.edu.ua/	3090,209	150,7837723
30	https://ifsk.sumdu.edu.ua/	462,868	241,1879723
31	https://kim.sumdu.edu.ua/	488,147	212,2933903
32	https://journ.sumdu.edu.ua/	458,242	154,1885801
33	https://gf.sumdu.edu.ua/	444,543	103,457858
34	https://ppst.sumdu.edu.ua/	402,146	81,2398556
35	https://lang.sumdu.edu.ua/	428,851	268,130838
36	https://law.sumdu.edu.ua/	3501,966	3491,264641
37	https://agpfeb.law.sumdu.edu.ua/	8267,244	7059,501988
38	https://smp.law.sumdu.edu.ua/	4239,299	3824,860606
39	https://kfcp.law.sumdu.edu.ua/	6502,291	6551,443636
40	https://kmeep.law.sumdu.edu.ua/	2139,803	3014,472222
41	https://history.law.sumdu.edu.ua/	4259,824	2371,357488
42	https://med.sumdu.edu.ua/	548,114	133,0270228
43	https://anatomy.med.sumdu.edu.ua/	1010,167	21,75302383
44	https://pathology.med.sumdu.edu.ua/	916,403	8,534690061
45	https://physiology.med.sumdu.edu.ua/	458,185	64,65740883
46	https://oncology.med.sumdu.edu.ua/	993,688	24,29142546
47	https://biochem.med.sumdu.edu.ua/	321,832	37,48217141
48	https://obsgyn.med.sumdu.edu.ua/	916,506	15,29764848
49	https://infection.med.sumdu.edu.ua/	1229,132	138,6815116
50	https://gensurgery.med.sumdu.edu.ua/	918,696	17,16385168

Таблиця 3.4 Результат роботи статистичного моделювання для 20 запитів

Номер сайту	URL	Оцінка математичного очікування 20 запитів	Середньо-квадратичне відхилення 20 запитів
1	https://kvp.sumdu.edu.ua/	1672,1895	341,9057619
2	https://teset.sumdu.edu.ua/uk/	1148,686	103,6991456
3	https://ttf.teset.sumdu.edu.ua/	479,4205	18,90242381
4	https://ecolog.sumdu.edu.ua/uk/	931,759	77,61283832
5	https://pgm.sumdu.edu.ua/uk/	459,318	64,39170971
6	https://ce.sumdu.edu.ua/uk/	272,4445	68,22847372
7	https://pmitkm.teset.sumdu.edu.ua/uk/	1020,4135	46,33508723
8	https://tmvi.sumdu.edu.ua/uk/	556,1525	57,82658293
9	https://chem.teset.sumdu.edu.ua/uk/	950,73	27,51307192
10	https://zmdm.teset.sumdu.edu.ua/ua/	380,9305	24,02183513
11	https://elit.sumdu.edu.ua/	562,47	93,22070478
12	https://cs.sumdu.edu.ua/	632,215	47,4740599
13	https://pom.sumdu.edu.ua/	772,307	75,36222117
14	https://ezpf.elit.sumdu.edu.ua/	1144,3955	59,7572274
15	https://ekt.elit.sumdu.edu.ua/	673,8455	55,05759391
16	https://nano.elit.sumdu.edu.ua/	599,181	13,71356681
17	https://etech.sumdu.edu.ua/	128,8195	25,06584329
18	https://maimo.elit.sumdu.edu.ua/	658,722	31,39317519
19	https://cybersecurity.sumdu.edu.ua/	3602,527	353,6632246
20	https://itp.elit.sumdu.edu.ua/	673,843	33,41841237
21	https://ksu.sumdu.edu.ua/	544,6985	49,28720099
22	https://biem.sumdu.edu.ua/	2398,278	316,9087121
23	https://econ.biem.sumdu.edu.ua/	1768,8005	54,89392819
24	https://mev.biem.sumdu.edu.ua/	704,302	26,20648184
25	https://ek.biem.sumdu.edu.ua/	4306,733	82,74369787
26	https://management.biem.sumdu.edu.ua/	900,189	21,39773891
27	https://fintech.biem.sumdu.edu.ua/	798,647	31,73071434
28	https://dom.biem.sumdu.edu.ua/	1044,7265	33,47403507
29	https://op.biem.sumdu.edu.ua/	3031,172	104,1066321
30	https://ifsk.sumdu.edu.ua/	392,305	69,49036264
31	https://kim.sumdu.edu.ua/	408,192	67,36774073
32	https://journ.sumdu.edu.ua/	431,4895	130,0486804
33	https://gf.sumdu.edu.ua/	486,152	329,7015217
34	https://ppst.sumdu.edu.ua/	392,379	77,69791157
35	https://lang.sumdu.edu.ua/	428,381	267,1579498
36	https://law.sumdu.edu.ua/	4422,979	4059,132883
37	https://agpfeb.law.sumdu.edu.ua/	7258,392	6515,179986
38	https://smp.law.sumdu.edu.ua/	5451,5825	4405,509392
39	https://kfcp.law.sumdu.edu.ua/	4759,3125	3980,73607
40	https://kmeep.law.sumdu.edu.ua/	1884,524	2010,528033
41	https://history.law.sumdu.edu.ua/	4809,9265	3711,228637
42	https://med.sumdu.edu.ua/	596,149	115,9644691
43	https://anatomy.med.sumdu.edu.ua/	994,666	27,37289024
44	https://pathology.med.sumdu.edu.ua/	920,9795	9,870854213
45	https://physiology.med.sumdu.edu.ua/	424,069	53,29404903
46	https://oncology.med.sumdu.edu.ua/	975,8465	18,65288895
47	https://biochem.med.sumdu.edu.ua/	310,2235	37,45441458
48	https://obsgyn.med.sumdu.edu.ua/	924,6055	25,42127175
49	https://infection.med.sumdu.edu.ua/	1198,897	102,6786302
50	https://gensurgery.med.sumdu.edu.ua/	920,7925	12,67892071

Таблиця 3.5 Результат роботи статистичного моделювання для 30 запитів

Номер сайту	URL	Оцінка математичного очікування 30 запитів	Середньо-квадратичне відхилення 30 запитів
1	https://kvp.sumdu.edu.ua/	1697,992	196,0870003
2	https://teset.sumdu.edu.ua/uk/	1132,608	70,00775644
3	https://ttf.teset.sumdu.edu.ua/	510,3023333	190,1188938
4	https://ecolog.sumdu.edu.ua/uk/	893,3986667	57,3964526
5	https://pgm.sumdu.edu.ua/uk/	427,6966667	60,82849578
6	https://ce.sumdu.edu.ua/uk/	250,8463333	72,58520223
7	https://pmitkm.teset.sumdu.edu.ua/uk/	1022,434667	46,29888408
8	https://tmvi.sumdu.edu.ua/uk/	517,858	60,08573917
9	https://chem.teset.sumdu.edu.ua/uk/	1074,146	694,0259796
10	https://zmdm.teset.sumdu.edu.ua/ua/	373,0483333	11,53249176
11	https://elit.sumdu.edu.ua/	551,4843333	123,2642594
12	https://cs.sumdu.edu.ua/	632,2196667	51,0923017
13	https://pom.sumdu.edu.ua/	723,1406667	58,89457864
14	https://ezpf.elit.sumdu.edu.ua/	1160,880333	86,73462712
15	https://ekt.elit.sumdu.edu.ua/	651,2126667	15,73769775
16	https://nano.elit.sumdu.edu.ua/	600,6673333	15,74501496
17	https://etech.sumdu.edu.ua/	131,3106667	39,74140297
18	https://maimo.elit.sumdu.edu.ua/	668,1846667	34,19314165
19	https://cybersecurity.sumdu.edu.ua/	3611,250333	423,5284558
20	https://itp.elit.sumdu.edu.ua/	673,1603333	30,06635885
21	https://ksu.sumdu.edu.ua/	571,6423333	72,76672196
22	https://biem.sumdu.edu.ua/	2368,252667	307,7953102
23	https://econ.biem.sumdu.edu.ua/	1788,326667	92,92826675
24	https://mev.biem.sumdu.edu.ua/	759,9043333	180,5920916
25	https://ek.biem.sumdu.edu.ua/	4280,399	91,13261153
26	https://management.biem.sumdu.edu.ua/	893,661	18,51110335
27	https://fintech.biem.sumdu.edu.ua/	944,5863333	754,6464475
28	https://dom.biem.sumdu.edu.ua/	1049,129667	40,76502325
29	https://op.biem.sumdu.edu.ua/	3236,181	497,9918293
30	https://ifsk.sumdu.edu.ua/	377,9426667	67,10654441
31	https://kim.sumdu.edu.ua/	441,354	208,3221442
32	https://journ.sumdu.edu.ua/	429,697	218,5394622
33	https://gf.sumdu.edu.ua/	456,0913333	286,2873397
34	https://ppst.sumdu.edu.ua/	384,1446667	117,265998
35	https://lang.sumdu.edu.ua/	348,726	171,0127197
36	https://law.sumdu.edu.ua/	3656,225667	3434,504251
37	https://agpfeb.law.sumdu.edu.ua/	8582,818	7258,077858
38	https://smp.law.sumdu.edu.ua/	5989,645	4598,832424
39	https://kfcp.law.sumdu.edu.ua/	6694,504333	6678,732487
40	https://kmeep.law.sumdu.edu.ua/	2073,379667	2975,475562
41	https://history.law.sumdu.edu.ua/	6072,903667	4147,047977
42	https://med.sumdu.edu.ua/	631,628	836,6146256
43	https://anatomy.med.sumdu.edu.ua/	1023,988333	106,9877084
44	https://pathology.med.sumdu.edu.ua/	933,1603333	44,90687272
45	https://physiology.med.sumdu.edu.ua/	485,874	67,78669637
46	https://oncology.med.sumdu.edu.ua/	995,5393333	68,51677292
47	https://biochem.med.sumdu.edu.ua/	301,1023333	34,24939644
48	https://obsgyn.med.sumdu.edu.ua/	924,7796667	27,24625637
49	https://infection.med.sumdu.edu.ua/	1415,089	965,7048847
50	https://gensurgery.med.sumdu.edu.ua/	929,6923333	20,78481486

Таблиця 3.6 Результат роботи статистичного моделювання для 40 запитів

Номер сайту	URL	Оцінка математичного очікування 40 запитів	Середньо-квадратичне відхилення 40 запитів
1	https://kvp.sumdu.edu.ua/	1694,2275	152,4094535
2	https://teset.sumdu.edu.ua/uk/	1172,66825	97,60440078
3	https://ttf.teset.sumdu.edu.ua/	489,50975	67,64100791
4	https://ecolog.sumdu.edu.ua/uk/	956,27225	114,4125381
5	https://pgm.sumdu.edu.ua/uk/	447,99825	64,70244892
6	https://ce.sumdu.edu.ua/uk/	260,11325	60,0437237
7	https://pmitkm.teset.sumdu.edu.ua/uk/	1045,3195	83,94780057
8	https://tmvi.sumdu.edu.ua/uk/	538,42475	61,73030553
9	https://chem.teset.sumdu.edu.ua/uk/	972,91275	45,15848661
10	https://zmdm.teset.sumdu.edu.ua/ua/	369,67025	8,990715578
11	https://elit.sumdu.edu.ua/	543,11375	64,62234613
12	https://cs.sumdu.edu.ua/	640,688	45,31929657
13	https://pom.sumdu.edu.ua/	749,51625	69,56029521
14	https://ezpf.elit.sumdu.edu.ua/	1172,749	128,2872014
15	https://ekt.elit.sumdu.edu.ua/	667,99875	42,25021482
16	https://nano.elit.sumdu.edu.ua/	608,15375	16,01876155
17	https://etech.sumdu.edu.ua/	129,9235	25,13749886
18	https://maimo.elit.sumdu.edu.ua/	663,875	23,60047088
19	https://cybersecurity.sumdu.edu.ua/	4204,32575	442,1465194
20	https://itp.elit.sumdu.edu.ua/	694,3975	53,56379009
21	https://ksu.sumdu.edu.ua/	567,23975	62,2531472
22	https://biem.sumdu.edu.ua/	2699,30475	333,8501099
23	https://econ.biem.sumdu.edu.ua/	1932,44775	172,4518692
24	https://mev.biem.sumdu.edu.ua/	745,35375	53,26291242
25	https://ek.biem.sumdu.edu.ua/	4362,483	99,17264501
26	https://management.biem.sumdu.edu.ua/	902,125	27,09799094
27	https://fintech.biem.sumdu.edu.ua/	876,3535	462,8501286
28	https://dom.biem.sumdu.edu.ua/	1049,2665	22,38770685
29	https://op.biem.sumdu.edu.ua/	3119,64875	213,7565519
30	https://ifsk.sumdu.edu.ua/	397,1905	162,4245212
31	https://kim.sumdu.edu.ua/	468,565	562,5790486
32	https://journ.sumdu.edu.ua/	396,696	111,1587973
33	https://gf.sumdu.edu.ua/	416,10925	242,7588273
34	https://ppst.sumdu.edu.ua/	366,33175	96,83827798
35	https://lang.sumdu.edu.ua/	349,59875	166,0522754
36	https://law.sumdu.edu.ua/	4163,10175	3946,228801
37	https://agpfeb.law.sumdu.edu.ua/	8644,8795	8000,106985
38	https://smp.law.sumdu.edu.ua/	6098,29075	5216,415551
39	https://kfcp.law.sumdu.edu.ua/	6207,96	7147,157445
40	https://kmeep.law.sumdu.edu.ua/	2047,34675	2117,583913
41	https://history.law.sumdu.edu.ua/	5495,10875	4445,140128
42	https://med.sumdu.edu.ua/	474,34675	98,72191839
43	https://anatomy.med.sumdu.edu.ua/	1004,732	26,1459064
44	https://pathology.med.sumdu.edu.ua/	930,06225	27,77576138
45	https://physiology.med.sumdu.edu.ua/	475,30625	72,31172202
46	https://oncology.med.sumdu.edu.ua/	992,7225	25,39584528
47	https://biochem.med.sumdu.edu.ua/	310,3455	46,85708183
48	https://obsgyn.med.sumdu.edu.ua/	919,557	12,62712164
49	https://infection.med.sumdu.edu.ua/	1299,07925	850,0532612
50	https://gensurgery.med.sumdu.edu.ua/	983,73125	370,2932906

Таблиця 3.7 Результат роботи статистичного моделювання для 50 запитів

Номер сайту	URL	Оцінка математичного очікування 50 запитів	Середньо-квадратичне відхилення 50 запитів
1	https://kvp.sumdu.edu.ua/	1654,8188	157,5459747
2	https://teset.sumdu.edu.ua/uk/	1116,4932	56,65296406
3	https://ttf.teset.sumdu.edu.ua/	474,851	13,35438707
4	https://ecolog.sumdu.edu.ua/uk/	916,8342	78,69233933
5	https://pgm.sumdu.edu.ua/uk/	427,997	71,77138055
6	https://ce.sumdu.edu.ua/uk/	239,9324	53,388589
7	https://pmitkm.teset.sumdu.edu.ua/uk/	1023,3406	53,6947874
8	https://tmvi.sumdu.edu.ua/uk/	514,1486	77,34891717
9	https://chem.teset.sumdu.edu.ua/uk/	961,2922	36,06375807
10	https://zmdm.teset.sumdu.edu.ua/ua/	435,1704	310,844842
11	https://elit.sumdu.edu.ua/	515,0982	53,92347513
12	https://cs.sumdu.edu.ua/	625,9998	43,32459685
13	https://pom.sumdu.edu.ua/	736,811	70,71516603
14	https://ezpf.elit.sumdu.edu.ua/	1157,7452	35,81698384
15	https://ekt.elit.sumdu.edu.ua/	668,9272	59,80018975
16	https://nano.elit.sumdu.edu.ua/	615,85	21,5834294
17	https://etech.sumdu.edu.ua/	126,9706	22,39970885
18	https://maimo.elit.sumdu.edu.ua/	657,8982	21,02970627
19	https://cybersecurity.sumdu.edu.ua/	3795,7684	364,4875968
20	https://itp.elit.sumdu.edu.ua/	672,0824	19,2548629
21	https://ksu.sumdu.edu.ua/	559,1788	52,81037501
22	https://biem.sumdu.edu.ua/	2343,3722	256,4134894
23	https://econ.biem.sumdu.edu.ua/	1922,6094	420,4228309
24	https://mev.biem.sumdu.edu.ua/	780,974	169,2298197
25	https://ek.biem.sumdu.edu.ua/	4313,5458	173,2777956
26	https://management.biem.sumdu.edu.ua/	966,0412	443,3512828
27	https://fintech.biem.sumdu.edu.ua/	860,2602	343,0986154
28	https://dom.biem.sumdu.edu.ua/	1047,4242	24,2346069
29	https://op.biem.sumdu.edu.ua/	3069,474	100,6037483
30	https://ifsk.sumdu.edu.ua/	386,4934	110,2453519
31	https://kim.sumdu.edu.ua/	393,315	107,3169708
32	https://journ.sumdu.edu.ua/	401,4196	111,7788943
33	https://gf.sumdu.edu.ua/	438,4842	394,0874377
34	https://ppst.sumdu.edu.ua/	360,6568	68,32839654
35	https://lang.sumdu.edu.ua/	342,42	154,3204881
36	https://law.sumdu.edu.ua/	4523,1154	4034,497704
37	https://agpfeb.law.sumdu.edu.ua/	8773,0612	8303,059483
38	https://smp.law.sumdu.edu.ua/	5577,0024	4361,429146
39	https://kfcp.law.sumdu.edu.ua/	6000,92	6644,972378
40	https://kmeep.law.sumdu.edu.ua/	2165,82	2537,039567
41	https://history.law.sumdu.edu.ua/	5920,6568	4100,58514
42	https://med.sumdu.edu.ua/	502,4524	191,8310763
43	https://anatomy.med.sumdu.edu.ua/	1008,7608	57,29529602
44	https://pathology.med.sumdu.edu.ua/	941,676	59,60065357
45	https://physiology.med.sumdu.edu.ua/	464,8964	73,25363079
46	https://oncology.med.sumdu.edu.ua/	1017,5124	94,93680984
47	https://biochem.med.sumdu.edu.ua/	322,5718	43,92058072
48	https://obsgyn.med.sumdu.edu.ua/	918,5906	11,50701798
49	https://infection.med.sumdu.edu.ua/	1252,2456	134,8760886
50	https://gensurgery.med.sumdu.edu.ua/	927,151	13,38098777

Як видно з наведених результатів в таблицях 3.3 – 3.7 із збільшенням кількості запитів оцінки математичних очікувань починають змінюватися в межах, які задовольняють практичні потреби. В той же час реальна робота сайтів не дає можливості збільшити кількість запитів для моделювання. Тому можна вважати, що результати, приведені в таблиці 3.7 для усіх 50 сайтів, відображають кінцеві результати. Їх можна використовувати для аналізу роботи системи сайтів Сумського державного університету.

ВИСНОВКИ

Розробленений алгоритм та комп’ютерна програма для отримання статистичних характеристик роботи вебсайтів структурних підрозділів Сумського державного університету. Роботу програми перевірено в реальних умовах. Розроблена програма має практичне застосування. Вона надає користувачам зручні інструменти для налаштування параметрів інформаційної системи. Завдяки використанню асинхронного програмування, програма забезпечує ефективне та швидке виконання завдань аналізу.

Результати аналізу вебсторінок, що надаються програмою, можуть бути використані для вдосконалення якості та продуктивності вебсайтів. Користувачі зможуть отримати інформацію про час відгуку сторінок, наявність або відсутність заданих елементів та відсоток успішності аналізу. Це допоможе виявити можливі проблеми з завантаженням сторінок та покращити їх продуктивність.

Розроблена програма використовує передові технології і стандарти веброзробки, такі як HTML, CSS, C# та платформа .NET. Вона підтримує асинхронну обробку даних, що дозволяє ефективно виконувати аналіз вебсторінок і забезпечувати високу продуктивність програми.

Використання CSS селекторів дозволяє точно визначати елементи на веб-сторінках для подальшого аналізу. Це надає гнучкість і можливість налаштування програми під конкретні вимоги та потреби користувача.

Результати аналізу вебсторінок, які зберігаються у файлі HTML, можуть бути легко використані для подальшої обробки і аналізу. Інформація про час відгуку сторінок, наявність елементів і відсоток успішності аналізу дозволяють отримати цінні відомості про якість та продуктивність вебсторінок.

СПИСОК ЛІТЕРАТУРИ

1. Mozilla Developer Network - HTML [Електронний ресурс] // developer.mozilla.org. Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/HTML.

2. W3Schools - HTML Tutorial [Електронний ресурс] // www.w3schools.com. Режим доступу до ресурсу: https://www.w3schools.com/html/

3. Mozilla Developer Network - CSS [Електронний ресурс] // developer.mozilla.org. Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/CSS

4. W3Schools - CSS Tutorial [Електронний ресурс] // www.w3schools.com. Режим доступу до ресурсу: https://www.w3schools.com/css/

5. Mozilla Developer Network - JavaScript [Електронний ресурс] // developer.mozilla.org. Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/JavaScript

6. Wikipedia - Client-server model [Електронний ресурс] // en.wikipedia.org. Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Client%E2%80%93server_model

7. Codecademy - Client-Server Communication [Електронний ресурс] // www.codecademy.com. Режим доступу до ресурсу: https://www.codecademy.com/articles/what-is-cli-server-communication

8. MDN Web Docs - HTTP Methods [Електронний ресурс] // developer.mozilla.org. Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

9. GlobalSign - What is SSL? [Електронний ресурс] // www.globalsign.com. Режим доступу до ресурсу: https://www.globalsign.com/en/ssl-information-center/what-is-ssl

10. Statistics Solutions - What is Statistical Modeling? [Електронний ресурс] // www.statisticssolutions.com. Режим доступу до ресурсу: https://www.statisticssolutions.com/statistical-modeling/

11. Towards Data Science - Statistical Modeling [Електронний ресурс] // towardsdatascience.com. Режим доступу до ресурсу: https://towardsdatascience.com/tagged/statistical-modeling

12. Investopedia - Standard Deviation [Електронний ресурс] // www.investopedia.com. Режим доступу до ресурсу: https://www.investopedia.com/terms/s/standarddeviation.asp

13. Microsoft - C# Programming Guide [Електронний ресурс] // docs.microsoft.com. Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/csharp/

14. Microsoft - .NET [Електронний ресурс] // dotnet.microsoft.com. Режим доступу до ресурсу: https://dotnet.microsoft.com/

15. Microsoft - Visual Studio [Електронний ресурс] // visualstudio.microsoft.com. Режим доступу до ресурсу: https://visualstudio.microsoft.com/

16. Microsoft - WPF Documentation [Електронний ресурс] // docs.microsoft.com. Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/desktop-wpf/

17. Microsoft - ASP.NET [Електронний ресурс] // dotnet.microsoft.com. Режим доступу до ресурсу: https://dotnet.microsoft.com/apps/aspnet

18. Microsoft - .NET Framework [Електронний ресурс] // dotnet.microsoft.com. Режим доступу до ресурсу: https://dotnet.microsoft.com/download/dotnet-framework

ДОДАТОК А

```
using AngleSharp;
using HtmlAgilityPack;
using Newtonsoft.Json.Linq;
using System.Diagnostics;

class Program
{
    static async Task Main(string[] args)
    {
        // Read data from file
        var data = await DataReader.ReadDataAsync("data.json");

        // Create a new browsing context
        var context = BrowsingContext.New(Configuration.Default.WithDefaultLoader());

        // Initialize the analyzer and analyze the data
        var analyzer = new DataAnalyzer(context);
        var results = await analyzer.AnalyzeDataAsync(data);

        // Write results to file
        var fileWriter = new FileWriter();
        fileWriter.WriteResultsToFile(results, "index.html");
    }
}

class DataReader
{
    public static async Task<Data> ReadDataAsync(string fileName)
    {
        var json = await File.ReadAllTextAsync(fileName);
        var jsonObj = JObject.Parse(json);

        var searchValues = jsonObj["SearchValues"].ToObject<JArray>();
        var dataArray = jsonObj["data"].ToObject<JArray>();
        var searchLinks = jsonObj["SearchLinks"].ToObject<JArray>();

        return new Data(searchValues, dataArray, searchLinks);
    }
}

class Data
{
    public JArray SearchValues { get; }
    public JArray DataArray { get; }
    public JArray SearchLinks { get; }

    public Data(JArray searchValues, JArray dataArray, JArray searchLinks)
    {
        SearchValues = searchValues;
        DataArray = dataArray;
        SearchLinks = searchLinks;
    }
}


class DataAnalyzer
```

```
{
    private readonly IBrowsingContext _context;

    public DataAnalyzer(IBrowsingContext context)
    {
        _context = context;
    }

    public async Task<List<Result>> AnalyzeDataAsync(Data data)
    {
        var results = new List<Result>();

        foreach (var item in data.DataArray)
        {
            var url = item["Url"].ToString();
            var cssSelectors = item["CssSelectors"].ToString();
            var faculty = item["Faculty"].ToArray();

            // Navigate to search results page
            var stopwatch = new Stopwatch();
            stopwatch.Start();
            var document = await _context.OpenAsync(url);
            stopwatch.Stop();

            // Get through cssSelectors
            var elements = document.QuerySelectorAll(cssSelectors);

            // Get all links
            var links = document.QuerySelectorAll("a")
                .Select(link => link.GetAttribute("href"))
                .Where(href => !string.IsNullOrEmpty(href))
                .Distinct();

            var httpClient = new HttpClient();

            try
            {
                // Send a GET request to the URL
                HttpResponseMessage httpResponse = await httpClient.GetAsync(url);

                var titletable = new Result
                {
                    IsParent = true,
                    LinkUrl = url.ToString(),
                    Response = stopwatch.ElapsedMilliseconds,
                    LinkText = httpResponse.StatusCode.ToString()
                };
                // Add result to list of results
                results.Add(titletable);
                // Check if the status code indicates success

                if (httpResponse.IsSuccessStatusCode)
                {
                    var responseStatusCode = httpResponse.StatusCode;

                    Console.WriteLine($"Response status code: {responseStatusCode}");

                    decimal success = 0;
                    decimal successfulCount = 0;
```

```
decimal failedCount = 0;

//Search for values by text
foreach (var searchValue in data.SearchValues)
{
    var name = searchValue["name"].ToString();

    var values = searchValue["values"];

    bool countElements = false;

    foreach (var value in values)
    {
        failedCount++;

        foreach (var element in elements)
        {


            if (element.TextContent.Contains(value.ToString()))
            {
                var result = new Result
                {
                    IsParent = false,
                    SearchValue = name.ToString(),
                    LinkText = name.ToString(),
                    LinkUrl = url.ToString(),
                    TrueOrFalse = true,
                    Response = stopwatch.ElapsedMilliseconds
                };
                // Add result to list of results
                results.Add(result);

                successfulCount++;
                countElements = true;
                break;
            }
            else
            {
                var result = new Result
                {
                    IsParent = false,
                    SearchValue = name.ToString(),
                    LinkText = name.ToString(),
                    LinkUrl = url.ToString(),
                    TrueOrFalse = false,
                    Response = stopwatch.ElapsedMilliseconds,


                };
                // Add result to list of results
                results.Add(result);

                break;
            }
        }
        if (!countElements)
        {
            break;
```

```
                        }
                        break;
                    }
                }

                //Search for values by reference
                foreach (var facultyItem in faculty)
                {
                    var itemName = facultyItem.ToString();

                    foreach (var searchLink in data.SearchLinks)
                    {
                        try
                        {
                            var searchLinkItems = searchLink[itemName].ToArray();
                            foreach (var sitem in searchLinkItems)
                            {
                                foreach (var searchLinkItem in searchLinkItems)
                                {
                                    var searchLinkItemName =
searchLinkItem["name"];
                                    var searchLinkItemValues =
searchLinkItem["values"];

                                    bool countElements = false;

                                    foreach (var searchLinkItemValue in
searchLinkItemValues)
                                    {
                                        if (!countElements)
                                        {
                                            string value =
searchLinkItemValue.ToString();

                                            foreach (var link in links)
                                            {

                                                if (link.Equals(value))
                                                {

                                                    countElements = true;

                                                    var result = new Result
                                                    {
                                                        IsParent = false,
                                                        SearchValue =
searchLinkItemName.ToString(),

                                                        LinkUrl = link.ToString(),
                                                        TrueOrFalse = true,
                                                        Response =
stopwatch.ElapsedMilliseconds

                                                    };
                                                    // Add result to list of
results

                                                    results.Add(result);
                                                    successfulCount++;
                                                    break;

                                                }
```

```
                                                else
                                                {
                                                }
                                            }
                                        }

                                    }
                                    if (!countElements)
                                    {
                                        var result = new Result
                                        {
                                            IsParent = false,
                                            SearchValue =
searchLinkItemName.ToString(),
                                            LinkUrl = "Посилання не знайдено",
                                            TrueOrFalse = false,
                                            Response =
stopwatch.ElapsedMilliseconds
                                        };
                                        // Add result to list of results
                                        results.Add(result);
                                        continue;
                                    }
                                }
                                break;
                            }
                        }
                        catch (Exception)
                        {

                            continue;
                        }

                    }
                }

                //Count the success of a value lookup
                if (successfulCount != 0 && failedCount != 0)
                {
                    success += (successfulCount / failedCount) * 100;
                    titletable.Success = Math.Round(success, 2);
                }
                else
                {
                    success = 0;
                    titletable.Success = Math.Round(success, 2);
                }
            }
            else
            {
                // The site is not working
                Console.WriteLine("Site is not working. Status code: {0}",
httpResponse.StatusCode);
            }

        }
        catch (HttpRequestException ex)
        {
            // Handle the exception if the connection attempt fails
```

```
                Console.WriteLine("Site is not working. Exception message: {0}",
ex.Message);
                var result = new Result
                {
                    IsParent = true,
                    Response = 0,
                    LinkText = "Перевірте посилання",
                    LinkUrl = url.ToString()
                };
                // Add result to list of results
                results.Add(result);
            }
        }
        return results;


    }
}

class FileWriter
{
    public void WriteResultsToFile(IEnumerable<Result> results, string fileName)
    {
        // Create a new HTML document
        var document = new HtmlDocument();

        // Create the document head
        var head = document.DocumentNode.AppendChild(document.CreateElement("head"));
        head.AppendChild(document.CreateElement("title")).InnerHtml = "Data Analysis
Results";
        var rerol = head.AppendChild(document.CreateElement("link"));
        rerol.SetAttributeValue("rel", "stylesheet");
        rerol.SetAttributeValue("href", "D:\\CS\\MonitorSys\\MonitorSys\\style.css");

        // Create the document body
        var body = document.DocumentNode.AppendChild(document.CreateElement("body"));
        body.AppendChild(document.CreateElement("h1")).InnerHtml = "Data Analysis
Results";

        // Create a table to display the results
        var table = body.AppendChild(document.CreateElement("table"));
        var thead = table.AppendChild(document.CreateElement("thead"));
        var headers = thead.AppendChild(document.CreateElement("tr"));

        // Create the table headers

        /*var headers = table.AppendChild(document.CreateElement("tr"));*/
        headers.AppendChild(document.CreateElement("th")).InnerHtml = "Saites";
        headers.AppendChild(document.CreateElement("th")).InnerHtml = "Time";
        headers.AppendChild(document.CreateElement("th")).InnerHtml = "Response";
        headers.AppendChild(document.CreateElement("th")).InnerHtml = "Success";
        //headers.AppendChild(document.CreateElement("th")).InnerHtml = "Response
received";


        // Create the table rows for each result
        foreach (var result in results)
        {
            if (result.IsParent)
```

```
            {
                var row = table.AppendChild(document.CreateElement("tr"));
                row.AddClass("parent");
                row.AppendChild(document.CreateElement("td")).InnerHtml =
result.LinkUrl;
                row.AppendChild(document.CreateElement("td")).InnerHtml =
$"{result.Response} ms.";
                row.AppendChild(document.CreateElement("td")).InnerHtml =
$"{result.LinkText}";
                row.AppendChild(document.CreateElement("td")).InnerHtml =
$"{result.Success}%";
            }
            else
            {
                if (result.TrueOrFalse)
                {
                    var row = table.AppendChild(document.CreateElement("tr"));
                    row.AddClass("child true");

                    row.AppendChild(document.CreateElement("td")).InnerHtml =
result.SearchValue;

                    var linkTd = row.AppendChild(document.CreateElement("td"));
                    var link = linkTd.AppendChild(document.CreateElement("a"));
                    link.SetAttributeValue("href", $"{result.LinkUrl}");
                    link.InnerHtml = result.LinkUrl;

                    row.AppendChild(document.CreateElement("td")).InnerHtml =
"Позитивна";
                }
                if (!result.TrueOrFalse)
                {
                    var row = table.AppendChild(document.CreateElement("tr"));
                    row.AddClass("child false");

                    row.AppendChild(document.CreateElement("td")).InnerHtml =
result.SearchValue;

                    var linkTd = row.AppendChild(document.CreateElement("td"));
                    var link = linkTd.AppendChild(document.CreateElement("a"));
                    link.SetAttributeValue("href", $"{result.LinkUrl}");
                    link.InnerHtml = result.LinkUrl;

                    row.AppendChild(document.CreateElement("td")).InnerHtml =
"Негативна";
                }
                else
                {

                }
            }
        }
        var jquery = document.CreateElement("script");
        jquery.SetAttributeValue("src", "https://code.jquery.com/jquery-3.6.0.min.js");
        body.AppendChild(jquery);

        var script = document.CreateElement("script");
        script.SetAttributeValue("src", "D:\\CS\\MonitorSys\\MonitorSys\\script.js");
        body.AppendChild(script);
```

```
        // Save the document to a file
        document.Save(fileName);
    }
}

class Result
{
    public string SearchValue { get; set; }
    public string LinkUrl { get; set; }
    public bool TrueOrFalse { set; get; }
    public long Response { get; set; }
    public bool IsParent { get; set; }
    public string LinkText { get; set; }
    public decimal Success { get; set; }
}
```

ДОДАТОК Б

```
          using System.Diagnostics;
using Newtonsoft.Json;
using HtmlAgilityPack;

namespace SpeedTest
{
    class Program
    {
        static void Main(string[] args)
        {
            string jsonFilePath = "test.json";

            Console.WriteLine("Введіть кількість операцій:");
            int operationCount = int.Parse(Console.ReadLine());
            Console.WriteLine("Введіть шлях до файлу HTML для збереження
результатів:");
            string nubbOfOper = Console.ReadLine();


            List<string> urls = LoadUrlsFromJson(jsonFilePath);
            List<SiteMeasurement> measurements = MeasurePageLoadTimes(urls,
operationCount);

            string htmlFilePath = $"D:\\{nubbOfOper}.html";

            SaveMeasurementsToHtml(measurements, htmlFilePath);

            Console.WriteLine("Результати було збережено в HTML-файл.");
            Console.WriteLine("Натисніть будь-яку клавішу, щоб вийти...");
            Console.ReadKey();
        }

        static List<string> LoadUrlsFromJson(string jsonFilePath)
        {
            List<string> urls = new List<string>();

            try
            {
                string json = File.ReadAllText(jsonFilePath);
                dynamic data = JsonConvert.DeserializeObject(json);

                foreach (var url in data.urls)
                {
                    urls.Add(url.ToString());
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Помилка при завантаженні JSON-файлу:
{ex.Message}");
            }

            return urls;
        }
```

```
        static List<SiteMeasurement> MeasurePageLoadTimes(List<string> urls, int
operationCount)
        {

            int totalSites = urls.Count;
            List<SiteMeasurement> measurements = new List<SiteMeasurement>();

            for (int siteIndex = 0; siteIndex < urls.Count; siteIndex++)
            {
                string url = urls[siteIndex];
                int siteNumber = siteIndex + 1;

                Console.WriteLine($"Розігрування номеру сайту (звернення до сайту
#{siteNumber}/{totalSites}): {url}");

                List<double> loadTimes = new List<double>();

                for (int i = 0; i < operationCount; i++)
                {
                    int requestNumber = i + 1;

                    Stopwatch stopwatch = Stopwatch.StartNew();
                    var loadTime = MeasurePageLoadTime(url);
                    stopwatch.Stop();

                    double roundedLoadTime = Math.Round(loadTime.TotalMilliseconds, 2);
                    Console.WriteLine($"Час виконання запиту
{requestNumber}/{operationCount}: {roundedLoadTime} мс");

                    loadTimes.Add(roundedLoadTime);

                }

                double averageLoadTime = loadTimes.Average();
                Console.WriteLine($"Математичного очікування для сайту
#{siteNumber}/{totalSites}: {averageLoadTime} мс");

                double standardDeviation = CalculateStandardDeviation(loadTimes);
                Console.WriteLine($"Середньо-квадратичне відхилення для сайту
#{siteNumber}/{totalSites}: {standardDeviation}");

                SiteMeasurement measurement = new SiteMeasurement
                {
                    SiteNumber = siteNumber,
                    Url = url,
                    AverageLoadTime = averageLoadTime,
                    StandardDeviation = standardDeviation
                };
                measurements.Add(measurement);

                Console.WriteLine();
            }

            return measurements;
        }

        static double CalculateStandardDeviation(List<double> values)
        {
            double average = values.Average();
```

```
            double sumOfSquaredDifferences = values.Sum(x => Math.Pow(x - average, 2));
            double variance = sumOfSquaredDifferences / (values.Count - 1);
            double standardDeviation = Math.Sqrt(variance);
            return standardDeviation;
        }

        static TimeSpan MeasurePageLoadTime(string url) //Звернення до сайту string url
        {
            using (HttpClient client = new HttpClient())
            {
                try
                {
                    Stopwatch stopwatch = Stopwatch.StartNew();
                    var response = client.GetAsync(url).Result;
                    stopwatch.Stop();
                    return stopwatch.Elapsed;
                }
                catch (Exception ex)
                {
                    Console.WriteLine($"Помилка при вимірюванні швидкості завантаження:
{ex.Message}");
                    return TimeSpan.Zero;
                }
            }
        }
        static void SaveMeasurementsToHtml(List<SiteMeasurement> measurements, string
htmlFilePath)
        {
            // Створення HTML-структури з даними вимірювань
            HtmlDocument document = new HtmlDocument();
            HtmlNode html = document.CreateElement("html");
            HtmlNode head = document.CreateElement("head");
            HtmlNode body = document.CreateElement("body");

            // Додавання заголовка до HTML-файлу
            HtmlNode title = document.CreateElement("title");
            title.InnerHtml = "Результати вимірювання швидкості завантаження";
            head.AppendChild(title);

            // Створення таблиці для відображення даних
            HtmlNode table = document.CreateElement("table");
            table.AddClass("measurements-table");

            // Створення заголовків таблиці
            HtmlNode thead = document.CreateElement("thead");
            HtmlNode trHead = document.CreateElement("tr");

            HtmlNode thSiteNumber = document.CreateElement("th");
            thSiteNumber.InnerHtml = "Номер сайту";

            HtmlNode thUrl = document.CreateElement("th");
            thUrl.InnerHtml = "URL";

            HtmlNode thAverageLoadTime = document.CreateElement("th");
            thAverageLoadTime.InnerHtml = "Середній час завантаження (мс)";

            HtmlNode thStandardDeviation = document.CreateElement("th");
            thStandardDeviation.InnerHtml = "Середньо-квадратичне відхилення";
```

```
            trHead.AppendChild(thSiteNumber);
            trHead.AppendChild(thUrl);
            trHead.AppendChild(thAverageLoadTime);
            trHead.AppendChild(thStandardDeviation);

            thead.AppendChild(trHead);
            table.AppendChild(thead);

            // Додавання даних про кожен сайт
            HtmlNode tbody = document.CreateElement("tbody");

            foreach (var measurement in measurements)
            {
                HtmlNode trData = document.CreateElement("tr");

                HtmlNode tdSiteNumber = document.CreateElement("td");
                tdSiteNumber.InnerHtml = measurement.SiteNumber.ToString();

                HtmlNode tdUrl = document.CreateElement("td");
                tdUrl.InnerHtml = measurement.Url;

                HtmlNode tdAverageLoadTime = document.CreateElement("td");
                tdAverageLoadTime.InnerHtml = measurement.AverageLoadTime.ToString();

                HtmlNode tdStandardDeviation = document.CreateElement("td");
                tdStandardDeviation.InnerHtml =
measurement.StandardDeviation.ToString();

                trData.AppendChild(tdSiteNumber);
                trData.AppendChild(tdUrl);
                trData.AppendChild(tdAverageLoadTime);
                trData.AppendChild(tdStandardDeviation);

                tbody.AppendChild(trData);
            }

            table.AppendChild(tbody);
            body.AppendChild(table);

            html.AppendChild(head);
            html.AppendChild(body);
            document.DocumentNode.AppendChild(html);

            // Збереження HTML-файлу
            document.Save(htmlFilePath);
        }
    }

    class SiteMeasurement
    {
        public int SiteNumber { get; set; }
        public string Url { get; set; }
        public double AverageLoadTime { get; set; }
        public double StandardDeviation { get; set; }
    }
}
```