**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**
**Сумський державний університет**
Факультет електроніки та інформаційних технологій
Кафедра комп'ютених наук

«До захисту допущено»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ
(підпис)

_____19 травня 2023 р.____

# КВАЛІФІКАЦІЙНА РОБОТА
## на здобуття освітнього ступеня магістр

зі спеціальності 122 - Комп'ютених наук,

освітньо-наукової програми «Інформатика»

на тему: «Інформаційна технологія динамічної пріоритезації вхідного трафіку в умовах нестачі обчислювальних ресурсів»

здобувача групи ІН.м-11н Наталухи Максима Романовича

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

_____ Максим НАТАЛУХА
(підпис)

Керівник, старший викладач, к.т.н. Борис КУЗІКОВ _____
(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 магістр**

зі спеціальності 122 - Комп'ютерних наук, освітньо-наукової програми «Інформатика»
здобувача групи ІН.м-11н Наталухи Максима Романовича

1. Тема роботи: «Інформаційна технологія динамічної пріоритезації вхідного трафіку в умовах нестачі обчислювальних ресурсів»
затверджую наказом по СумДУ від «08» травня 2023 р. № 0475-VI
2. Термін здачі здобувачем кваліфікаційної роботи до 19 травня 2023 року
3. Вхідні дані до кваліфікаційної роботи завдання на розробку інформаційної технології динамічної пріоритезації вхідного трафіку в умовах нестачі обчислювальних ресурсів
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
*Аналіз предметної області, постановка задачі та методи дослідження, проектування додатку, практична реалізація*
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 _____ (підпис) Керівник _____ (підпис)

# КАЛЕНДАРНИЙ ПЛАН

| № п/п | Назва етапів кваліфікаційної роботи | Термін виконання | Примітка |
|---|---|---|---|
| *1* | *Аналіз проблеми розпізнавання шкідливого програмного забезпечення* | | |
| *2* | *Формалізована постановка задачі дослідження* | | |
| *3* | *Огляд існуючих рішень* | | |
| *4* | *Опис інформаційної технології* | | |
| *5* | *Програмна реалізація інформаційної технології* | | |
| *6* | *Тестування і аналіз отриманих результатів* | | |
| *7* | *Оформлення кваліфікаційної магістерської роботи* | | |

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

# АНОТАЦІЯ

**Записка:** 78 стор., 12 рис., 10 табл., 11 додатків, 29 джерел.

**Об'єкт дослідження** — процес забезпечення стабільного доступу до ресурсу під навантаженням.

**Мета роботи** — розробка інформаційної технології динамічної пріоритезації вхідного трафіку та її імплементація.

**Методи дослідження** — інформаційний аналіз, метод моделювання.

**Результати** — розроблено інформаційну технологію динамічної пріоритезації трафіку та імплементовано у як конфігурацію модуля для застосунку NGINX який використовує накопичену бази знань про користування ресурсом, наданої допустимої моделі взаємодії користувачів, метрик які відображають навантаження на ресурс. І як результат застосування фільтрів до аномального трафіку, який вишиковується в чергу обробки сервісом під час перевантаженості системи, таким чином щоб аномальні запити були переміщені в кінець черги або відкинуті.

СИСТЕМА АНАЛІЗУ ТРАФІКУ, ПРІОРИТЕЗАЦІЇ ТРАФІКУ, ІНФОРМАЦІЙНА БЕЗПЕКА, NGINX, ЗАБЕЗПЕЧЕННЯ НЕФУНКЦІОНАЛЬНИХ ВИМОГ, МОНІТОРИНГОВА СИСТЕМА.

# ЗМІСТ

# ВСТУП

На сучасному ринку інформаційних технологій та веб-розробки велика увага приділяється покращенню ефективності та надійності ресурсів. Конкуренція серед компаній, які пропонують послуги у цій галузі, зростає, тому важливо розробляти та впроваджувати інноваційні рішення, які дозволяють підвищувати якість обслуговування та задоволеність користувачів.

Так, зважаючи на те, що сучасний ринок характеризується динамічним розвитком технологій, компанії повинні постійно вдосконалювати свої продукти та послуги. Загалом, інноваційні розробки, які дозволяють покращувати якість та надійність ресурсів, знаходячи ринки збуту своїх послуг, що є ключовими факторами успіху на сучасному ринку інформаційних технологій та веб-розробки. Тому у зв'язку з ростом кількості користувачів та обсягу трафіку виникає необхідність у вдосконаленні системи моніторингу та фільтрації трафіку.

У світі, де зростає кількість кібератак та шахрайських дій, важливість ефективного та точного фільтрування аномального трафіку стає все більш високою. Бо деякі користувачі зловживають наданими ресурсами і як результат можуть погіршити досвід користування інших користувачів. Це може стати проблемою для бізнесів, які мають великий обсяг трафіку, але не мають достатньої кількості ресурсів для обробки цього трафіку.

Інформаційна технологія, яка використовує накопичену базу знань, може якісно вплинути на дотримання нефункціональних вимог та продуктивності ресурсу для доброчесних користувачів. А також дозволяють не лише покращувати якість роботи ресурсу, але й знижувати витрати на його обслуговування.

Розробка інформаційної технології динамічної пріоритезації трафіку є важливим для поліпшення ефективності та надійності ресурсу, і буде застосовувати фільтри у випадку перевантаження системи, коли і без такої ресурс і так не може задовольнити потреби всіх користувачів.

Розроблення такого підходу дозволяє збирати та аналізувати дані про користування ресурсом, створювати моделі взаємодії користувачів з ресурсом та використовувати їх для покращення ефективності обробки запитів та знаходити нові паттерни нецільового використання ресурсу і будувати на основі них нові фільтри. А своєчасне застосування фільтрів до аномального трафіку дозволяє уникнути перевантаження системи та забезпечити стабільну роботу ресурсу, що і є обґрунтуванням актуальності такого підходу.

Для досягнення зазначеної мети можна виділити наступні задачі:

– Аналіз існуючих методів пріоритезації трафіку і визначення їх переваг та недоліків.

– Розробка алгоритму динамічної пріоритезації трафіку, який забезпечує ефективний розподіл ресурсів надаючи перевагу доброчесним користувачам.

– Проведення експериментальних досліджень з метою оцінки ефективності розробленого алгоритму.

– Розробка програмного забезпечення для реалізації розробленого алгоритму в практичних застосуваннях.

Науковою новизною такої розробки є підхід до розпізнавання та фільтрації аномального трафіку на основі накопиченої бази знань про користувачів та їх взаємодію з ресурсом за допомогою відповідних метрик і використання такої інформації для пріоритезації трафіку. Такий підхід дозволяє забезпечити більш точну та ефективну фільтрацію аномалій, за рахунок можливості поєднання з існуючими методами, що, наприклад, використовують статистичні моделі та математичні алгоритми. Така розробка використовує практичний підхід до розпізнавання та фільтрації аномального трафіку на основі реальних даних та побудованої на їх основі моделі взаємодії користувачів з ресурсом. Цей підхід дозволяє досягнути більш високої точності та швидкості фільтрування аномального трафіку. Ця розробка може виявитись незамінною для бізнесу та організацій, які прагнуть забезпечити безпеку своїх користувачів та позитивно вплинути на доступність своїх ресурсів від незапланованого використання.

# 1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ

В сучасному світі веб-додатки стають все більш популярними, що в свою чергу зростає ризик їх атаки. З цієї причини безпека веб-додатків стала однією з ключових проблем для бізнесу та організацій. У зв'язку з цим, на ринку з'явилося багато різноманітних засобів, що дозволяють забезпечити високий рівень захисту веб-додатків від різних видів атак.

Одним з таких засобів є WAF (Web Application Firewall), що є рішенням для захисту веб-додатків від різноманітних атак, таких як SQL ін'єкції, кросс-сайт скриптінг (XSS), вразливості, пов'язані з налагодженням та інші. Ці засоби можуть захистити веб-додатки від атак, які використовуються для злому веб-сайтів, втрати даних користувачів та інших проблем безпеки.

У цьому контексті розглядаємо різні WAF-засоби, такі як Cloudflare WAF[1], ModSecurity[2], F5 Networks WAF[3], Imperva Web Application Firewall[4] та Akamai Kona Site Defender[5]. Кожен з цих засобів має свої переваги та недоліки, тому перед вибором певного засобу потрібно оцінити їхню функціональність, вартість, підтримку та інші параметри, які можуть вплинути на ефективність захисту веб-додатків.

## 1.1 Cloudflare WAF

Cloudflare WAF - це безкоштовний веб-застосунок з мережі доставки вмісту (CDN), призначений для захисту веб-додатків від широкого спектру атак, таких як SQL-ін'єкції, XSS-атаки, а також від інших видів атак, спрямованих на веб-додатки.

Cloudflare WAF може працювати як самостійний захист веб-додатків, або разом з іншими засобами безпеки Cloudflare, такими як DDoS-захист та захист від ботів.

Cloudflare WAF аналізує трафік, що надходить на веб-додатки, та виявляє та блокує атаки на основі правил безпеки, які можна налаштувати. Правила безпеки можуть бути настроєні для конкретного веб-додатку або для групи веб-додатків, що дозволяє забезпечити високий рівень захисту.

Cloudflare WAF має кілька переваг:

– Легкість в настройці та використанні.

– Наявність великої кількості правил безпеки, що забезпечує захист від багатьох типів атак.

– Здатність відслідковувати та блокувати атаки на ранніх стадіях.

– Можливість налаштовувати правила безпеки відповідно до потреб веб-додатка.

– Підтримка різних протоколів та форматів даних, таких як HTTP, HTTPS та інші.

– Швидка реакція на нові загрози та виправлення їх відповідними правилами.

Незважаючи на переваги, Cloudflare WAF також має деякі недоліки, такі як можливість хибних спрацювань правил захисту, які можуть призвести до відмов в обслуговуванні легітимних запитів, а також обмеження у настройці правил безпеки.

### 1.2 ModSecurity

ModSecurity - це безкоштовний веб-застосунок (WAF) з відкритим вихідним кодом, призначений для захисту веб-додатків від широкого спектру атак, таких як SQL-ін’єкції, XSS-атаки, а також від інших видів атак, що спрямовані на веб-додатки. Він може працювати як самостійний модуль, а також як модуль для веб-серверів, таких як Apache, Nginx та IIS. Він аналізує трафік між веб-сервером та клієнтом, виявляє атаки та відхиляє їх, якщо вони не відповідають правилам безпеки.

ModSecurity має можливості налаштування правил, що дозволяє налаштувати його на конкретні потреби веб-додатка. Також ModSecurity підтримує мову OWASP ModSecurity Core Rule Set (CRS), яка забезпечує засоби захисту від широкого спектру атак та розробляється спільнотою.

Основні переваги ModSecurity:

– Відкритий вихідний код та безкоштовність.

– Можливості налаштування та підтримка різних мов програмування.

– Підтримка різних протоколів та форматів даних, таких як HTTP, HTTPS, XML та інші.

– Наявність правил безпеки, що забезпечує захист від багатьох типів атак.

– Можливість аналізувати та блокувати атаки на рівні трафіку, що дозволяє виявляти та блокувати атаки на ранніх стадіях.

Незважаючи на переваги, ModSecurity має деякі недоліки, такі як складність в настройці та можливість ложних спрацювань правил захисту, які можуть призвести до відмов в обслуговуванні легітимних запитів.

### 1.3 F5 Networks WAF

F5 Networks WAF - це засіб захисту веб-додатків від різних видів атак, таких як SQL-ін'єкції, XSS-атаки та інші. Це продукт компанії F5 Networks, який дозволяє забезпечити високий рівень захисту веб-додатків та даних користувачів.

Основні функції F5 Networks WAF включають:

– Аналіз трафіку: засіб аналізує вхідний та вихідний трафік, що проходить через веб-додаток, для виявлення атак.

– Правила безпеки: F5 Networks WAF містить багато правил безпеки, що дозволяє блокувати різні види атак.

– Настройка: засіб можна настроїти відповідно до потреб веб-додатка та бізнесу.

– Моніторинг та звітність: засіб забезпечує можливість моніторингу веб-додатку та надає звіти про потенційні загрози.

– Підтримка протоколів: F5 Networks WAF підтримує різні протоколи, такі як HTTP, HTTPS, FTP та інші.

– Швидкість та надійність: засіб працює з високою швидкістю та надійністю.

Окрім перелічених переваг, F5 Networks WAF має деякі недоліки, такі як складність в настройці та використанні, високі вимоги до апаратного

забезпечення та вартість ліцензії, що може бути значною для невеликих компаній.

В цілому, F5 Networks WAF - це потужний та надійний засіб захисту веб-додатків, який дозволяє забезпечити високий рівень безпеки для веб-додатків та даних користувачів.

### 1.4 Imperva Web Application Firewall (WAF)

Imperva Web Application Firewall (WAF) - це засіб захисту веб-додатків від різних видів атак, таких як SQL-ін'єкції, XSS-атаки, DDoS-атаки та багато інших. Це продукт компанії Imperva, який дозволяє забезпечити високий рівень захисту веб-додатків та даних користувачів.

Основні функції Imperva WAF включають:

– Аналіз трафіку: засіб аналізує вхідний та вихідний трафік, що проходить через веб-додаток, для виявлення атак.

– Правила безпеки: Imperva WAF містить багато правил безпеки, що дозволяє блокувати різні види атак.

– Налаштування: засіб можна настроїти відповідно до потреб веб-додатка та бізнесу.

– Моніторинг та звітність: засіб забезпечує можливість моніторингу веб-додатку та надає звіти про потенційні загрози.

– Підтримка протоколів: Imperva WAF підтримує різні протоколи, такі як HTTP, HTTPS, FTP та інші.

– Швидкість та надійність: засіб працює з високою швидкістю та надійністю.

– Advanced Bot Protection: Imperva WAF включає інструменти захисту від ботів, що дозволяють виявляти та блокувати різні види ботів та шкідливих скриптів.

– Machine Learning: Imperva WAF використовує машинне навчання для постійного вдосконалення аналізу трафіку та виявлення нових видів атак.

Окрім перелічених переваг, Imperva WAF має деякі недоліки, такі як висока вартість ліцензії та складність в настройці та використанні для не досвідчених користувачів.

### 1.5 Akamai Kona Site Defender

Akamai Kona Site Defender є засобом захисту веб-додатків від різних видів атак, що базується на хмарній інфраструктурі. Це продукт компанії Akamai, що дозволяє забезпечити високий рівень захисту веб-додатків та даних користувачів.

Основні функції Akamai Kona Site Defender включають:

Аналіз трафіку: засіб аналізує вхідний та вихідний трафік, що проходить через веб-додаток, для виявлення атак.

Правила безпеки: Akamai Kona Site Defender містить багато правил безпеки, що дозволяє блокувати різні види атак.

Настройка: засіб можна настроїти відповідно до потреб веб-додатка та бізнесу.

Моніторинг та звітність: засіб забезпечує можливість моніторингу веб-додатку та надає звіти про потенційні загрози.

Підтримка протоколів: Akamai Kona Site Defender підтримує різні протоколи, такі як HTTP, HTTPS, FTP та інші.

Швидкість та надійність: засіб працює з високою швидкістю та надійністю.

Advanced Bot Protection: Akamai Kona Site Defender включає інструменти захисту від ботів, що дозволяють виявляти та блокувати різні види ботів та шкідливих скриптів.

Використання хмарної інфраструктури: Akamai Kona Site Defender базується на хмарній інфраструктурі, що дозволяє забезпечувати високий рівень захисту веб-додатків та масштабування в залежності від потреб.

### 1.6 Недоліки існуючих рішень

Недоліками існуючих рішень захисту є те, що вони можуть лише заборонити доступ або змушувати користувача вводити капчу. Ці рішення не завжди актуальні, оскільки вони є надмірно радикально і в деяких випадках це є надлишковим. Що може призвести до негативного враження про веб-ресурс та його розробників.

Заборона доступу може бути неактуальною в тому випадку, якщо шахрії використовують IP-адреси, які належать користувачам, або якщо вони користуються проксі-серверами. Це може призвести до того, що валідація на основі IP-адрес може призвести до блокування легітимних користувачів.

Змушування користувача вводити капчу також може мати негативні наслідки. Це може призвести до зниження якості користувацького досвіду від взаємодії з веб-ресурсом і збільшення часу, необхідного для завершення бізнес-процесів, що підштовхує користувача знайти інший ресурс де подібні незручності є меншими.

В умовах коли є достатньо обчислювальних ресурсів і запит не має шкідливого вмісту можна запропонувати наступний механізм обробки – сервер оброблює подібні запити до тих пір, поки лишаються вільні обчислювальні ресурси, а у разі їх вичерпання подібні запити з аномальною активність складається в чергу, і перевага віддається запитам де немає подібної активності. І коли обчислювальні ресурси звільняться можна знову продовжити оброблювати штатному режимі. Цей підхід може бути актуальним, наприклад коли користувач використовує публічний проксі сервер і навіть у випадку коли станеться пік навантаження – такому користувачу доведеться трохи більше почекати, але це не буде мати менші репутаційні втрати, це може бути спричинено станом самого проксі серверу.

# 2 ФОРМАЛІЗАЦІЯ ЗАДАЧІ. ВИБІР ЗАСОБІВ ВИРІШЕННЯ

## 2.1 Побудова архітектури інформаційної технології пріоритезації

Архітектура складається з таких компонентів:

– Реверсивний проксі сервер

– Додаток контролю трафіку

– Веб-сервер

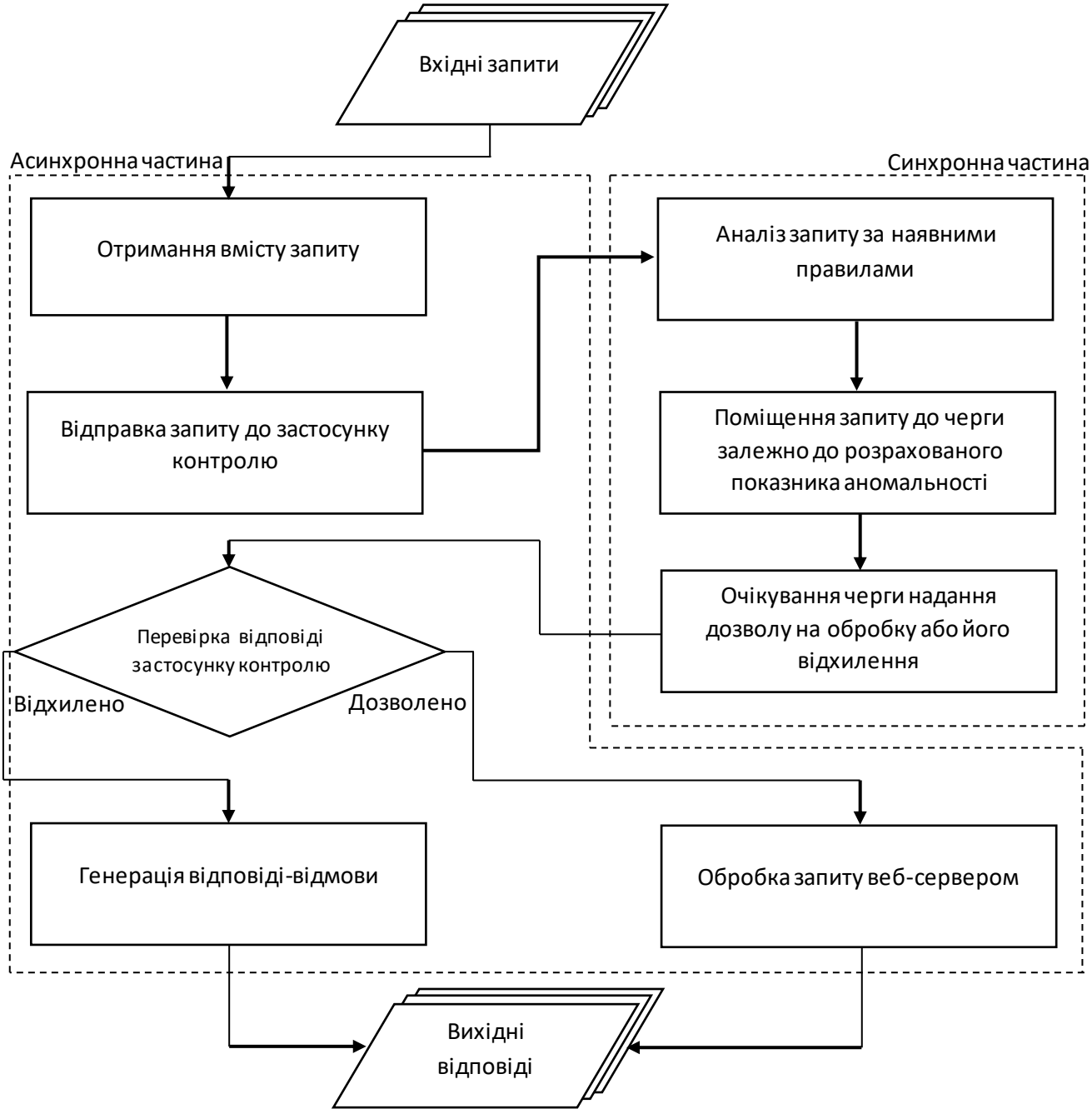


Рисунок 2.1 – Блок-схема архітектури додатку

### 2.1.1 Роль реверсивного проксі серверу

Реверсивний проксі сервер приймає всі запити і надсилає запит спершу на додаток контролю трафіку для аналізу, чекає відповіді і після погодження зі сторони додатку контролю трафіку, запит надсилається на обробку веб-сервером, або інакше повертає повідомлення про помилку. Також реверсивний проксі пише логи доступу, що містить інформацію про оброблені запити і час їх виконання.

### 2.1.2 Роль додатку контролю

У додатку контролю, відбувається застосування фільтрів, які дають оцінку на основі даних отриманих з логів, що пише реверсивний проксі стосовно запитів, що вже були оброблені раніше. І на основі цієї оцінки кожен запит додається до черги отримуючи перевагу над запитами, що отримали більший показник аномальності. Ця логіка застосовується у випадку надмірного навантаження і у разі відсутності такого всі запити проходять без додаткових перевірок. Такий підхід дозволяє уникнути впливу на роботу серверу в нормальному режимі. Також роль цього додатку полягає у здійснюванні моніторингу стану обчислювальних ресурсів серверу, наприклад, на предмет зростання часу відповіді. Виявлення погіршення показників призводить до відключення прозорого пропускання запитів та формування черги на основі наявних правил. Як зазначалося раніше однією з ключових метрик, що використовується в даному додатку, є час обробки запиту. Якщо цей показник продовжує зростати, тоді, вікно одночасно оброблюваних запитів звужується для запитів, які можуть бути оброблені одночасно. Такий підхід дозволяє оптимізувати використання обчислювальних ресурсів серверу та підвищити продуктивність додатку і не впливати на продуктивність при типовій завантаженості.

### 2.1.3 Роль веб-серверу

Веб-сервер може бути будь-яким, доступ до якого можемо забезпечити за допомогою реверсивного проксі.

### 2.2 Вибір реверсивного проксі

Реверсивний проксі може допомогти в ряді випадків:

– Захист веб-сайту від атаки: реверсивний проксі може фільтрувати вхідний трафік, захищати веб-сервер від атак DDoS, SQL-ін'єкцій та інших видів кібератак.

– Збільшення продуктивності: реверсивний проксі може розподіляти трафік між декількома веб-серверами, зменшуючи навантаження на кожен з них і підвищуючи продуктивність.

– Забезпечення доступності: реверсивний проксі може бути налаштований для автоматичної переадресації трафіку на репліку у разі відмови основного сервера. Це допомагає забезпечити доступність веб-сайту у випадку відмови окремого сервера.

– Керуванням трафіком: реверсивний проксі може здійснювати керуванням трафіком, зокрема здійснювати балансування навантаження та керуванням політикою кешування, що допомагає підвищити продуктивність та знизити навантаження на сервери.

– Захистом конфіденційної інформації: реверсивний проксі може забезпечити захист конфіденційної інформації шляхом шифрування трафіку за допомогою SSL або TLS.

Одним з найпопулярніших реверсивних проксі є Nginx, який має високу продуктивність та можливості конфігурації. Його використання дозволяє, за допомогою директиви auth_request, контролювати доступ до ресурсів на основі результатів підзапитів. Це можливо завдяки використанню механізму авторизації, що забезпечує безпеку ресурсів. З рештою цей механізм використовує асинхронні запити і тому дозволяє тримати підключення і не переадресувати запит безпосередньо на веб-сервер.

Тому даний застосунок за рахунок своїх особливостей дозволяє легко взяти контроль за запитами, що будуть надходити нам на наш веб-сервер.

### 2.3 Вимоги та задачі додатку контролю трафіку

Додаток контролю це синхронний керуючий модуль, який отримує всі запити з реверсивного проксі та здійснює моніторинг за допомогою наявних метрик, які вказують на стан сервера та додатку, що обслуговуються. Наприклад, можуть бути вимірювані такі показники, як час відповіді сервера, кількість запитів, навантаження на процесор, диск і т.д.

Якщо на основі моніторингу виявляється погіршення стану сервера, додаток контролю має прийняти рішення про зміну режиму роботи з прозорого на режим пріоритезації. Наприклад, якщо середній час відповіді сервера став більше, ніж певний поріг, то можна встановити обмеження на кількість оброблюваних запитів за певний часовий проміжок. Це дозволяє зменшити навантаження на сервер і підвищити його продуктивність за рахунок відтермінування запитів що були відмічені як аномальні.

Додаток контролю також може використовувати правила, які можуть бути доповнені та розширені плагінами, це основний спосіб кастомізації цього рішення. Це допомагає визначити, які запити мають бути оброблені в першу чергу. Наприклад, можна встановити правило, що запити з певним токеном мають бути оброблені першими. Він може бути наданий адміністраторам ресурсу. Це може бути необхідно для швидкого адміністрування коли обчислювальні ресурси вичерпані. Або навпроти ми можемо відкласти в кінець черги, ті запити, що відправлені від користувачів, які надсилають аномальну кількість запитів, зменшуючи їх пріоритет. Ці правила використовуються для побудови черги обробки запитів, що будуть переслані на веб-сервер.

У разі, якщо черга стає занадто довгою, додаток контролю може почати відкидати запити з найнижчим пріоритетом (найбільшим показником аномальності). Таким чином, веб-сервер залишається стабільно доступним і може продовжувати обробляти запити навіть при великому навантаженні.

# 3 ПРОГРАМНА РЕАЛІЗАЦІЯ

## 3.1 Підготовка середовища

Для реалізації застосунку було обрано сервер під керуванням операційної системи Debian 11. Який має один гігабайт оперативної пам'яті та одне ядро процесора.

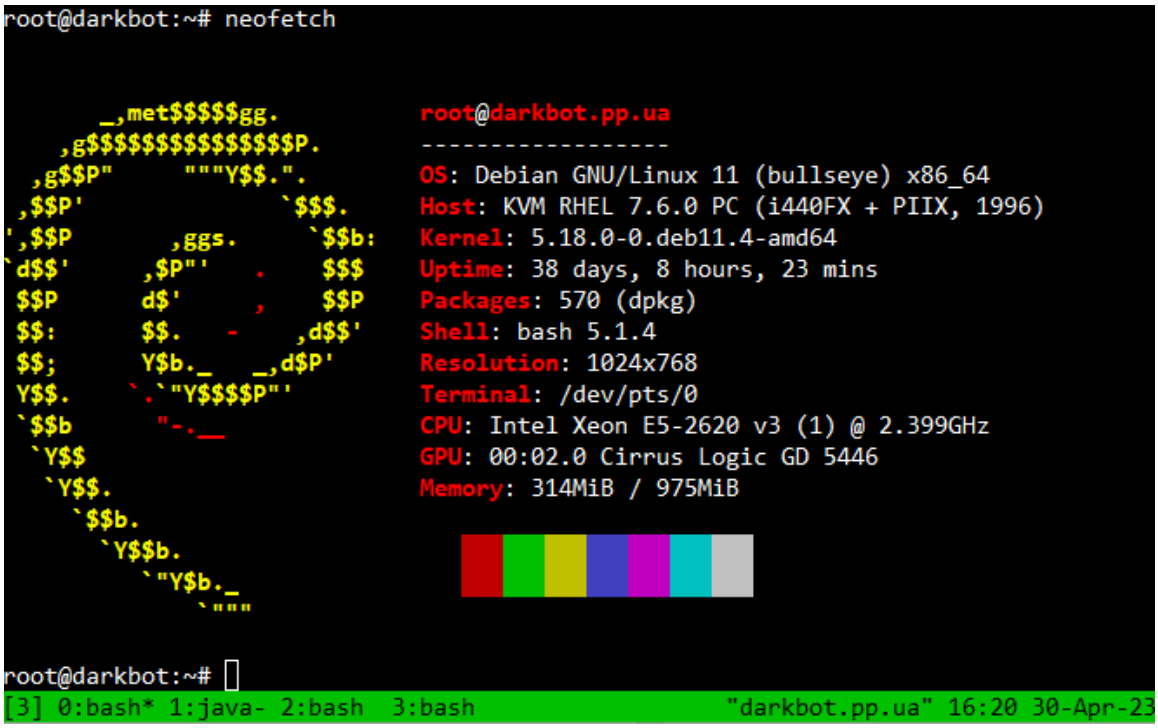


Рисунок 3.1 – Характеристики системи, відображені за допомогою утиліти neofetch

Використання незначної кількості ресурсів обумовлено подальшою необхідністю їх завантаження до моменту коли вони будуть вичерпані.

## 3.2 Налаштування реверсивного проксі серверу

Для встановлення NGINX необхідно виконати команду

```
sudo apt install nginx
```

Після завершення можна перейти до конфігураційного файлу, що знаходиться зв шляхом `/etc/nginx/nginx.conf`

В секції `http` за допомогою директиви

```
log_format mylog '{"ip":"$remote_addr","duration":$request_time,
"time":"$time_local","action":"$request","code":$status,
"size":$body_bytes_sent}';
```

вказуємо формат запису логування, ці данні будуть використані для аналізу хронології запитів та стану серверу додатком контролю трафіку.

Також в секції `http` треба додати секцію `server`, яка містить директиву `listen 80;`, яка вказує на те, що сервіс буде доступний по 80-тому поту.

Також потрібно вказати файл куди будуть писатися логи за допомогою директиви `access_log /var/log/nginx/access.log mylog;`.

Дочірня секція `location /` необхідна для надання доступу до статичного контенту розміщеного за шляхом `/www/WEB-APP;` і вона має наступний вигляд:

```
location / {
   root /www/WEB-APP;
   autoindex on;
}
```

Наступна секція конфігурується залежно від використовуваного веб-серверу, куди будуть надсилатися запити за реверсивного проксі серверу і вона має наступний вигляд:

```
location /api/ {
    auth_request /queue;
    rewrite ^/api/?(.*)$ /$1 break;
    include conf/proxyset.conf;
    proxy_pass http://127.0.0.1:8080;
}
```

Останні дві секції тісно пов’язані між собою

```
location /queue {
    internal;
    proxy_pass http://127.0.0.1:8090;
    include conf/proxyset.conf;
    error_page 502 =200 @success;
}

location @success {
    return 200 'Success';
}
```

Основною їх задачею є відправка запиту до застосунку контролю трафіку що знаходиться за адресою `http://127.0.0.1:8090;` з наданням усієї необхідної інформації за допомогою директиви `proxy_set_header`.

Але у разі недоступності додатку контролю трафіку, наприклад по причині оновлення, директива `error_page 502 =200 @success;` дозволить коректно працювати сервісу переписуючи відповідь як успішну, для цього і потрібна остання секція `location @success`.

Також для зручності був використано тунелювання трафіку через сервери Cloudflare і для коректної обробки IP адреси було додано директиву `real_ip_header CF-Connecting-IP;`.

Фінальний і повний текст файлу конфігурації представлений у додатку А.

Для застосування змін внесених до конфігураційного файлу треба виконати команду

```
nginx -s reload
```

### 3.3 Розробка застосунку контролю трафіку

Додаток контролю трафіку виступає менеджером трафіку для ресурсів що мають директиву `auth_request /queue;`. Це дозволяє гнучко налаштовувати ті компоненти, на які буде вплив з боку додатку контролю трафіку. Як зазначалося в попередньому розділі застосунок доступний за адресою `127.0.0.1:8090`, на яку приймає інформацію від реверсивного проксі сервера у вигляді хедерів вхідного запиту і повертає одну з можливих відповідей зазначених у таблиці 3.1. Саме ними відбувається керування потоком запитів шляхом відтермінування дозволу для запитів, що мали більший показник аномальності що був розрахований за допомогою правил. Для додавання правил була розроблена система плагінів, дозволяє розробити власне рішення для потреб кожного проекту.

Таблиця 3.1 – можливі відповіді від додатку контролю трафіку

| Відповідь для надання дозволу на обробку | Відповідь при переповненні черги для останнього елемента |
|---|---|
| `HTTP/1.1 204 No Content\r\n Content-Length: 0\r\n Connection: Closed\r\n\r\n` | `HTTP/1.1 403 Forbidden\r\n Content-Length: 0\r\n Connection: Closed\r\n\r\n` |

Для того щоб додати плагін треба створити клас, що буде проанотовано за допомогою «@Plugin», ця анотація вказує на те що, даний клас є плагіном нашого додатку. Для того щоб додати правило необхідно імплементувати наступний інтерфейс:

```
public interface RulePlugin {
    int applyRule(Map<String, String> headers);
}
```

Цей інтерфейс містить єдиний метод, який приймає на вхід список у вигляді ключ-значення тих хедерів, що містив в собі запит. У відповідь правило має повернути ціле значення, яке буде використано для формування черги і визначає пріоритет – чим менше це значення, тим вищий пріоритет. Приклад плагіну показано у додатку В.

Для керування застосовуваними правилами використовується файл «config.json». Він містить в собі конфігурацію компонентів додатку і має наступний вигляд:

```
{
    "accessLog":{
        "FilePath":"/var/log/nginx/access.log",
        "HistoryTime":"172800",
        "HistoryRecords":"500000"
    },
    "plugins":[
         "org.plugins.DemoPlugin"
     ],
    "PluginValues":{
         "org.plugins.DemoPlugin":{
             "limit requests per second": "2"
         }
    }
}
```

Даний файл має формат JSON і в основному об’єкті конфігурації є характеристика «plugins» що містить в собі список класів які були проанотовані за допомогою анотації «@Plugin» і мають бути використані для розрахунку впливу на систему. Це дозволяє створити один Jar файл, що може містити декілька класів правил, які можуть бути увімкнені шляхом додавання у конфігураційний файл. Плагіни можуть використовувати вбудований клас Config для отримання значень конфігурації додатку.

Ще одною компонентною додатку є моніторинг оброблених запитів, ця логіка описана в класі LogHandler, який зчитує оновлення логів описаних директивою access_log у конфігурації застосунку реверсивного проксі серверу. Робота з файлом та його вмістом організована наступним чином:

```
ProcessBuilder pb = new ProcessBuilder("tail -n 0 -f ", file);
try(InputStream is = pb.start().getInputStream();
  Scanner s = new Scanner(is).useDelimiter("\\n")){
  while(true) {
    if (s.hasNext()) {
      logs.addFirst(gson.fromJson(s.next(), LogRecord.class));
      if (historyRecords > 0 && logs.size() > historyRecords)
        logs.removeLast();
    }
  }
}
```

Даний клас конфігурується за допомогою характеристики «accessLog», що має містити наступні поля:

– «FilePath» - шлях до файлу, який був зказаний в конфігуації реверсивного проксі серверу за допомогою директиви `access_log /var/log/nginx/access.log mylog;`.

– «HistoryTime» - час в секундах записи яких будуть зберігатися в пам’яті і можуть бути доступними для аналізу.

– «HistoryRecords» - кількість записів, що будуть зберігатися в пам’яті і можуть бути доступними для аналізу.

Для аналізу поточного запиту може бути корисним метод:

```
List<LogRecord> getRecordsByIP(String ip)
```

Який повертає список записів, що містять відомості зчитані з логу для конкретної IP адреси. Клас «LogRecord» має наступний опис:

```
public static class LogRecord {
    public final int code, size;
    public final String ip, time, action;
    public final float duration;
    Date getDate();
}
```

Цей клас містить метод для отримання дати у звичному вигляді для Java.

Програмний код додатку контролю трафіку представлений у додатку Б, а конфігурація для збирання проекту за допомогою Maven представлена у додатку В. Запуск додатку виконується шляхом додавання всіх необхідних компонентів до classpath, такі як сам додаток, бібліотека роботи з json та необхідні плагіни. В якості точки входу треба використати клас Init.

Приклад команди для запуску додатку контролю трафіку:

```
java -cp controller-1.0-SNAPSHOT.jar:gson-2.10.1.jar Init
```

### 3.4 Розробка веб-серверного застосунку

Для цієї задачі використано веб-сервер KTOR[8] для REST побудови простого застосунку який імплементує розширену мета модель за допомогою бібліотеки Exposed для бази даних SQLite, такий додаток працює не надто швидко після заповнення бази даних випадковими сутностями. І тому досягти ліміту обробки запитів буде не так складно.

Веб-сервер доступний на порту 8080 і дозволяє керувати сутностями такі як атрибути, списковізначення, об’єктні типи та об’єкти.

Для взаємодії зазначеними сутностями необхідно використовувати наступні шляхи:

- /attr – для взаємодії з атрибутами
- /list – для взаємодії з списовими значеннями
- /type – для взаємодії з об’єктними типами
- /obj – для взаємодії з об’єктами

Таблиця 3.2 – Методи взаємодії з сутностями

| Тип attr | Тип list |
|---|---|
| – GET<br>– GET ("/{attr}")<br>– GET ("/{attr}/values")<br>– POST<br>– PUT<br>– DELETE ("/{attr}") | – GET<br>– GET ("/{list}")<br>– POST<br>– PUT ("/{list}")<br>– DELETE ("/{list}") |
| **Тип type** | **Тип obj** |
| – GET<br>– GET ("/{type}")<br>– GET ("/{type}/attrs")<br>– POST<br>– PUT<br>– PUT ("/{type}/bind/{attr}")<br>– PUT ("/{type}/unbind/{attr}")<br>– DELETE ("/{type}") | – GET<br>– GET ("/{obj}")<br>– GET ("/{obj}/params")<br>– GET ("/{obj}/param/{paramId}")<br>– POST("/{obj}/param/{paramId}")<br>– POST<br>– PUT<br>– DELETE ("/{obj}") |

В таблиці 3.2 вказані всі можливі методи та набори. Загалом метод GET без аргументів повертає сутності того типу, до якого він був застосований. Метод GET якому був переданий ідентифікатор повертає опис конкретної сутності. Метод POST використовується для оновлення уже існуючих сутностей приймаючи на вхід у JSON форматі новий стан сутності. Метод PUT використовується для створення нових сутностей, він працює аналогічно методу POST за виключенням лише того, що не треба вказувати ідентифікатор сутності. Новий ідентифікатор разом з іншими характеристиками буде повернуто у відповідь. Метод DELETE використовується для видалення сутності.

Вихідні коди додатку веб серверу представлені у додатку Д. Опис всіх залежностей та компонентів для збірки проекту, за допомогою Gradle версії 7.3 або сучаснішої, знаходиться у додатку Е.

Для полегшення взаємодії створено проект для середовища Postman, що дозволяє у віконному режимі виконувати взаємодію з REST додатком, а також заповнити базу різноманітними даними. Повний список всіх запитів, для кожної сутності, зображено на рисунку 3.2, а повний текст вмісту цього проекту розміщено у додатку Л.

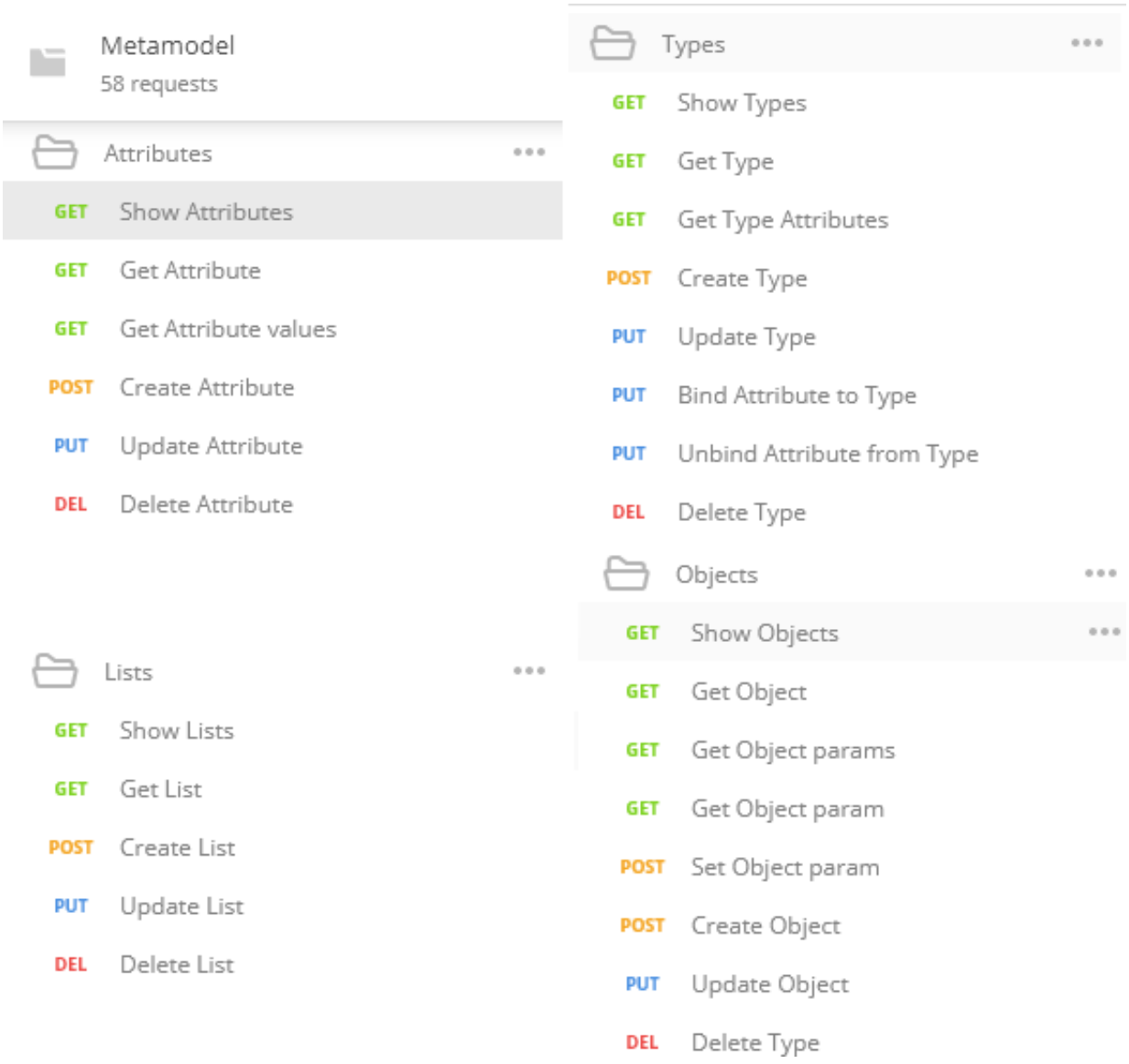


Рисунок 3.2 – Організація взаємодії за допомогою Postman

# 4 ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

## 4.1 Підготовка середовища тестування

Для тестування під навантаженням розроблено скрипт який доступний по статичному шляху «/js/metaMpdel.js». Цей скрипт імплементує методи взаємодії з мета моделлю веб-сервера у вигляді JavaScript об’єктів з відповідними функціями, які приймають необхідні аргументи та створення сутностей Promise для асинхронної взаємодії, що дозволяє виконувати декілька запитів одночасно.

```
const Attr = [
   lst(),
   get(attr),
   val(attr),
   crt(name, type, single, listGroup),
   upd(attr),
   del(attr)
]
const List = [
   lst(),
   get(list),
   crt(list),
   upd(list, content),
   del(list)
]
const Type = [
   lst(),
   get(type),
   atr(type),
   crt(name, parent),
   upd(type),
   bind(type, attr),
   unbind(type, attr),
   del(type)
]
const Obj = [
   lst(),
   get(obj),
   getParams(obj),
   getParam(obj, attr),
   setParam(obj, attr, val),
   crt(name, typeId, parent),
   upd(obj), del:(obj),
   del(obj)
]
```

Деталі реалізації інтерфейсу взаємодії відображені у додатку Ж.

Для тестування навантаження та графічного відображення кількісних показників створено скрипт load.js з використанням бібліотеки Chart.js для побудови графіків. Його вміст представлено у додатку И. Даний скрипт має два режими роботи, а саме «легкий» – коли запит надсилається лише один за 2 секунди та «важкий» – коли створюється значна кількість паралельних запитів, кількість яких може бути змінено за допомогою методу `updateThreadCount(count)`, за замовчуванням виконується 30 паралельних запитів.

Особливих вимог до середовища де виконується скрипт у «легкому» режимі – немає. А ось для «важкого» режиму необхідний якісний зв'язок і достатня кількість обчислювальних ресурсів. Саме тому скрипт у цьому режимі буде запущений за допомогою хмарних браузерів, таких як Puffin[6] або AppOnFly[7]. Ці застосунки надають можливість виконувати скрипт у зовнішньому середовищі де є потужні обчислювальні ресурси та якісний канал зв'язку зі швидкістю близько 1 Гбіт/с.

### 4.2 Сценарії тестування

– Сценарій з використанням 50 паралельних запитів та з вимкненою фільтрацією, тобто з видаленим рядком `auth_request /queue;` із конфігурації NGINX.

– Сценарій з використанням 50 паралельних запитів та з вимкненим додатком фільтрації.

– Сценарій з незначним навантаженням, одночасно з використанням 20, 50, 100 паралельних запитів з зовнішнього середовища.

### 4.3 Виконання сценаріїв тестування

Сценарій 1. Навантаження 50 паралельних запитів без фільтрації.

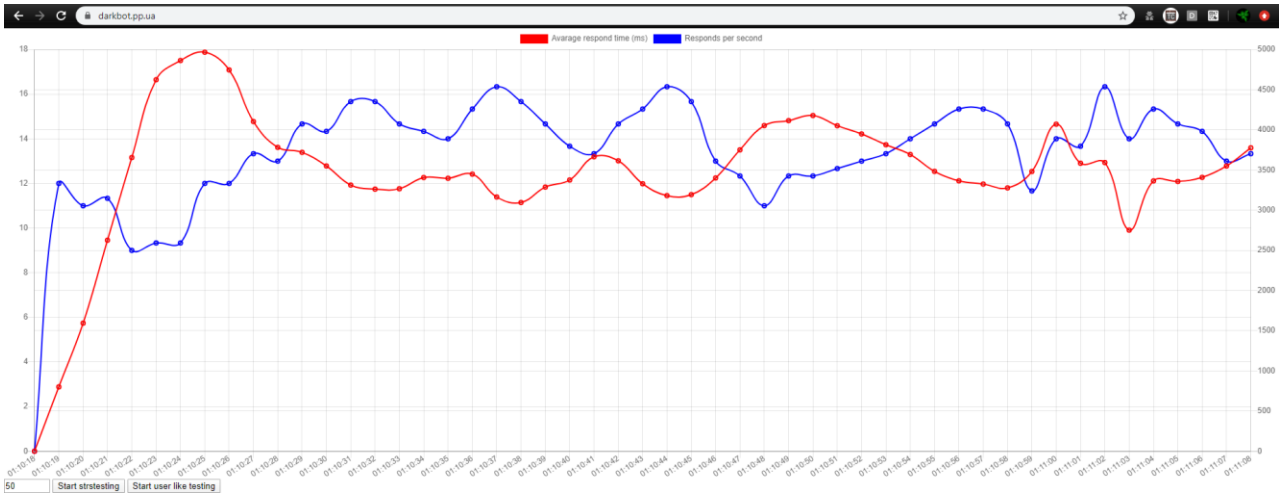


Рисунок 4.1 – Графік усередненого часу відповіді (червона лінія) та кількості оброблених запитів за секунду (синя лінія) для першого сценарію

Дані для побудови графіку наведені у таблиці М.1. Середній час відповіді – 3528,94 ms, середня кількість оброблених запитів – 13,62 rps.

Сценарій 2. Навантаження 50 паралельних запитів з недоступним додатком фільтрації.

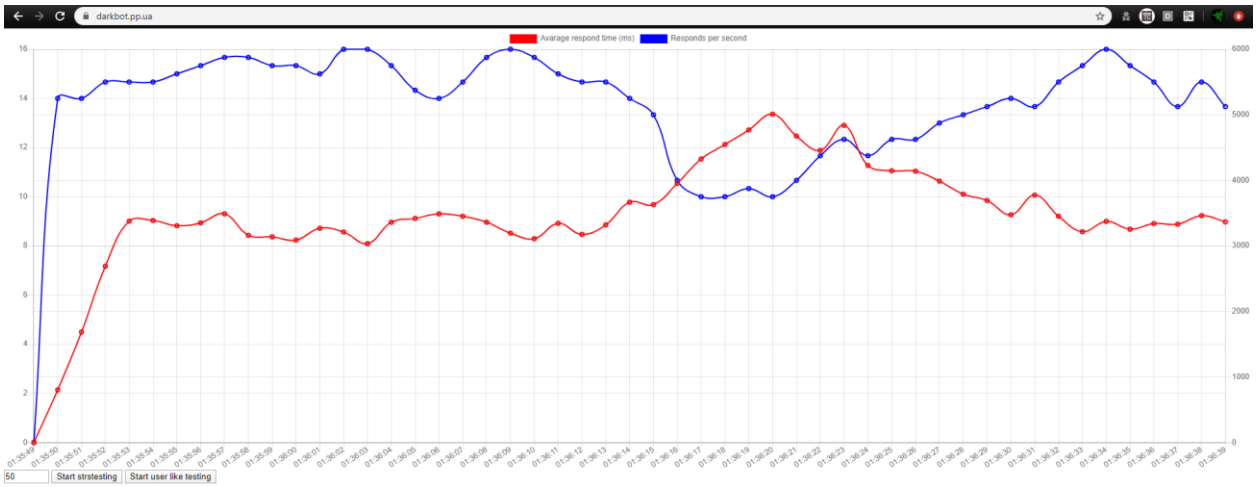


Рисунок 4.2 – Графік усередненого часу відповіді (червона лінія) та кількості оброблених запитів за секунду (синя лінія) для другого сценарію

Дані для побудови графіку наведені у таблиці М.2. Середній час відповіді – 3522,44 ms, середня кількість оброблених запитів – 13,92 rps.

Проміжний результат – середній час відповіді та кількість оброблених запитів значимим чином не змінилася, а отже можна стверджувати, що у випадках відмови, перезавантаження, відсутності та інших причин недоступності додатку контролю трафіку не несе в собі погіршення роботи серверу. Незначний приріст середніх значень у другому сценарії, у порівнянні з першим, можна пояснити нестабільністю мережі.

Сценарій 3. Додаток контролю працює у штатному режимі, визначаємо вплив сесії зі значним навантаженням різних величин на сесію з незначним типовим навантаженням.

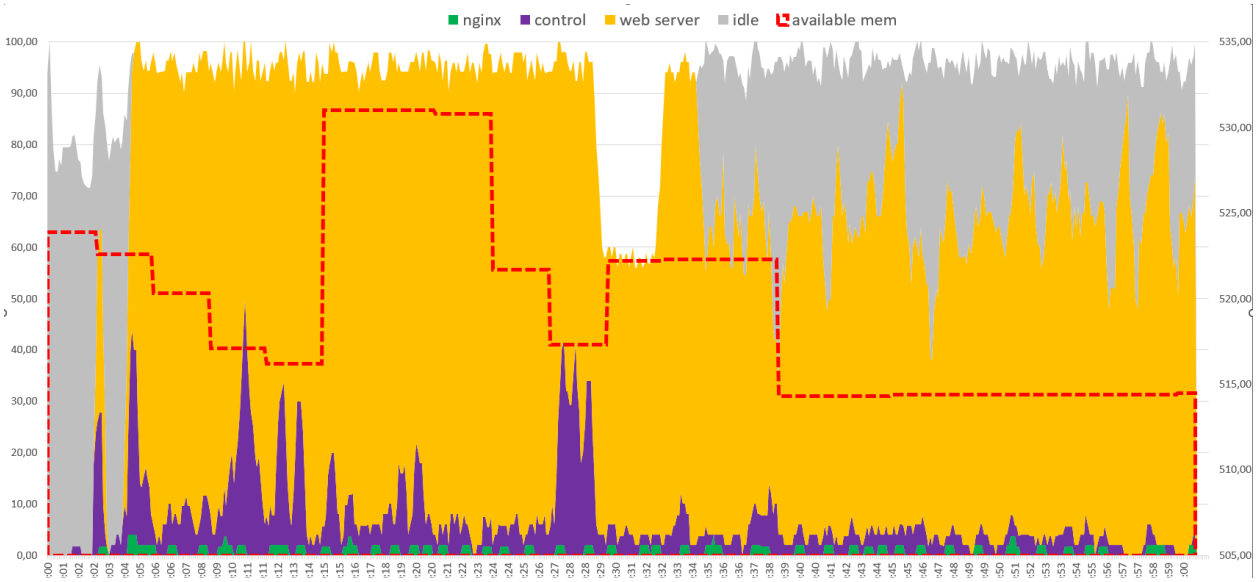


Рисунок 4.3 – Графік завантаженості CPU (%) і кількість вільної пам’яті (MB) під час усунення впливу надмірного навантаження

Данні використання процесору зібрані за допомогою команди:

```
top -p3181213 -p3594953 -p3589310 -d 0.1 -b -o PID >> perfom.txt
```

Де «3181213», «3594953» та «3589310» ідентифікатори досліджуваних процесів.

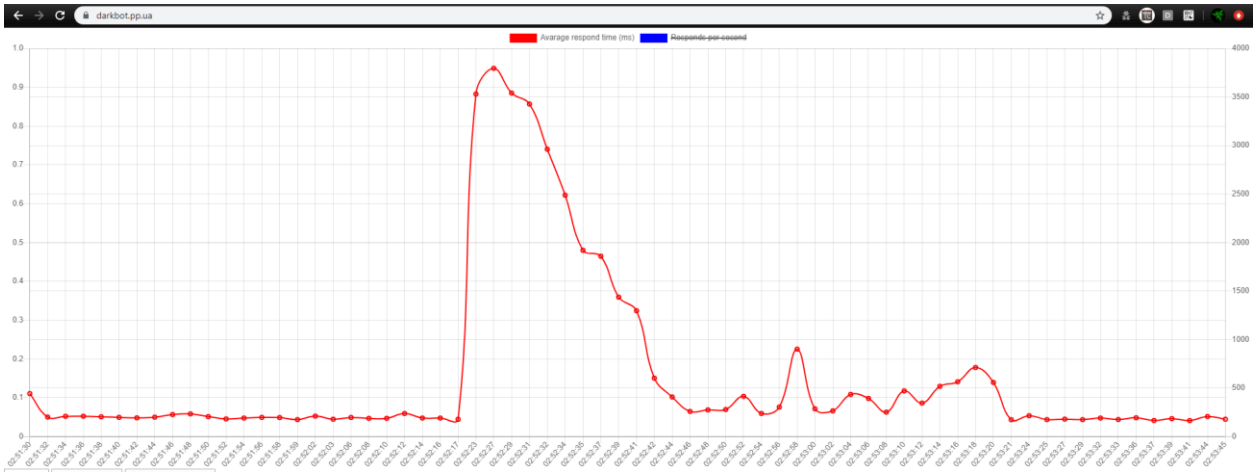


Рисунок 4.4 – Графік впливу паралельної сесії (x50) та усунення впливу за допомогою пріоритезації трафіку

Графік на рисунку 4.4 побудований на основі таблиці М.4.

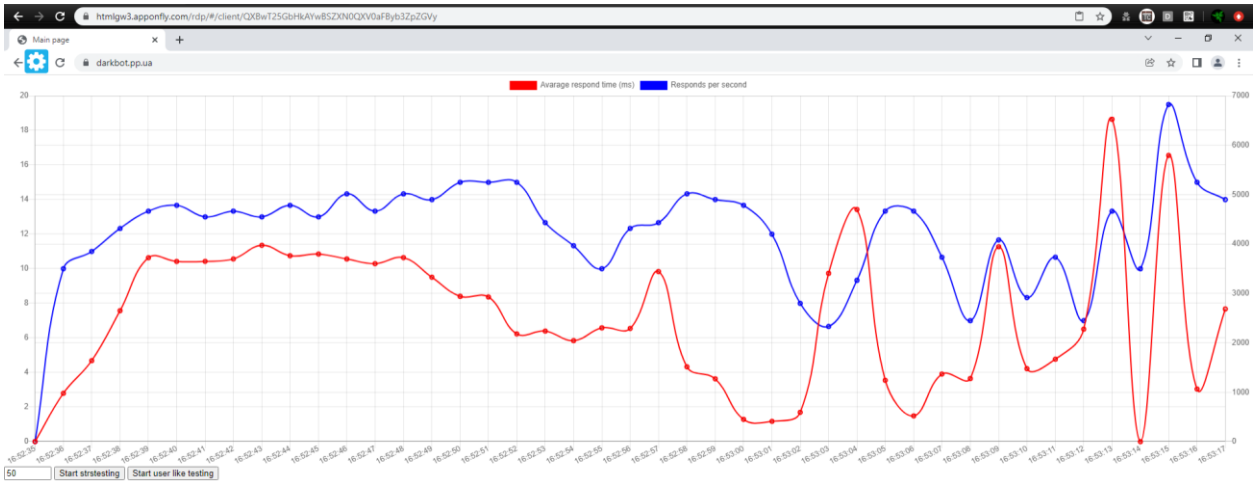


Рисунок 4.5 – Графік очікування відповідей сесії із навантаженням 50 паралельних запитів із-за впливу додатку контролю пріоритезації трафіку

Графік на рисунку 4.5 побудований на основі таблиці М.3.

Підсценарій з відмінністю кількості навантажуючих запитів – 20 на секунду.

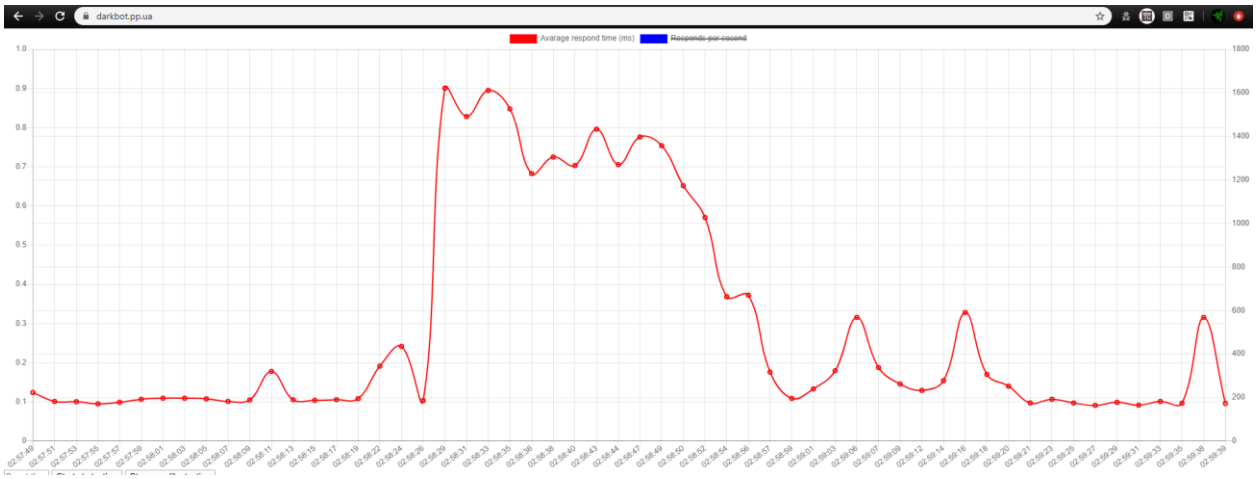


Рисунок 4.6 – Графік впливу паралельної сесії (x20) та усунення впливу за допомогою пріоритезації трафіку

Графік на рисунку 4.6 побудований на основі таблиці М.6.

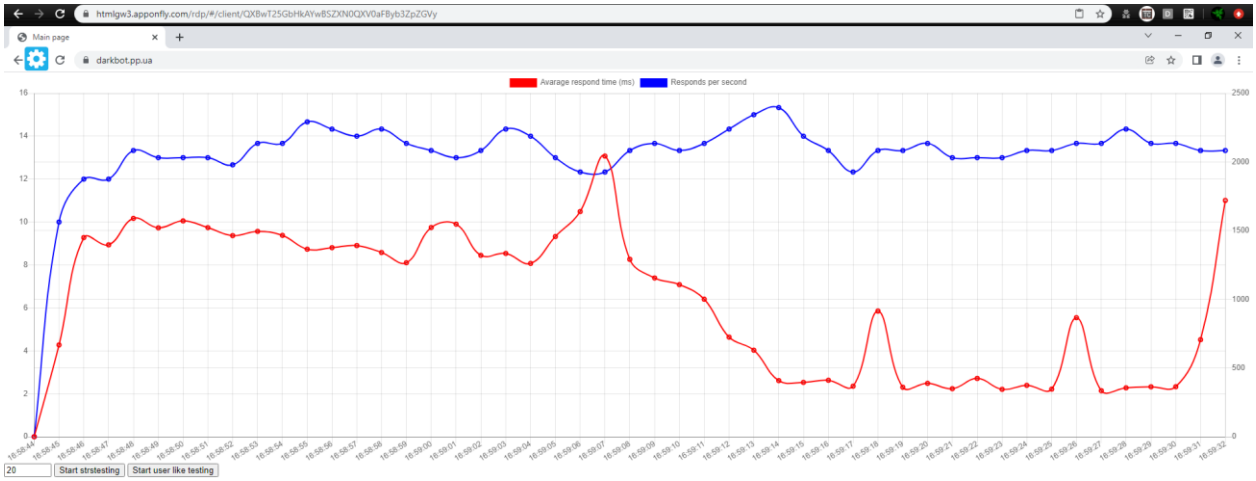


Рисунок 4.6 – Графік очікування відповідей сесії із навантаженням 20 паралельних запитів із-за впливу додатку контролю пріоритезації трафіку

Графік на рисунку 4.6 побудований на основі таблиці М.5.

Підсценарій з відмінністю кількості навантажуючих запитів – 100 на секунду.

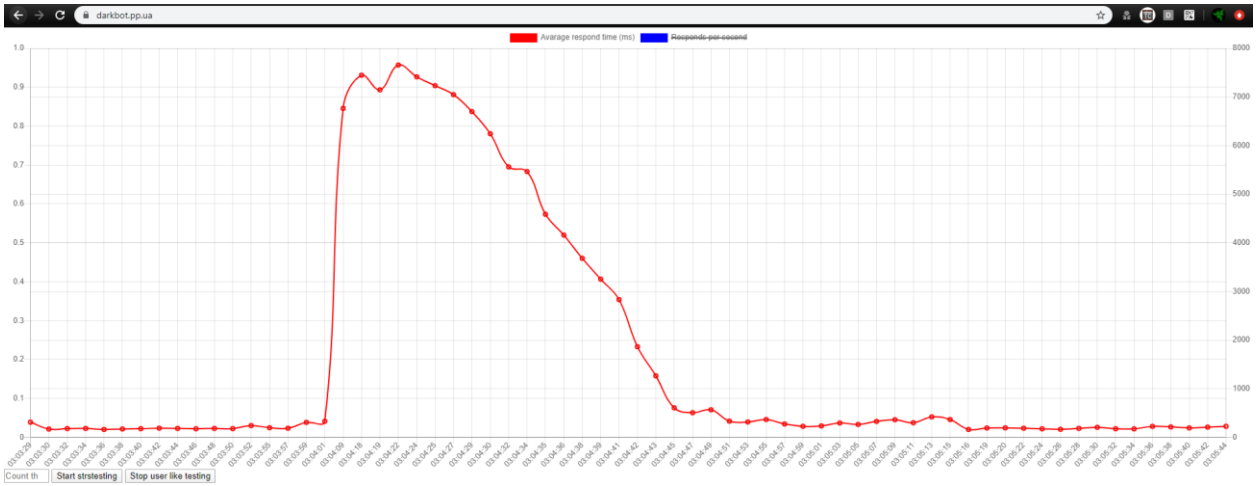


Рисунок 4.7 – Графік впливу паралельної сесії (x100) та усунення впливу за допомогою пріоритезації трафіку

Графік на рисунку 4.7 побудований на основі таблиці М.8.

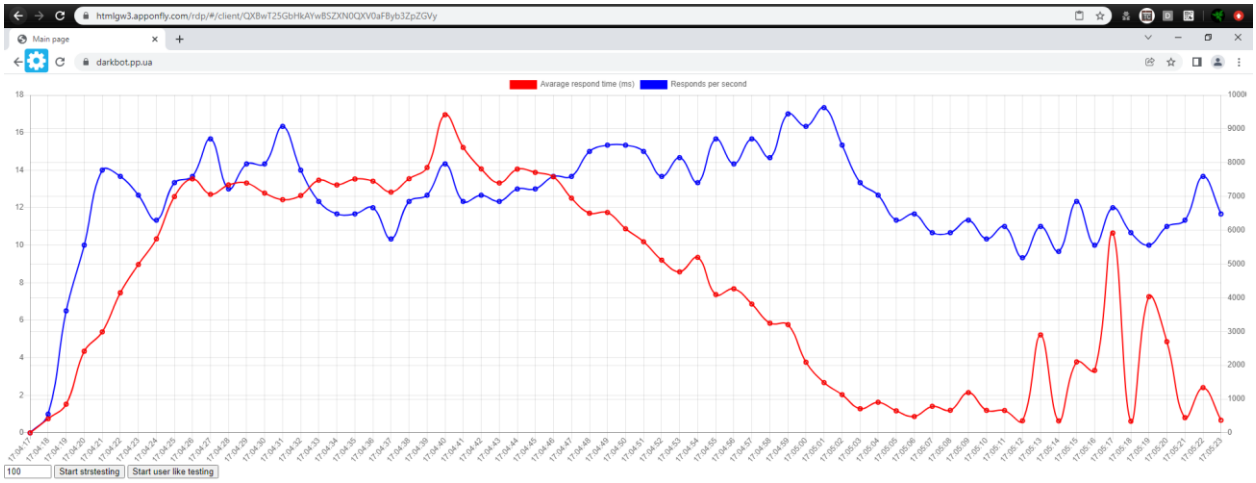


Рисунок 4.8 – Графік очікування відповідей сесії із навантаженням 100 паралельних запитів із-за впливу додатку контролю пріоритезації трафіку

Графік на рисунку 4.8 побудований на основі таблиці М.7.

### 4.4 Результати тестування

Результати тестування показали, що наявність додатку фільтрації трафіку немає значимого впливу на роботу веб-серверу у нормальному режимі. У разі перевантаження фільтрація аномального трафіку дозволяє за короткий час усунути вплив перевантаження за рахунок пріоритезації трафіку, що на має аномальної складової. За результатами перших двох сценаріїв тестування можна стверджувати, що у випадках відмови, перезавантаження, відсутності та інших причин недоступності додатку контролю трафіку не несе в собі погіршення роботи серверу. Результати третього сценарію показали, що:

– при навантаженні 20 запитів на секунду вдалося подолати перевантаження за 23 секунди.

– при навантаженні 50 запитів на секунду вдалося подолати перевантаження за 19 секунд.

– при навантаженні 100 запитів на секунду вдалося подолати перевантаження за 40 секунд.

# ВИСНОВКИ

У результаті магістерської роботи було розроблено технологію динамічної пріоритезації трафіку та продемонстровано її імплементацію у вигляді компоненти інтегрованої до NGINX в якості посередника керування асинхронним потоком. Метою роботи було забезпечити надійну та ефективну роботу ресурсу, покращити користувацький досвід та запобігти незапланованому використанню ресурсу. Для досягнення цієї мети було сформульовано основні цілі розробки застосунку, які полягають у розробці технології яка може бути адаптована для широкого спектру випадків де необхідна гнучкість рішень для покращення ефективності обробки запитів за рахунок фільтрації аномального трафіку.

Для досягнення зазначеної мети та виконання завдань було розроблено додаток для фільтрації аномального трафіку. А саме використано практичний підхід до розпізнавання та фільтрації аномального трафіку з використанням поточних даних користування ресурсом та застосування для них правил на основі бажаної моделі взаємодії користувачів з ресурсом. Такий підхід дозволяє забезпечити більш точну та ефективну фільтрацію аномалій та мінімізувати можливий негативний вплив.

У результаті дослідження було показано, що розроблений застосунок динамічної пріоритезації трафіку покращує ефективність та надійність ресурсу. Додаток було протестовано на декількох тестових випадках та результати видалися цілком успішними.

Впровадження розробленого застосунку має позитивний вплив на роботу ресурсу де відбуваються періодичні перевантаження. Зокрема, він певною мірою допомагає зменшити навантаження на мережу, забезпечити кращу пропускну здатність та забезпечити більш якісне надання ресурсів доброчесним користувачам.

Ця технологія може бути використана як ще один крок у забезпеченні надійності та ефективності роботи ресурсу. Дослідження показало, що цей застосунок допомагає забезпечити більш якісне та швидке обслуговування

користувачів та покращити загальну роботу ресурсу, а отже, впровадження розробленої технології у вигляді застосунку може бути одним із важливих етапів в розвитку та оптимізації роботи мережевого ресурсу.

Ця технологія є додатковим кроком у покращенні ефективності та надійності ресурсу. Вона має все необхідне, щоб стати основою для розвитку подібних ідей у майбутньому, забезпечуючи їх високу надійність нефункціональних вимог.

А саме напрацювання, проведені в ході цього дослідження, можуть бути використані для подальшого розвитку технологій динамічної пріоритезації трафіку у напрямку вивчення більшої кількості метрик та аномалій які трапляються в реальних системах на реальних даних. Такий напрямок розвитку допоможе покращити якість обслуговування користувачів, забезпечуючи надійність та ефективність функціонування мережі.

Отже, можна стверджувати, що результати дослідження підтверджують наукову новизну та важливість розробленого.

Результати цієї роботи мають наштовхнути на думку до практичного застосування цієї розробки для бізнесу та організацій, які мають потреби у забезпеченні якості обслуговування своїх користувачів та захисту доступності своїх ресурсів від незапланованого використання. Ця технологія може бути особливо корисною для телекомунікаційних компаній та постачальників інтернет-послуг, які стикаються зі значним обсягом трафіку та потребують ефективного управління ним.

Додатково, динамічна пріоритезація трафіку може бути корисною для компаній, які надають хмарні послуги або забезпечують доступ до великої кількості онлайн-ресурсів. Застосування розробки може допомогти таким компаніям зменшити затримки та покращити якість послуг для їхніх користувачів, забезпечуючи більш ефективну обробку трафіку та покращення доступності ресурсів.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Cloudflare, Web Application Firewall (WAF), Application Security. https://www.cloudflare.com/waf
2. ModSecurity, open source, cross platform web application firewall (WAF) engine for Apache, IIS and Nginx that is developed by Trustwave's SpiderLabs.ModSecurity. https://www.modsecurity.org
3. F5 Networks WAF. Proactive bot defense, and application-layer encryption of sensitive data. https://www.f5.com/products/big-ip-services/advanced-waf
4. Imperva Web Application Firewall. component of a comprehensive Web Application and API Protection (WAAP) stack that secures from edge to database. https://www.imperva.com/products/web-application-firewall-waf
5. Akamai Kona Site Defender. The industry-leading cloud-based web application firewall (WAF) — harnesses visibility across the Akamai Intelligent Edge. https://www.akamai.com/site/en/documents/product-brief/akamai-kona-site-defender-product-brief.pdf
6. Puffin Cloud Browser is the client app that accesses the cloud server which runs the web browser.. https://www.puffin.com
7. AppOnFly. Secure remote desktop for accessing professional apps. https://www.apponfly.com
8. KTOR. Framework for easily build web applications, HTTP services, mobile and browser applications. https://ktor.io/docs/welcome.html
9. George, A. S., & George, A. S. H. (2021). A Brief Study on The Evolution of Next Generation Firewall and Web Application Firewall. *International Journal of Advanced Research in Computer and Communication Engineering*, *10*(5).
10. Mauricio, L., & Rubinstein, M. (2023). A Network Function Virtualization Architecture for Automatic and Efficient Detection and Mitigation against Web Application Malware. *Journal of Internet Services and Applications*, *14*(1). https://doi.org/10.5753/jisa.2023.2847

11. Thang, N. M., Ho, T. P., & Nam, H. T. (2022). A new approach to improving web application firewall performance based on support vector machine method with analysis of Http request. *Journal of Science and Technology on Information Security*, *1*(15). https://doi.org/10.54654/isj.v1i15.842
12. Thang, N. M., Ho, T. P., & Nam, H. T. (2022). A new approach to improving web application firewall performance based on support vector machine method with analysis of Http request. *Journal of Science and Technology on Information Security*, *1*(15). https://doi.org/10.54654/isj.v1i15.842
13. Allen, L. (2015). Advanced Penetration Testing for Highly-Secured Environments: The Ultimate Security Guide. In *Climate Change 2013 - The Physical Science Basis* (Vol. 1).
14. Cheng, Z., Cui, B., Qi, T., Yang, W., & Fu, J. (2021). An Improved Feature Extraction Approach for Web Anomaly Detection Based on Semantic Structure. *Security and Communication Networks*, *2021*. https://doi.org/10.1155/2021/6661124
15. Handi Prasetyo. (2016). ANALISA KEAMANAN WEB SERVER MENGGUNAKAN WEB APPLICATION FIREWALL MODSECURITY Artikel Ilmiah. In *Fakultas Teknologi Informasi Universitas Kristen Satya Wacana*.
16. Pingale, S. v., & Sutar, S. R. (2022). Analysis of Web Application Firewalls, Challenges, and Research Opportunities. *Lecture Notes in Electrical Engineering*, *783*. https://doi.org/10.1007/978-981-16-3690-5_21
17. Cao, L. (2009). Data mining and multi-agent integration. In *Data Mining and Multi-Agent Integration*. https://doi.org/10.1007/978-1-4419-0522-2
18. Chen, H. C., Nshimiyimana, A., Damarjati, C., & Chang, P. H. (2021). Detection and prevention of cross-site scripting attack with combined approaches. *2021 International Conference on Electronics, Information, and Communication, ICEIC 2021*. https://doi.org/10.1109/ICEIC51217.2021.9369796

19. Sepczuk, M. (2023). Dynamic Web Application Firewall detection supported by Cyber Mimic Defense approach. *Journal of Network and Computer Applications*, *213*. https://doi.org/10.1016/j.jnca.2023.103596
20. Rao Vemuri, V. (2005). Enhancing Computer Security with Smart Technology. In *Enhancing Computer Security with Smart Technology*. https://doi.org/10.1201/9781420031225
21. Chakir, O., Sadqi, Y., & Maleh, Y. (2023). Evaluation of Open-source Web Application Firewalls for Cyber Threat Intelligence. In *Big Data Analytics and Intelligent Systems for Cyber Threat Intelligence*. https://doi.org/10.1201/9781003373384-3
22. Dhiatama Ayunda, K., Widjajarto, A., Budiono, A., Telkom, U., Telekomunikasi Terusan Buah Batu Bandung -, J., Sistem Informasi, J., & Rekayasa Industri, F. (2021). Implementation and Analysis ModSecurity on Web-Based Application with OWASP Standards. *Jurnal.Mdp.Ac.Id*, *8*(3).
23. Suryana, M. L. N., Muda, N. R. S., Minggu, D., Agustiady, R., & Herkariawan, C. (2021). Implementation of firewall for web server access management based on application gateway for TNI AD website. *IOP Conference Series: Materials Science and Engineering*, *1098*(2). https://doi.org/10.1088/1757-899x/1098/2/022105
24. Suryana, M. L. N., Muda, N. R. S., Minggu, D., Agustiady, R., & Herkariawan, C. (2021). Implementation of firewall for web server access management based on application gateway for TNI AD website. *IOP Conference Series: Materials Science and Engineering*, *1098*(2). https://doi.org/10.1088/1757-899x/1098/2/022105
25. Handa, A., Negi, R., & Shukla, S. K. (2021). Implementing Enterprise Cybersecurity With Open-Source Software and Standard Architecture. In *Implementing Enterprise Cybersecurity with Open-source Software and Standard Architecture*.
26. Nguyen, T. C. H., Le-Nguyen, M. K., Le, D. T., Nguyen, V. H., Tôn, L. P., & Nguyen-An, K. (2022). Improving Web Application Firewalls with

Automatic Language Detection. *SN Computer Science*, *3*(6). https://doi.org/10.1007/s42979-022-01327-2

27. Rizal, R., & Sumaryana, Y. (2021). Peningkatan Keamanan Aplikasi Web Menggunakan Web Application Firewall (WAF) Pada Sistem Informasi Manajemen Kampus Terintegrasi. *Jurnal ICT : Information Communication & Technology*, *20*(2). https://doi.org/10.36054/jict-ikmi.v20i2.416
28. Chen, X., Shen, Q., Cheng, P., Xiong, Y., & Wu, Z. (2022). RuleCache: Accelerating Web Application Firewalls by On-line Learning Traffic Patterns. *Proceedings - IEEE International Conference on Web Services, ICWS 2022*. https://doi.org/10.1109/ICWS55610.2022.00044
29. Jemal, I., Haddar, M. A., Cheikhrouhou, O., & Mahfoudhi, A. (2022). SWAF: A Smart Web Application Firewall Based on Convolutional Neural Network. *Proceedings of the 2022 15th IEEE International Conference on Security of Information and Networks, SIN 2022*. https://doi.org/10.1109/SIN56466.2022.9970545
30. Melchior, F., Kreutz, D., & Fiorenza, M. (2021). *Web Application Firewalls (WAFs): o impacto do número de regras na latência das requisições Web*. https://doi.org/10.5753/errc.2019.9229

# ДОДАТОК А

Повний текст конфігураційних файлів NGINX

## nginx.conf

```
user nginx;
worker_processes auto;

error_log  /var/log/nginx/error.log notice;
pid        /var/run/nginx.pid;

events {
  worker_connections  1024;
}

http {
  log_format mylog '{"ip":"$remote_addr", "duration":$request_time,
"time":"$time_local", "action":"$request", "code":$status,
"size":$body_bytes_sent}';
  server {
    listen 80;
    access_log /var/log/nginx/access.log mylog;

    include conf/CF.conf;

    location / {
      root /www/WEB-APP;
      autoindex on;
    }

    location /api/ {
      auth_request /queue;
      rewrite ^/api/?(.*)$ /$1 break;
      include conf/proxyset.conf;
      proxy_pass http://127.0.0.1:8080;
    }

    location /queue {
      internal;
      proxy_pass http://127.0.0.1:8090;
      proxy_pass_request_body off;
      include conf/proxyset.conf;
      error_page 502 =200 @success;
    }

    location @success {
      return 200 'Success';
    }
  }
}
```

## CF.conf

```
#Cloudflare ip addresses

  # - IPv4
    set_real_ip_from 173.245.48.0/20;
    set_real_ip_from 103.21.244.0/22;
    set_real_ip_from 103.22.200.0/22;
    set_real_ip_from 103.31.4.0/22;
    set_real_ip_from 141.101.64.0/18;
    set_real_ip_from 108.162.192.0/18;
    set_real_ip_from 190.93.240.0/20;
    set_real_ip_from 188.114.96.0/20;
    set_real_ip_from 197.234.240.0/22;
    set_real_ip_from 198.41.128.0/17;
    set_real_ip_from 162.158.0.0/15;
    set_real_ip_from 104.16.0.0/13;
    set_real_ip_from 104.24.0.0/14;
    set_real_ip_from 172.64.0.0/13;
    set_real_ip_from 131.0.72.0/22;

  # - IPv6
    set_real_ip_from 2400:cb00::/32;
    set_real_ip_from 2606:4700::/32;
    set_real_ip_from 2803:f800::/32;
    set_real_ip_from 2405:b500::/32;
    set_real_ip_from 2405:8100::/32;
    set_real_ip_from 2a06:98c0::/29;
    set_real_ip_from 2c0f:f248::/32;

real_ip_header CF-Connecting-IP;
```

## proxyset.conf

```
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Original-URI $request_uri;
proxy_set_header X-Original-Method $request_method;
```

# ДОДАТОК Б

Вихідний код додатку контролю трафіку

## Init.java

```
import java.io.IOException;
import java.io.InputStream;
import java.util.*;
import java.util.stream.Collectors;

public class Init {
    public static void main(String[] args) throws IOException {
        LogHandler.init();
        PluginHandler.loadPlugins();
        Queue.setMaxSize(100);
        Queue.Categorization.initPool(5);
        Controller.initController();
    }
}
```

## Config.java

```
package org.config;

import com.google.gson.Gson;

import java.io.FileReader;
import java.util.HashMap;
import java.util.Map;

public class Config {
    public static final Map<String, Object> config = new HashMap<>();
    static {
        try {
            config.putAll(new Gson().fromJson(new FileReader("config.json"),
Map.class));
        } catch (Exception e) {
            System.err.println(e.getMessage());
        }
    }

    public static Object getValue(String key){
        return config.get(key);
    }
}
```

## LogHandler.java

```
import com.google.gson.Gson;
import com.google.gson.JsonSyntaxException;
import org.config.Config;

import java.io.IOException;
import java.io.InputStream;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.stream.Collectors;

public class LogHandler {
  private static final SimpleDateFormat format = new
SimpleDateFormat("dd/MMM/yyyy:hh:mm:ss Z", Locale.ENGLISH);
  static final Map<String, Object> accessLogConfig = (Map<String, Object>)
Config.getValue("accessLog");
  static final String file = getValue("FilePath");
static final int historyTime = Integer.parseInt(getValue("HistoryTime","0"));
static final int historyRecords = Integer.parseInt(getValue("HistoryRecords",
"0"));
  private static final LinkedList<LogRecord> logs = new LinkedList<>();

  static void init() {
    if (file != null) {
      Gson gson = new Gson();
      Thread thread = new Thread(() -> {
        while(true)
          try {
            ProcessBuilder pb =
                  new ProcessBuilder("tail", "-n", "0", "-f", file);
            try(InputStream is = pb.start().getInputStream();
                Scanner s = new Scanner(is).useDelimiter("\\n")){
              while(true) {
                try {
                  if (s.hasNext()) {
                    logs.addFirst(gson.fromJson(s.next(), LogRecord.class));
                    if (historyRecords > 0 && logs.size() > historyRecords)
                      logs.removeLast();
                  }
                } catch(JsonSyntaxException e){e.printStackTrace();}
              }
            }
          } catch (IOException e){
            e.printStackTrace();
          try {
            Thread.sleep(5000);
          } catch (InterruptedException ex) {}
        }
      }, "LogHandlerThread");
      thread.setDaemon(true);
      thread.start();
    }
  }

  private static String getValue(String key, String... def){
    if(accessLogConfig == null || !accessLogConfig.containsKey(key))
      return def.length != 0 ? def[0] : null;
    return (String) accessLogConfig.get(key);
  }

  public static int getMinRespondTime(long milliseconds){
    long timePoint = new Date().getTime() - milliseconds;
```

```
    float min = Float.MAX_VALUE;
    for (LogRecord log : logs)
      if(log.getDate().getTime() > timePoint && log.duration < min)
      min = log.duration;
    return (int) (min*1000);
  }

  public static List<LogRecord> getRecordsByIP(String ip){
    return logs.stream()
      .filter(logRecord -> Objects.equals(logRecord.ip, ip))
      .collect(Collectors.toList());
  }

  public static class LogRecord {
  public final int code, size;
  public final String ip, time, action;
  public final float duration;

  public LogRecord(int code, int size, String ip, String time,
                   String action, float duration) {
    this.code = code;
    this.size = size;
    this.ip = ip;
    this.time = time;
    this.action = action;
    this.duration = duration;
  }

  Date getDate() {
    try {
      return format.parse(time);
    } catch (ParseException e) {
        throw new RuntimeException(e);
      }
    }
  }
}
```

## Plugin.java

```
package org.plugins;

import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;

@Retention(RetentionPolicy.RUNTIME)
public @interface Plugin {
}
```

## RulePlugin.java

```
package org.plugins;

import java.util.Map;

public interface RulePlugin {
    int applyRule(Map<String, String> headers);
}
```

## PluginHandler.java

```
import org.config.Config;
import org.plugins.Plugin;
import org.plugins.RulePlugin;

import java.util.*;

public class PluginHandler {
    private static final Set<Class<?>> plugins = new HashSet<>();
    private static final Set<RulePlugin> rules = new HashSet<>();
    public static void loadPlugins(){
        try {
            List<String> pluginsConfig =
                        (List<String>) Config.getValue("plugins");
            if(pluginsConfig == null) return;
            pluginsConfig.stream()
                    .map(PluginHandler::loadClass)
                    .filter(Objects::nonNull)
                    .filter(p -> p.isAnnotationPresent(Plugin.class))
                    .forEach(PluginHandler.plugins::add);
            plugins.forEach(PluginHandler::init);
            rules.clear();
            plugins.stream().map(PluginHandler::toRule).forEach(rules::add);
        } catch (Exception e){
            e.printStackTrace();
            System.exit(1);
        }
    }
    public static int applyAllRules(Map<String, String> headers){
        return rules.stream()
                    .map(p-> p.applyRule(headers))
                    .max(Integer::compareTo)
                    .orElse(0);
    }
    private static Class<?> loadClass(String className){
        try {
            return Class.forName(className);
        }catch (Exception e){
            return null;
        }
    }
    private static void init(Class<?> c){
        try {
            c.getMethod("init").invoke(null);
        } catch (Exception ignored){}
    }
    private static RulePlugin toRule(Class<?> c){
        try {
            Object o = c.newInstance();
            return (o instanceof RulePlugin) ? (RulePlugin) o : null;
        } catch (Exception e){
            return null;
        }
    }
}
```

## LinkedQueueList.java

```
import java.util.Comparator;
import java.util.HashMap;
import java.util.Map;

public class LinkedQueueList<E> {
    private Node first;
    private Node last;
    private int size = 0;

    private final Map<Integer, Node> hashTable = new HashMap<>();

    public int size(){
        return size;
    }

    public synchronized void add(E obj, int order){
        Node node = new Node(obj, order);
        if(first == null || last == null) {
            first = node;
            last = node;
            hashTable.put(node.order, node);
        } else if(first.order >= node.order) {
            if(first.order > node.order) hashTable.put(node.order, node);
            node.next = first;
            first = node;
        } else if(last.order < node.order){
            node.prev = last;
            last = node;
            hashTable.put(node.order, node);
        } else {
            boolean updateHashTable = false;
            Node nextNode;
            if(hashTable.containsKey(node.order))
                nextNode = hashTable.get(node.order);
            else {
                updateHashTable = true;
                nextNode = hashTable.entrySet().stream()
                        .filter(i -> i.getKey() < node.order)
                        .max(Comparator.comparingInt(Map.Entry::getKey))
                        .get()
                        .getValue();
            }
            Node prevNode = nextNode.prev;
            prevNode.next = node;
            nextNode.prev = node;
            node.prev = prevNode;
            node.next = nextNode;
            if(updateHashTable) hashTable.put(node.order, node);
        }
        size++;
    }

    public synchronized E removeLast(){
        if(last == null) return null;
        if(first == last) first = null;
        Node node = last;
        if(node.order == node.prev.order)
            hashTable.put(node.order, node.prev);
        else hashTable.remove(node.order);
        last = last.prev;
        size--;
        return node.obj;
```

```
    }

    public synchronized E removeFirst(){
        if(first == null) return null;
        if(first == last) last = null;
        Node node = first;
        if(hashTable.get(node.order) == node) hashTable.remove(node.order);
        first = first.next;
        size--;
        return node.obj;
    }

    public class Node {
        int order;
        Node prev;
        Node next;
        final E obj;
        public Node(E obj, int order) {
            this.obj = obj;
            this.order = order;
        }
    }
}
```

## Queue.java

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.util.*;
import java.util.concurrent.ConcurrentLinkedQueue;
import java.util.stream.Collectors;

public class Queue {
    private static int queueSize = 100;
    private static final LinkedQueueList<Socket> QUEUE = new
LinkedQueueList<>();
    private static final ConcurrentLinkedQueue<Socket> UNCATEGORIZED = new
ConcurrentLinkedQueue<>();
    private static final byte[] ALLOW_RESPONSE = "HTTP/1.1 204 No
Content\r\nContent-Length: 0\r\nConnection: Closed\r\n\r\n".getBytes();
    private static final byte[] DENY_RESPONSE = "HTTP/1.1 403
Forbidden\r\nContent-Length: 0\r\nConnection: Closed\r\n\r\n".getBytes();

    public static void push(Socket s) {
        UNCATEGORIZED.offer(s);
    }
    public static void setMaxSize(int count){
        if(count < 1) return;
        queueSize = count;
    }
    public static boolean release(){
        Socket first = QUEUE.removeFirst();
        if(first == null) return false;
        if(!send(first, true)) return release();
        return true;
    }
    private static boolean trim(){
        Socket last = QUEUE.removeLast();
```

```
        if(last == null) return false;
        return send(last, false);
    }
    static boolean send(Socket s, boolean allow){
        if(!s.isConnected()) return false;
        try(Socket ss = s; OutputStream os = ss.getOutputStream()){
            os.write(allow ? ALLOW_RESPONSE : DENY_RESPONSE);
            os.flush();
            return true;
        } catch (IOException e){
            return false;
        }
    }

    public static int size() {
        return QUEUE.size();
    }

    public static class Categorization implements Runnable {
        public static void initPool(int countThreads){
            for (int i = 0; i < countThreads; i++) {
                Thread t = new Thread(new Categorization(),
                                      "Categorization#"+i);
                t.setDaemon(true);
                t.start();
            }
        }
        private Categorization(){}

        @Override
        public void run() {
            while(true){
                try {
                    Socket a = UNCATEGORIZED.remove();
                    int order =
PluginHandler.applyAllRules(parseHeaders(a.getInputStream()));
                    QUEUE.add(a, order);
                    while(QUEUE.size() > queueSize) trim();
                } catch (NoSuchElementException e) {
                    try { Thread.sleep(100);
                    } catch (InterruptedException ignored) {}
                } catch (IOException | NullPointerException ignored) {}
            }
        }
    }
    private static Map<String, String> parseHeaders(InputStream inputStream)
throws IOException {
        List<String> metadataLines = new ArrayList<>();
        StringBuilder metadataBuilder = new StringBuilder();
        boolean wasNewLine = true;
        int lineNumber = 1;
        int b;
        while ((b = inputStream.read()) >= 0) {
            if (b == '\r') {
                int next = inputStream.read();
                if (next < 0 || next == '\n') {
                    lineNumber++;
                    if (wasNewLine) break;
                    metadataLines.add(metadataBuilder.toString());
                    if (next < 0) break;
                    metadataBuilder = new StringBuilder();
                    wasNewLine = true;
                } else {
                    inputStream.close();
```

```
                    throw new RuntimeException("Illegal char" + lineNumber);
                }
            } else if (b == '\n') {
                lineNumber++;
                if (wasNewLine) break;
                metadataLines.add(metadataBuilder.toString());
                metadataBuilder = new StringBuilder();
                wasNewLine = true;
            } else {
                metadataBuilder.append((char) b);
                wasNewLine = false;
            }
        }

        if (metadataBuilder.length() > 0) {
            metadataLines.add(metadataBuilder.toString());
        }

        return metadataLines.stream()
                 .map(s -> s.split(":", 2))
                 .collect(Collectors.toMap(a -> a[0],
                                           a -> a.length>1 ? a[1].trim(): "")
                 );
    }
}
```

## HealthMonitor.java

```
public class HealthMonitor {
    private static int lastVal;
    private static long lastCheck;
    public static synchronized int getServerHealth(){
        if(lastCheck < System.currentTimeMillis()-200) {
            lastCheck = System.currentTimeMillis();
            int val = LogHandler.getMinRespondTime(5000);
            lastVal = logFunc(val);
            if(lastVal < 80) {
                if (Queue.size() < 2) Controller.window += 50;
                else Controller.window++;
            } else if(Controller.window > 10 && lastVal > 85 )
                 Controller.window--;
        }
        return lastVal;
    }
    private static int logFunc(int val){
        return (int) Math.max(Math.min(78.169774-
32.123215*Math.log(val/1000.), 100), 0);
    }
}
```

## Controller.java

```
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

public class Controller {
    public static volatile int window = 100;
    public static void initController() throws IOException {
        final ServerSocket ss = new ServerSocket(8090);
        Thread thread = new Thread(() -> {
            while (true) try {
                Socket s = ss.accept();
                if(HealthMonitor.getServerHealth() > 90 && window < 200)
                     Queue.send(s, true);
                else Queue.push(s);
            } catch (Exception ignored) {}
        }, "ServerSocketThread");
        thread.setDaemon(true);
        thread.start();

        while(true){
            try {
                while(Queue.size() != 0){
                    Queue.release();
                    Thread.sleep(window/10);
                }
                Thread.sleep(100);
            } catch (InterruptedException e) {
                throw new RuntimeException(e);
            }
        }
    }
}
```

# ДОДАТОК В

Приклад простого плагіну

```
import org.plugins.Plugin;
import org.plugins.RulePlugin;

import java.util.Map;

@Plugin
public class DemoPlugin implements RulePlugin {
    @Override
    public int applyRule(Map<String, String> headers) {
        int out = 0;
        long prev = 0;
        for (LogHandler.LogRecord log :
LogHandler.getRecordsByIP(headers.get("X-Real-IP"))) {
            if(out > 100) break;
            long time = log.getDate().getTime();
            if(time - prev < 1000) out++;
            prev = time;
        }
        return out;
    }
}
```

# ДОДАТОК Г

## Maven конфігурація для додатку контролю трафіку

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>controller</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>
    <dependencies>
        <dependency>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
            <version>2.10.1</version>
        </dependency>
    </dependencies>
</project>
```

# ДОДАТОК Д

Вихідний код додатку веб-серверу

## Server.kt

```
package com.example

import io.ktor.http.*
import io.ktor.serialization.gson.*
import io.ktor.server.application.*
import io.ktor.server.engine.*
import io.ktor.server.netty.*
import io.ktor.server.plugins.contentnegotiation.*
import io.ktor.server.request.*
import io.ktor.server.response.*
import io.ktor.server.routing.*
import org.jetbrains.exposed.sql.transactions.transaction
import java.text.DateFormat

fun main() {
    init()
    embeddedServer(Netty, port = 8080) { main() }.start(wait = true)
}

fun Application.main() {
    install(ContentNegotiation) {
        gson {
            setDateFormat(DateFormat.LONG)
            setPrettyPrinting()
            disableHtmlEscaping()
        }
    }
    routing {
        route("/attr") {
            get {
                call.respond(transaction { Attributes.all() })
            }
            get("/{attr}") {
                call.respond(transaction {
Attributes.byId(call.param("attr")) } ?: HttpStatusCode.NotFound)
            }
            get("/{attr}/values") {
                call.respond(transaction {
Attributes.byId(call.param("attr"))?.values() } ?: HttpStatusCode.NotFound)
            }
            post {
                call.respond(HttpStatusCode.Created,
call.receive<Attribute>().let { transaction { Attribute(it) } })
            }
            put {
                call.respond(call.receive<Attribute>().run { transaction {
update() } })
            }
            delete("/{attr}") {
                call.respond(if (transaction {
Attributes.delete(call.param("attr")) } != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
            }
        }

        route("/list") {
            get {
```

```
                call.respond(transaction { ListValues.all() })
            }
            get("/{list}") {
                call.respond(transaction {
ListValues.byId(call.param("list")) })
            }
            post {
                call.respond(
                    HttpStatusCode.Created,
                    call.receive<List<String>>().let { transaction {
ListValues.create(it) } })
            }
            put("/{list}") {
                call.respond(
                    call.receive<Map<String, String>>()
                        .run { transaction {
ListValues.update(call.param("list"), this@run) } })
            }
            delete("/{list}") {
                call.respond(if (transaction {
ListValues.delete(call.param("list")) } != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
            }
        }

        route("/type") {
            get {
                call.respond(transaction { ObjectTypes.all() })
            }
            get("/{type}") {
                call.respond(transaction {
ObjectTypes.byId(call.param("type")) } ?: HttpStatusCode.NotFound)
            }
            get("/{type}/attrs") {
                call.respond(transaction {
ObjectTypes.byId(call.param("type"))?.getAttributes() }
                    ?: HttpStatusCode.NotFound)
            }
            post {
                call.respond(HttpStatusCode.Created,
call.receive<ObjectType>().let { transaction { ObjectType(it) } })
            }
            put {
                call.respond(HttpStatusCode.OK,
call.receive<ObjectType>().run { transaction { update() } })
            }
            put("/{type}/bind/{attr}") {
                call.respond(if ((transaction {
                        Attributes.byId(call.param("attr"))?.let {
ObjectTypes.byId(call.param("type"))?.bind(it) }
                    } ?: 0) != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
            }
            put("/{type}/unbind/{attr}") {
                call.respond(if ((transaction {
                        Attributes.byId(call.param("attr"))?.let {
ObjectTypes.byId(call.param("type"))?.unbind(it) }
                    } ?: 0) != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
            }
            delete("/{type}") {
                call.respond(if (transaction {
ObjectTypes.delete(call.param("type")) } != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
```

```
            }
        }

        route("/obj") {
            get {
                call.respond(transaction { Objects.all() })
            }
            get("/{obj}") {
                call.respond(transaction { Objects.byId(call.param("obj")) }
?: HttpStatusCode.NotFound)
            }
            get("/{obj}/params") {
                call.respond(transaction {
Objects.byId(call.param("obj"))?.getParams() } ?: HttpStatusCode.NotFound)
            }
            get("/{obj}/param/{paramId}") {
                call.respond(transaction {
Objects.byId(call.param("obj"))?.getParam(call.param("paramId")) }
                    ?: HttpStatusCode.NotFound)
            }
            post("/{obj}/param/{paramId}") {
                call.respond(
                    call.receive<List<String>>().run {
                        if (transaction {

Objects.byId(call.param("obj"))?.setParam(call.param("paramId"), this@run)
                            } != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound
                    })
            }
            post {
                call.respond(HttpStatusCode.Created, call.receive<Obj>().let
{ transaction { Obj(it) } })
            }
            put {
                call.respond(HttpStatusCode.OK, call.receive<Obj>().run {
transaction { update() } })
            }
            delete("/{obj}") {
                call.respond(if (transaction {
Objects.delete(call.param("obj")) } != 0) HttpStatusCode.OK else
HttpStatusCode.NotFound)
            }
        }

    }
}
private fun ApplicationCall.param(id: String) =
this.parameters[id].toString()
```

## Utils.kt

```
package com.example
import java.util.*
import java.util.stream.Stream
import kotlin.reflect.KFunction1

fun<T> iterate(seed: T?, operator: KFunction1<T, T?>): Stream<T> {
    return LinkedList<T>().apply {
        var i : T? = seed
        while (i?.also { add(it) } != null) i = operator.invoke(i)
    }.stream()
}
```

## MetaModel.kt

```
package com.example

import com.example.AttributeTypes.*
import org.jetbrains.exposed.sql.*
import org.jetbrains.exposed.sql.statements.InsertStatement
import org.jetbrains.exposed.sql.statements.api.ExposedBlob
import org.jetbrains.exposed.sql.transactions.transaction
import java.util.*
import kotlin.collections.HashMap
import kotlin.streams.toList

enum class AttributeTypes { TEXT, LIST, BLOB, REFERENCE }

fun init() {
    Database.connect("jdbc:sqlite:file.db", "org.sqlite.JDBC")
    transaction {
        SchemaUtils.create(Attributes)
        SchemaUtils.create(ObjectTypes)
        SchemaUtils.create(ObjectsTypesAttributes)
        SchemaUtils.create(Objects)
        SchemaUtils.create(Params)
        SchemaUtils.create(References)
    }
}

fun getUUID(prefix: String = ""): String{
    return prefix+UUID.randomUUID().toString().substring(prefix.length)
}

object ListValues: Table() {
    val listId = char("list_id", 36).uniqueIndex().clientDefault {
getUUID("LST") }
    val listGroup = char("list_group", 36).clientDefault { getUUID("GRP") }
    val name = varchar("name", 200)

    fun all() = selectAll().map { Triple(it[listGroup], it[listId], it[name])
}.run {
        HashMap<String, HashMap<String, String>>().apply { this@run.forEach {
getOrPut(it.first, ::HashMap)[it.second] = it.third } }
    }
    fun byId(groupId: String) = select { listGroup eq groupId }.map {
ListValue(it) }.associate { it.listId to it.name }
    fun create(list: List<String>) = getUUID("GRP").apply { batchInsert(list)
{ this[name] = it; this[listGroup] = this@apply } }.run { mapOf(Pair(this,
byId(this))) }
    fun update(list: String, map: Map<String, String>) = byId(list).run {
        map.filter { it.key in keys }.forEach { entry ->
this@ListValues.update ({ listId eq entry.key }) {it[name] = entry.value} }
        map.filter { it.key !in keys }.run {
this@ListValues.batchInsert(this.values) {this[name] = it; this[listGroup] =
list} }
        filter { it.key !in map.keys }.let { this@ListValues.deleteWhere {
listId inList it.keys } }
    }.run { byId(list) }
    fun delete(groupId: String) = deleteWhere { listGroup eq groupId }.also {
Attributes.deleteWhere { Attributes.listGroup eq groupId } }
}
data class ListValue(val listId: String = "", val listGroup: String = "", val
name: String){
    constructor(value: ResultRow) : this(value[ListValues.listId],
value[ListValues.listGroup], value[ListValues.name])
}
```

```
object Attributes: Table() {
    val attrId = char("attr_id", 36).uniqueIndex().clientDefault {
getUUID("ATR") }
    val type = enumeration("type", AttributeTypes::class)
    val listGroup = char("list_group", 36).references(ListValues.listGroup,
onDelete = ReferenceOption.CASCADE).nullable()
    val name = varchar("name", 200)
    val single = bool("single")

    fun all() : List<Attribute> = selectAll().map { Attribute(it) }
    fun byId(attrId: String) : Attribute? = select { Attributes.attrId eq
attrId }.map { Attribute(it) }.firstOrNull()
    fun delete(attrId: String) = deleteWhere { Attributes.attrId eq attrId }
        .also { ObjectsTypesAttributes.deleteWhere {
ObjectsTypesAttributes.attrId eq attrId } }
        .also { Params.deleteWhere { Params.attrId eq attrId } }
        .also { References.deleteWhere { References.attrId eq attrId } }
}
data class Attribute(val attrId: String = "", val type: AttributeTypes, val
listGroup: String?, val name: String, val single: Boolean){
    constructor(value: ResultRow) : this(value[Attributes.attrId],
value[Attributes.type], value[Attributes.listGroup], value[Attributes.name],
value[Attributes.single])
    constructor(value: InsertStatement<Number>) :
this(value[Attributes.attrId], value[Attributes.type],
value[Attributes.listGroup], value[Attributes.name],
value[Attributes.single])
    constructor(value: Attribute) : this(Attributes.insert {it[type] =
value.type; it[listGroup] = value.listGroup; it[name] = value.name;
it[single] = value.single })

    fun update(): Attribute {
        Attributes.update({ Attributes.attrId eq attrId }) { it[type] =
this@Attribute.type; it[listGroup] = this@Attribute.listGroup; it[name] =
this@Attribute.name; it[single] = this@Attribute.single }
        return this
    }

    fun values(): List<ListValue> {
        if(type != LIST || listGroup == null) return Collections.emptyList()
        return ListValues.select { ListValues.listGroup eq
this@Attribute.listGroup }.map { ListValue(it) }
    }
}

object ObjectTypes: Table() {
    val typeId = char("type_id", 36).uniqueIndex().clientDefault {
getUUID("TYP") }
    val parent = char("parent", 36).references(typeId, onDelete =
ReferenceOption.CASCADE).nullable()
    val name = varchar("name", 200)

    fun all() : List<ObjectType> = selectAll().map { ObjectType(it) }
    fun byId(typeId: String) : ObjectType? = select { ObjectTypes.typeId eq
typeId }.map { ObjectType(it) }.firstOrNull()
    fun byParent(typeId: String) : List<ObjectType> = select { parent eq
typeId }.map { ObjectType(it) }
    fun delete(typeId: String): Int {
        return deleteWhere { ObjectTypes.typeId eq typeId }
            .apply { ObjectsTypesAttributes.deleteWhere {
ObjectsTypesAttributes.typeId eq typeId } }
            .apply { Objects.deleteWhere { Objects.typeId eq typeId } }
            .apply { byParent(typeId).forEach { delete(it.typeId) } }
```

```
    }
}
data class ObjectType(val typeId: String = "", val parent: String?, val name:
String){
    constructor(value: ResultRow) : this(value[ObjectTypes.typeId],
value[ObjectTypes.parent], value[ObjectTypes.name])
    constructor(value: InsertStatement<Number>) :
this(value[ObjectTypes.typeId], value[ObjectTypes.parent],
value[ObjectTypes.name])
    constructor(value: ObjectType) : this(ObjectTypes.insert {it[parent] =
value.parent; it[name] = value.name })

    fun update(): ObjectType{
        ObjectTypes.update({ ObjectTypes.typeId eq typeId }) { it[parent] =
this@ObjectType.parent; it[name] = this@ObjectType.name }
        return this
    }

    fun getAttributes(attrType: List<AttributeTypes> =
Collections.emptyList()): List<Attribute> =
ObjectsTypesAttributes.join(Attributes, JoinType.INNER,
ObjectsTypesAttributes.attrId, Attributes.attrId)
        .select {(if(attrType.isNotEmpty()) Attributes.type inList attrType
else Op.TRUE) and (ObjectsTypesAttributes.typeId inList getParents().map {
it.typeId })}.distinct().map { Attribute(it) }
    fun getParents(): List<ObjectType> = iterate(this,
ObjectType::getParent).toList()
    fun getParent(): ObjectType? = parent?.let { ObjectTypes.byId(it) }
    fun bind(attr: Attribute) = ObjectsTypesAttributes.insert { it[typeId] =
this@ObjectType.typeId; it[attrId] = attr.attrId}
    fun unbind(attr: Attribute) = ObjectsTypesAttributes.deleteWhere {
ObjectsTypesAttributes.typeId eq typeId and( ObjectsTypesAttributes.attrId eq
attr.attrId ) }
}

object ObjectsTypesAttributes: Table() {
    val typeId = char("type_id", 36).references(ObjectTypes.typeId, onDelete
= ReferenceOption.CASCADE)
    val attrId = char("attr_id", 36).references(Attributes.attrId, onDelete =
ReferenceOption.CASCADE)
}

object Objects: Table() {
    val objId = char("obj_id", 36).uniqueIndex().clientDefault {
getUUID("OBJ") }
    val parentId = char("parent", 36).references(objId, onDelete =
ReferenceOption.CASCADE).nullable()
    val typeId = char("type_id", 36).references(ObjectTypes.typeId, onDelete
= ReferenceOption.CASCADE)
    val name = varchar("name", 200)

    fun all() : List<Obj> = selectAll().map { Obj(it) }
    fun byId(obj: String) : Obj? = select { objId eq obj }.map { Obj(it)
}.firstOrNull()
    fun byParent(obj: String) : List<Obj> = select { parentId eq obj }.map {
Obj(it) }
    fun delete(obj: String): Int {
        return deleteWhere { objId eq obj }
            .also { References.deleteWhere { (References.objId eq obj) or
(References.ref eq obj) } }
            .also { byParent(obj).forEach { delete(it.objId) } }
    }
}
```

```
data class Obj(val objId: String = "", val parentId: String?, val typeId:
String, val name: String) {
    constructor(value: ResultRow) : this(value[Objects.objId],
value[Objects.parentId], value[Objects.typeId], value[Objects.name])
    constructor(value: InsertStatement<Number>) : this(value[Objects.objId],
value[Objects.parentId], value[Objects.typeId], value[Objects.name])
    constructor(value: Obj) : this( Objects.insert { it[parentId] =
value.parentId; it[typeId] = value.typeId; it[name] = value.name } )

    fun update(): Obj{
        Objects.update({ Objects.objId eq objId }) {
            it[parentId] = this@Obj.parentId; it[typeId] = this@Obj.typeId;
it[name] = this@Obj.name
        }
        return this
    }

    fun setParam(attrId: String, value: String?) = setParam(attrId, if(value
== null) Collections.emptyList() else listOf(value))

    fun setParam(attrId: String, value: List<String>): Int {
        return Attributes.byId(attrId)?.run {
            if(single && value.size > 1) throw
IllegalArgumentException("Attribute $name is single")
            else Params.update(this, objId, value)
        } ?: 0
    }

    fun getParams(): Map<String, Any?> {
        return ObjectTypes.byId(typeId)?.run {
            HashMap<String, Any?>().apply {
                putAll(getAttributes().associate { it.attrId to
getParam(it.attrId) })
            }
        } ?: Collections.emptyMap()
    }

    fun getParam(attrId: String): Any? {
        return Attributes.byId(attrId)?.run {
            (if (type == REFERENCE) References.find(attrId, objId) else
Params.find(attrId, objId))
                .mapNotNull {
                    when (type) {
                        TEXT -> it[Params.value]
                        LIST -> mapOf(Pair(it[Params.list],
it[Params.value]))
                        BLOB -> it[Params.blob]?.run {
String(Base64.getEncoder().encode(bytes))}
                        REFERENCE -> Objects.byId(it[References.ref])
                    }
                }.run { if(single) firstOrNull() else if (type == LIST)
this.associate { (it as Map<*, *>).entries.first().toPair() } else this }
        }
    }

    fun getType(): ObjectType = ObjectTypes.byId(typeId) ?: throw
IllegalStateException("Wrong object type")
}

object Params: Table() {
    val attrId = char("attr_id", 36).references(Attributes.attrId, onDelete =
ReferenceOption.CASCADE)
    val objId = char("obj_id", 36).references(Objects.objId, onDelete =
ReferenceOption.CASCADE)
```

```
    val value = varchar("value", 4000).nullable()
    val list = char("list", 36).nullable()
    val blob = blob("blob").nullable()

    fun find(attr: String, obj: String) = this.select { attrId eq attr and
(objId eq obj) }
    fun delete(attr: String, obj: String) = deleteWhere { attrId eq attr and
(objId eq obj) }
    fun update(attr: Attribute, obj: String, data: List<String>): Int {
        val listValues = if(attr.type == LIST) ListValues.byId(attr.listGroup
?: throw IllegalStateException("Attribute ${attr.attrId} is not have
listGroup")) else Collections.emptyMap()
        if(attr.type == REFERENCE) return References.update(attr.attrId, obj,
data)
        return delete(attr.attrId, obj) + Params.batchInsert(data) {
            this[attrId] = attr.attrId
            this[objId] = obj
            when(attr.type){
                TEXT -> this[value]=it
                LIST -> if(listValues.keys.contains(it)) {this[list]=it;
this[value] = listValues[it] } else return@batchInsert
                BLOB ->
this[blob]=ExposedBlob(Base64.getDecoder().decode(it))
                else -> null
            }
        }.size
    }
}

object References: Table() {
    val attrId = char("attr_id", 36).references(Attributes.attrId, onDelete =
ReferenceOption.CASCADE)
    val objId = char("obj_id", 36).references(Objects.objId, onDelete =
ReferenceOption.CASCADE)
    val ref = char("ref", 36).references(Objects.objId, onDelete =
ReferenceOption.CASCADE)
    val order = integer("order").default(0)

    fun find(attrId: String, objId: String) = select { References.attrId eq
attrId and ( References.objId eq objId) }.orderBy(order)
    fun delete(attrId: String, objId: String) = deleteWhere {
References.attrId eq attrId and( References.objId eq objId) }
    fun update(attrId: String, objId: String, data: List<String>): Int {
        delete(attrId, objId)
        var i = 0
        return References.batchInsert(data) { this[References.attrId] =
attrId; this[References.objId] = objId; this[ref] = it; this[order] = i++
}.size
    }
}
```

# ДОДАТОК Е

## Gradle конфігурація веб-сереру

```
val kotlinVersion: String by project
val logbackVersion: String by project
val ktorVersion = "1.2"

group = "com.example"
version = "1.0-SNAPSHOT"

plugins {
    kotlin("jvm") version "1.8.0"
    id("io.ktor.plugin") version "2.2.4"
    id("com.github.johnrengelman.shadow") version "7.0.0"
}

application {
    mainClass.set("com.example.ServerKt")
}

repositories {
    mavenCentral()
    maven { url = uri("https://maven.pkg.jetbrains.space/public/p/kotlinx-
html/maven") }
}

dependencies {
    implementation("io.ktor:ktor-server-core-jvm")
    implementation("io.ktor:ktor-server-netty-jvm")
    implementation("ch.qos.logback:logback-classic:$logbackVersion")
    implementation("io.ktor:ktor-server-content-negotiation:2.2.4")
    implementation("io.ktor:ktor-gson:latest.release")
    implementation("io.ktor:ktor-serialization-gson:latest.release")
    implementation("org.jetbrains.exposed:exposed-core:0.35.2")
    implementation("org.jetbrains.exposed:exposed-dao:0.35.2")
    implementation("org.jetbrains.exposed:exposed-jdbc:0.35.2")
    implementation("com.squareup.sqldelight:sqlite-driver:1.5.0")
}

buildscript {
    repositories {
        mavenCentral()
    }
}

tasks.shadowJar {
    archiveFileName.set(rootProject.name + ".jar")
}
```

# ДОДАТОК Ж

## Вихідний код metaModel.js

```
function request(url, method="GET", body) {
    return new Promise((resolve, reject) => {
        let xhr = new XMLHttpRequest();
        xhr.open(method, url);
        if(body) xhr.setRequestHeader('Content-Type', 'application/json');
        xhr.onload = () => {
            if (xhr.status >= 200 && xhr.status < 300) {
                try { resolve(JSON.parse(xhr.response));
                } catch (e){ resolve(xhr.response); }
            } else reject(xhr.statusText);
        };
        xhr.onerror = () => reject(xhr.statusText);
        xhr.send(body);
    });
}
const attrTypes = {TEXT:"TEXT", LIST:"LIST", BLOB:"BLOB",
REFERENCE:"REFERENCE"};
const Attr = {
    lst:()=> request(`/api/attr`),
    get:(attr)=> request(`/api/attr/${attr}`),
    val:(attr)=> request(`/api/attr/${attr}/values`),
    crt:(name, type=attrTypes.TEXT, single=true, listGroup)=>
request(`/api/attr`, "POST", JSON.stringify({type:type, listGroup:listGroup,
name:name, single:single})),
    upd:(attr)=> request(`/api/attr`, "PUT", JSON.stringify(attr)),
    del:(attr)=> request(`/api/attr/${attr}`, "DELETE")
}
const List = {
    lst:()=> request(`/api/list`),
    get:(list)=> request(`/api/list/${list}`),
    crt:(list)=> request(`/api/list`, "POST", JSON.stringify(list)),
    upd:(list, content)=> request(`/api/list/${list}`, "PUT",
JSON.stringify(content)),
    del:(list)=> request(`/api/list/${list}`, "DELETE")
}
const Type = {
    lst:()=> request(`/api/type`),
    get:(type)=> request(`/api/type/${type}`),
    atr:(type)=> request(`/api/type/${type}/attrs`),
    crt:(name, parent)=> request(`/api/type`, "POST",
JSON.stringify({name:name, parent:parent})),
    upd:(type)=> request(`/api/type`, "PUT", JSON.stringify(type)),
    bind:(type, attr)=> request(`/api/type/${type}/bind/${attr}`, "PUT"),
    unbind:(type, attr)=> request(`/api/type/${type}/unbind/${attr}`, "PUT"),
    del:(type)=> request(`/api/type/${type}`, "DELETE")
}
const Obj = {
    lst:()=> request(`/api/obj`),
    get:(obj) => request(`/api/obj/${obj}`),
    getParams:(obj) => request(`/api/obj/${obj}/params`),
    getParam:(obj, attr)=> request(`/api/obj/${obj}/param/${attr}`),
    setParam:(obj, attr, val)=> request(`/api/obj/${obj}/param/${attr}`,
"POST", typeof val === "string" ? JSON.stringify([val]) : (val === null ?
'[]' : JSON.stringify(val))),
    crt:(name, typeId, parent)=> request(`/api/obj`, "POST",
JSON.stringify({name:name, typeId:typeId, parent:parent})),
    upd:(obj)=> request(`/api/obj`, "PUT", JSON.stringify(obj)),
    del:(obj)=> request(`/api/obj/${obj}`, "DELETE")
}
```

# ДОДАТОК И

## Вихідний код load.js

```
var drawInterval = null;

var threadController;
var threadCount = 50;
var threads = new Array(threadCount).fill(true);

var lifetimes = [];
var countResponces = 0, old_1, old_2;

function updateThreadCount(count){
    for (; threadCount < count; threadCount++) threads[threadCount] = true;
    threadCount = count;
}

function handleThreads(objs){
    for(let i = 0; i < threadCount; i++) if(threads[i]) { threads[i] = false;
initThread(objs, ()=> threads[i] = true);}
}

function startLoadTesting(button){
    if(drawInterval === null){
        button.innerHTML = "Stop strstesting";
        Obj.lst().then(s=>{
            var objects = [];
            s.forEach(i=>objects.push(i.objId));
            threadController = setInterval(handleThreads, 10, objects);
        });
        clear();
        drawInterval = setInterval(draw, 1000);
    } else {
        button.innerHTML = "Start strstesting";
        clearInterval(threadController);
        clearInterval(drawInterval);
        drawInterval = null;
    }
}

function startUserTest(button){
    document.chart.data.labels = [];
    document.chart.data.datasets[0].data = [];
    document.chart.data.datasets[1].data = [];
    function drw(t){
        document.chart.data.labels.push(currentTime());
        document.chart.data.datasets[0].data.push(t);
        document.chart.data.datasets[1].data.push(1);
        document.chart.update();
    }

    if(drawInterval == null){
        button.innerHTML = "Stop user like testing";
        let obj = null;
        drawInterval = setInterval(()=>{
            var time = new Date();
            if(obj === null) Obj.crt("test", "46f91022-f598-4ae1-8d24-
bf48c4a0c9e0").then(s=>{obj=s; drw(new Date().getTime()-time.getTime());});
            else Obj.del(obj.objId).then(()=>{obj=null; drw(new
Date().getTime()-time.getTime());});
        }, 2000);
    } else {
```

```
        button.innerHTML = "Start user like testing";
        clearInterval(drawInterval);
        drawInterval = null;
    }
}

function getRandomArbitrary(min, max) {
    return (Math.random() * (max - min) + min).toFixed();
}

function initThread(objs, end){
    let start = new Date().getTime();
    Obj.getParams(objs[getRandomArbitrary(0, objs.length -
1)]).then(s=>{compl(start);end();});
}

function compl(start){
    lifetimes.push(new Date().getTime() - start);
    countResponces++;
}

function zeroPad(num, places){
    return String(num).padStart(places, '0');
}

function currentTime(){
    let date = new Date();
    return zeroPad(date.getHours(), 2) +':'+ zeroPad(date.getMinutes(), 2)
+':'+ zeroPad(date.getSeconds(), 2);
}

function draw(){
    let time = currentTime();
    let avglifetime = (lifetimes.reduce((a, b) => a + b, 0) /
lifetimes.length) || 0;
    let avgCountResponces = (countResponces + (old_1 || 0) + (old_2 || 0)) /
(1 + !!old_1 + !!old_2);
    document.chart.data.labels.push(time);
    document.chart.data.datasets[0].data.push(avglifetime);
    document.chart.data.datasets[1].data.push(avgCountResponces);
    console.log(time +' '+ avglifetime.toFixed(2) +' '+ avgCountResponces)
    lifetimes = [];
    old_2 = old_1;
    old_1 = countResponces;
    countResponces = 0;
    document.chart.update();
}

function clear(){
    lifetimes = [];
    countResponces = 0;
    old_1 = 0;
    old_2 = 0;
    document.chart.data.labels = [];
    document.chart.data.datasets[0].data = [];
    document.chart.data.datasets[1].data = [];
    document.chart.update();
}

function init(){
    document.chart = new Chart("myChart", {
      type: "line",
      data: {
        labels: [],
```

```
        datasets: [{
          data: [],
          borderWidth: 2,
          borderColor: "red",
          fill: false,
          label: 'Avarage respond time (ms)',
          yAxisID: 'left-y-axis'
        },{
          data: [],
          borderWidth: 2,
          borderColor: "blue",
          fill: false,
          label: 'Responds per second',
          yAxisID: 'right-y-axis'
        }]
      },
      options: {
        scales: {
            yAxes: [
                {id: 'left-y-axis',type: 'linear',position: 'right'},
                {id: 'right-y-axis',type: 'linear',position: 'left'}
            ]
        }
      }
    });
}
init();
```

# ДОДАТОК К

## Вихідний код головної сторінки

```
<html>
    <head><title>Main page</title></head>
    <body style="margin:0">
        <canvas id="myChart" style="width:100%; height:70vh"></canvas>
        <input type="number" min=0 max=100 placeholder="Count threads"
onchange="updateThreadCount(+this.value)">
        <button onclick="startLoadTesting(this)">Start strstesting</button>
        <button onclick="startUserTest(this)">Start user like
testing</button>
        <script src="js/metaModel.js"></script>
        <script
src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.9.4/Chart.js">
</script>
        <script src="js/load.js"></script>
    </body>
</html>
```

# ДОДАТОК Л

## Конфігурація середовища Postman


```
{"variables": [],
"info": {
"name": "Metamodel",
"_postman_id": "41cc4fd9-184d-30a4-b339-ebba746b42da",
"description": "",
"schema":
"https://schema.getpostman.com/json/collection/v2.0.0/collection.json"},
"item": [{
"name": "Attributes",
"description": "",
"item": [{
"name": "Show Attributes",
"request": {
"url": "http://{{url}}/attr",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []
},{"name": "Get Attribute",
"request": {
"url": "http://{{url}}/attr/a6fd43b9-8e17-4b7c-9d59-bfd8d1f579c7",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []
},{"name": "Get Attribute values",
"request": {
"url": "http://{{url}}/attr/8be02c32-cb03-48da-8672-f5bb95ab26a6/values",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []
},{"name": "Create Attribute",
"request": {
"url": "http://{{url}}/attr",
"method": "POST",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n\t\"type\":\"REFERENCE\",\n\t\"name\": \"refer to
root\",\n\t\"single\": true\n}\n"},
"description": ""},
"response": []
},{"name": "Update Attribute",
"request": {
"url": "http://{{url}}/attr",
"method": "PUT","header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
```

```
"raw": "{\n    \"attrId\": \"8be02c32-cb03-48da-8672-f5bb95ab26a6\",\n
\"type\": \"LIST\",\n    \"listGroup\": \"262c6beb-dd20-461e-b968-
d607b5cefff3\",\n    \"name\": \"list attribute\",\n    \"single\":
false\n}"},
"description": ""},
"response": []},{
"name": "Delete Attribute",
"request": {
"url": "http://{{url}}/attr/71001e1f-60da-4cb7-8b37-c607a4033f38",
"method": "DELETE",
"header": [],"body": {},
"description": ""},
"response": []}]},{
"name": "Lists",
"description": "",
"item": [{
"name": "Show Lists",
"request": {
"url": "http://{{url}}/list",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get List",
"request": {
"url": "http://1{{url}}/list/262c6beb-dd20-461e-b968-d607b5cefff3",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Create List",
"request": {
"url": "http://{{url}}/list",
"method": "POST",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw":
"[\n\t\"val1\",\n\t\"val2\",\n\t\"val3\",\n\t\"val4\",\n\t\"val5\",\n\t\"val6
\"\n]"},
"description": ""},
"response": []},{
"name": "Update List",
"request": {
"url": "http://{{url}}/list/57f3af95-0a36-430d-a3c4-169108ebf0d5",
"method": "PUT",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n    \"42861453-5416-4bce-b6f9-86d0dd308bf1\": \"val6\",\n
\"ed971039-82a4-4b61-abe3-8d4345b3bbe9\": \"val2\",\n    \"2ba3d37e-41d1-
4de9-849f-a42e848ce0fb\": \"val5\",\n    \"10217561-8164-4df0-9cfc-
2b4033e56586\": \"val4\",\n    \"c3e9c673-cfc7-4288-94b2-400b3de14662\":
\"val3\",\n    \"new1\": \"val55\"\n}"},
"description": ""},
"response": []},{
```

```
"name": "Delete List",
"request": {
"url": "http://{{url}}/list/71001e1f-60da-4cb7-8b37-c607a4033f38",
"method": "DELETE",
"header": [],
"body": {},
"description": ""},
"response": []}]},{
"name": "Types",
"description": "",
"item": [{
"name": "Show Types",
"request": {
"url": "http://{{url}}/type",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get Type",
"request": {
"url": "http://{{url}}/type/38da4608-2613-4e09-951d-ffeb85e4f1af",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get Type Attributes",
"request": {
"url": "http://{{url}}/type/29d856b9-80cb-456e-9b77-c9d2ba288f99/attrs",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Create Type",
"request": {
"url": "http://{{url}}/type",
"method": "POST",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n\t\"parent\": \"38da4608-2613-4e09-951d-
ffeb85e4f1af\",\n\t\"name\": \"type2\"\n}\n"},
"description": ""},
"response": []},{
"name": "Update Type",
"request": {
"url": "http://{{url}}/type",
"method": "PUT",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n    \"typeId\": \"58ad8299-7cae-4251-a54a-623e69dde849\",\n
\"parent\": \"38da4608-2613-4e09-951d-ffeb85e4f1af\",\n    \"name\":
\"type2\"\n}"},
"description": ""},
"response": []},{
```

```
"name": "Bind Attribute to Type",
"request": {
"url": "http://{{url}}/type/29d856b9-80cb-456e-9b77-
c9d2ba288f99/bind/a6fd43b9-8e17-4b7c-9d59-bfd8d1f579c7",
"method": "PUT",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": ""},
"description": ""},
"response": []},{
"name": "Unbind Attribute from Type",
"request": {
"url": "http://{{url}}/type/29d856b9-80cb-456e-9b77-
c9d2ba288f99/unbind/8be02c32-cb03-48da-8672-f5bb95ab26a6",
"method": "PUT",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": ""},
"description": ""},
"response": []},{
"name": "Delete Type",
"request": {
"url": "http://{{url}}/type/58ad8299-7cae-4251-a54a-623e69dde849",
"method": "DELETE",
"header": [],
"body": {},
"description": ""},
"response": []}]},{
"name": "Objects",
"description": "",
"item": [{
"name": "Show Objects",
"request": {
"url": "http://{{url}}/obj",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get Object",
"request": {
"url": "http://{{url}}/obj/b7335577-11f6-4bd6-96a3-9e70dad287ec",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get Object params",
"request": {
"url": "http://{{url}}/obj/094383c4-e6f2-4fa6-a23a-22a9514b4447/params",
"method": "GET",
"header": [],
"body": {},
"description": ""},
"response": []},{
"name": "Get Object param",
```

```
"request": {
"url": "http://{{url}}/obj/b7335577-11f6-4bd6-96a3-
9e70dad287ec/param/5f47ef39-de7f-4628-a29b-6d7362b99f59",
"method": "GET",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": ""},
"description": ""},
"response": []},{
"name": "Set Object param",
"request": {
"url": "http://{{url}}/obj/094383c4-e6f2-4fa6-a23a-
22a9514b4447/param/a6fd43b9-8e17-4b7c-9d59-bfd8d1f579c7",
"method": "POST",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "b7335577-11f6-4bd6-96a3-9e70dad287ec"},
"description": ""},
"response": []},{
"name": "Create Object",
"request": {
"url": "http://{{url}}/obj",
"method": "POST","header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n\t\"parentId\":\"b7335577-11f6-4bd6-96a3-
9e70dad287ec\",\n\t\"typeId\":\"29d856b9-80cb-456e-9b77-
c9d2ba288f99\",\n\t\"name\": \"obj1\"\n}\n"},
"description": ""},
"response": []},{
"name": "Update Object",
"request": {
"url": "http://{{url}}/type",
"method": "PUT",
"header": [{
"key": "Content-Type",
"value": "application/json",
"description": ""}],
"body": {
"mode": "raw",
"raw": "{\n    \"objId\": \"b7335577-11f6-4bd6-96a3-9e70dad287ec\",\n
\"parentId\": \"38da4608-2613-4e09-951d-ffeb85e4f1af\",\n    \"typeId\":
\"58ad8299-7cae-4251-a54a-623e69dde849\",\n    \"name\": \"Obj2\"\n}"},
"description": ""
},"response": []},{
"name": "Delete Type",
"request": {
"url": "http://{{url}}/obj/58ad8299-7cae-4251-a54a-623e69dde849",
"method": "DELETE",
"header": [],
"body": {},
"description": ""},
"response": []}]}]}
```

# ДОДАТОК М

Таблиці результатів тестування

Таблиця М.1 – Дані першого сценарію тестування

| Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps | Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps |
|---|---|---|---|---|---|
| 1:10:19 | 801,08 | 12,00 | 1:10:44 | 3181,87 | 16,33 |
| 1:10:20 | 1594,4 | 11,00 | 1:10:45 | 3194,2 | 15,67 |
| 1:10:21 | 2626,67 | 11,33 | 1:10:46 | 3401,67 | 13,00 |
| 1:10:22 | 3655,4 | 9,00 | 1:10:47 | 3751,46 | 12,33 |
| 1:10:23 | 4624,18 | 9,33 | 1:10:48 | 4053,82 | 11,00 |
| 1:10:24 | 4863,08 | 9,33 | 1:10:49 | 4115,15 | 12,33 |
| 1:10:25 | 4966,77 | 12,00 | 1:10:50 | 4178,77 | 12,33 |
| 1:10:26 | 4746,55 | 12,00 | 1:10:51 | 4051,92 | 12,67 |
| 1:10:27 | 4103,81 | 13,33 | 1:10:52 | 3949,36 | 13,00 |
| 1:10:28 | 3782,42 | 13,00 | 1:10:53 | 3814,93 | 13,33 |
| 1:10:29 | 3720,38 | 14,67 | 1:10:54 | 3693,79 | 14,00 |
| 1:10:30 | 3550,73 | 14,33 | 1:10:55 | 3483,19 | 14,67 |
| 1:10:31 | 3313,13 | 15,67 | 1:10:56 | 3365,69 | 15,33 |
| 1:10:32 | 3262,25 | 15,67 | 1:10:57 | 3325,36 | 15,33 |
| 1:10:33 | 3267,17 | 14,67 | 1:10:58 | 3277,86 | 14,67 |
| 1:10:34 | 3407,13 | 14,33 | 1:10:59 | 3482,43 | 11,67 |
| 1:10:35 | 3397,67 | 14,00 | 1:11:00 | 4070,52 | 14,00 |
| 1:10:36 | 3449,06 | 15,33 | 1:11:01 | 3584,62 | 13,67 |
| 1:10:37 | 3165,06 | 16,33 | 1:11:02 | 3593,47 | 16,33 |
| 1:10:38 | 3095,92 | 15,67 | 1:11:03 | 2752,5 | 14,00 |
| 1:10:39 | 3288,08 | 14,67 | 1:11:04 | 3364,71 | 15,33 |
| 1:10:40 | 3376,07 | 13,67 | 1:11:05 | 3358 | 14,67 |
| 1:10:41 | 3665,08 | 13,33 | 1:11:06 | 3408,77 | 14,33 |
| 1:10:42 | 3614,94 | 14,67 | 1:11:07 | 3550,31 | 13,00 |
| 1:10:43 | 3329,47 | 15,33 | 1:11:08 | 3776,57 | 13,33 |

Таблиця М.2 – Дані другого сценарію тестування

| Час заміру | Середній час очікування, ms | Кількість запитів, rps |
|---|---|---|
| 1:35:50 | 802,79 | 14,00 |
| 1:35:51 | 1685,86 | 14,00 |
| 1:35:52 | 2690,5 | 14,67 |
| 1:35:53 | 3377,29 | 14,67 |
| 1:35:54 | 3387,5 | 14,67 |
| 1:35:55 | 3306,88 | 15,00 |
| 1:35:56 | 3352,07 | 15,33 |
| 1:35:57 | 3489,2 | 15,67 |
| 1:35:58 | 3161,47 | 15,67 |
| 1:35:59 | 3138 | 15,33 |
| 1:36:00 | 3088,53 | 15,33 |
| 1:36:01 | 3269,13 | 15,00 |
| 1:36:02 | 3211,24 | 16,00 |
| 1:36:03 | 3033,73 | 16,00 |
| 1:36:04 | 3361,29 | 15,33 |
| 1:36:05 | 3419,36 | 14,33 |
| 1:36:06 | 3488,93 | 14,00 |
| 1:36:07 | 3452,19 | 14,67 |
| 1:36:08 | 3360,65 | 15,67 |
| 1:36:09 | 3194,27 | 16,00 |
| 1:36:10 | 3110,07 | 15,67 |
| 1:36:11 | 3344,4 | 15,00 |
| 1:36:12 | 3175,57 | 14,67 |
| 1:36:13 | 3321,4 | 14,67 |
| 1:36:14 | 3666,69 | 14,00 |

| Час заміру | Середній час очікування, ms | Кількість запитів, rps |
|---|---|---|
| 1:36:15 | 3629,83 | 13,33 |
| 1:36:16 | 3953,29 | 10,67 |
| 1:36:17 | 4326,18 | 10,00 |
| 1:36:18 | 4546,83 | 10,00 |
| 1:36:19 | 4768,25 | 10,33 |
| 1:36:20 | 5009,6 | 10,00 |
| 1:36:21 | 4675,36 | 10,67 |
| 1:36:22 | 4457,82 | 11,67 |
| 1:36:23 | 4839,33 | 12,33 |
| 1:36:24 | 4226,83 | 11,67 |
| 1:36:25 | 4146,23 | 12,33 |
| 1:36:26 | 4139,67 | 12,33 |
| 1:36:27 | 3987,79 | 13,00 |
| 1:36:28 | 3788,43 | 13,33 |
| 1:36:29 | 3692,23 | 13,67 |
| 1:36:30 | 3474,27 | 14,00 |
| 1:36:31 | 3775,23 | 13,67 |
| 1:36:32 | 3451,69 | 14,67 |
| 1:36:33 | 3216 | 15,33 |
| 1:36:34 | 3374,6 | 16,00 |
| 1:36:35 | 3254,64 | 15,33 |
| 1:36:36 | 3340,87 | 14,67 |
| 1:36:37 | 3331,25 | 13,67 |
| 1:36:38 | 3461,35 | 14,67 |
| 1:36:39 | 3365,75 | 13,67 |

Таблиця М.3 – Дані третього сценарію, сесії з навантаженням 50 паралельних запитів

| Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps |
|---|---|---|
| 16:52:36 | 980,1 | 10 |
| 16:52:37 | 1641,25 | 11 |
| 16:52:38 | 2650,27 | 12,33 |
| 16:52:39 | 3723,85 | 13,33 |
| 16:52:40 | 3648,38 | 13,66 |
| 16:52:41 | 3649,77 | 13 |
| 16:52:42 | 3699,64 | 13,33 |
| 16:52:43 | 3975 | 13 |
| 16:52:44 | 3764,13 | 13,66 |
| 16:52:45 | 3798,33 | 13 |
| 16:52:46 | 3698,19 | 14,33 |
| 16:52:47 | 3603,08 | 13,33 |
| 16:52:48 | 3724,93 | 14,33 |
| 16:52:49 | 3329 | 14 |
| 16:52:50 | 2941,8 | 15 |
| 16:52:51 | 2929,13 | 15 |
| 16:52:52 | 2181,67 | 15 |
| 16:52:53 | 2238,5 | 12,66 |
| 16:52:54 | 2044,73 | 11,33 |
| 16:52:55 | 2304,27 | 10 |
| 16:52:56 | 2291,53 | 12,33 |
| 16:52:57 | 3443,67 | 12,66 |
| 16:52:58 | 1514,88 | 14,33 |
| 16:52:59 | 1271,29 | 14 |
| 16:53:00 | 450,27 | 13,66 |
| 16:53:01 | 411,73 | 12 |
| 16:53:02 | 593,5 | 8 |
| 16:53:03 | 3408 | 6,66 |
| 16:53:04 | 4698,89 | 9,33 |
| 16:53:05 | 1242,79 | 13,33 |
| 16:53:06 | 522 | 13,33 |
| 16:53:07 | 1369,18 | 10,66 |
| 16:53:08 | 1277,67 | 7 |
| 16:53:09 | 3946,05 | 11,66 |
| 16:53:10 | 1478 | 8,33 |
| 16:53:11 | 1669,9 | 10,66 |
| 16:53:12 | 2277,8 | 7 |
| 16:53:13 | 6526,2 | 13,33 |
| 16:53:14 | 0 | 10 |
| 16:53:15 | 5794,37 | 19,5 |
| 16:53:16 | 1064,64 | 15 |
| 16:53:17 | 2686,17 | 14 |

Таблиця М.4 – Дані третього сценарію, сесії з незначним навантаженням з впливом паралельної сесії з 50 паралельними запитами

| Час заміру | Час очікування, ms |
|---|---|
| 2:51:32 | 201 |
| 2:51:34 | 209 |
| 2:51:36 | 210 |
| 2:51:38 | 204 |
| 2:51:40 | 198 |
| 2:51:42 | 193 |
| 2:51:44 | 200 |
| 2:51:46 | 227 |
| 2:51:48 | 234 |
| 2:51:50 | 206 |
| 2:51:52 | 182 |
| 2:51:54 | 191 |
| 2:51:56 | 198 |
| 2:51:58 | 196 |
| 2:51:59 | 175 |
| 2:52:02 | 211 |
| 2:52:03 | 179 |
| 2:52:06 | 197 |
| 2:52:08 | 188 |
| 2:52:10 | 188 |
| 2:52:12 | 238 |
| 2:52:14 | 191 |
| 2:52:16 | 191 |
| 2:52:17 | 178 |
| *2:52:23* | *3531* |
| *2:52:27* | *3796* |
| *2:52:29* | *3541* |
| *2:52:31* | *3428* |
| *2:52:32* | *2962* |
| *2:52:34* | *2488* |
| *2:52:35* | *1920* |
| *2:52:37* | *1859* |
| *2:52:39* | *1437* |
| *2:52:41* | *1297* |
| *2:52:42* | *602* |
| 2:52:44 | 408 |
| 2:52:46 | 259 |
| 2:52:48 | 275 |
| 2:52:50 | 279 |
| 2:52:52 | 415 |
| 2:52:54 | 238 |
| 2:52:56 | 304 |
| 2:52:58 | 901 |
| 2:53:00 | 286 |
| 2:53:02 | 265 |
| 2:53:04 | 435 |
| 2:53:06 | 393 |
| 2:53:08 | 252 |
| 2:53:10 | 471 |
| 2:53:12 | 345 |
| 2:53:14 | 518 |
| 2:53:16 | 565 |
| 2:53:18 | 712 |
| 2:53:20 | 558 |
| 2:53:21 | 174 |
| 2:53:24 | 215 |
| 2:53:25 | 175 |
| 2:53:27 | 180 |
| 2:53:29 | 175 |
| 2:53:32 | 191 |
| 2:53:33 | 176 |
| 2:53:36 | 194 |
| 2:53:37 | 166 |
| 2:53:39 | 185 |
| 2:53:41 | 165 |
| 2:53:44 | 207 |
| 2:53:45 | 179 |

Таблиця М.5 – Дані третього сценарію, сесії з навантаженням 20 паралельних запитів

| Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps |
|---|---|---|
| 16:58:45 | 667,8 | 10 |
| 16:58:46 | 1449,71 | 12 |
| 16:58:47 | 1397,08 | 12 |
| 16:58:48 | 1589,36 | 13,33 |
| 16:58:49 | 1520,85 | 13 |
| 16:58:50 | 1571,5 | 13 |
| 16:58:51 | 1522,21 | 13 |
| 16:58:52 | 1464,08 | 12,66 |
| 16:58:53 | 1495 | 13,66 |
| 16:58:54 | 1465,86 | 13,66 |
| 16:58:55 | 1365,13 | 14,66 |
| 16:58:56 | 1375,93 | 14,33 |
| 16:58:57 | 1391,62 | 14 |
| 16:58:58 | 1341,06 | 14,33 |
| 16:58:59 | 1266,92 | 13,66 |
| 16:59:00 | 1523 | 13,33 |
| 16:59:01 | 1547,47 | 13 |
| 16:59:02 | 1320,46 | 13,33 |
| 16:59:03 | 1334,33 | 14,33 |
| 16:59:04 | 1261,79 | 14 |
| 16:59:05 | 1457,5 | 13 |
| 16:59:06 | 1639,38 | 12,33 |
| 16:59:07 | 2043,14 | 12,33 |
| 16:59:08 | 1291,62 | 13,33 |

| Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps |
|---|---|---|
| 16:59:09 | 1155,43 | 13,66 |
| 16:59:10 | 1107,54 | 13,33 |
| 16:59:11 | 1001,57 | 13,66 |
| 16:59:12 | 725,06 | 14,33 |
| 16:59:13 | 630,13 | 15 |
| 16:59:14 | 408,8 | 15,33 |
| 16:59:15 | 395,92 | 14 |
| 16:59:16 | 411,38 | 13,33 |
| 16:59:17 | 368,08 | 12,33 |
| 16:59:18 | 914,73 | 13,33 |
| 16:59:19 | 360,31 | 13,33 |
| 16:59:20 | 389,38 | 13,66 |
| 16:59:21 | 350,08 | 13 |
| 16:59:22 | 424,62 | 13 |
| 16:59:23 | 345,38 | 13 |
| 16:59:24 | 374,36 | 13,33 |
| 16:59:25 | 347,08 | 13,33 |
| 16:59:26 | 866,79 | 13,66 |
| 16:59:27 | 335,29 | 13,66 |
| 16:59:28 | 356,13 | 14,33 |
| 16:59:29 | 363,83 | 13,66 |
| 16:59:30 | 364,57 | 13,66 |
| 16:59:31 | 707,36 | 13,33 |
| 16:59:32 | 1720 | 13,33 |

Таблиця М.6 – Дані третього сценарію, сесії з незначним навантаженням з впливом паралельної сесії з 20 паралельними запитами

| Час заміру | Час очікування, ms |
|---|---|
| 2:57:49 | 223 |
| 2:57:51 | 182 |
| 2:57:53 | 181 |
| 2:57:55 | 171 |
| 2:57:57 | 178 |
| 2:57:59 | 192 |
| 2:58:01 | 197 |
| 2:58:03 | 197 |
| 2:58:05 | 194 |
| 2:58:07 | 182 |
| 2:58:09 | 189 |
| 2:58:11 | 320 |
| 2:58:13 | 190 |
| 2:58:15 | 187 |
| 2:58:17 | 190 |
| 2:58:19 | 195 |
| 2:58:22 | 344 |
| 2:58:24 | 435 |
| 2:58:26 | 186 |
| *2:58:29* | *1621* |
| *2:58:31* | *1491* |
| *2:58:33* | *1611* |
| *2:58:35* | *1526* |
| *2:58:36* | *1229* |
| *2:58:38* | *1305* |
| *2:58:40* | *1266* |
| *2:58:43* | *1433* |
| *2:58:44* | *1270* |

| Час заміру | Час очікування, ms |
|---|---|
| *2:58:47* | *1397* |
| *2:58:49* | *1357* |
| *2:58:50* | *1173* |
| *2:58:52* | *1027* |
| 2:58:54 | 663 |
| 2:58:56 | 670 |
| 2:58:57 | 317 |
| 2:58:59 | 196 |
| 2:59:01 | 240 |
| 2:59:03 | 323 |
| 2:59:06 | 568 |
| 2:59:07 | 338 |
| 2:59:09 | 262 |
| 2:59:12 | 233 |
| 2:59:14 | 278 |
| 2:59:16 | 591 |
| 2:59:18 | 306 |
| 2:59:20 | 252 |
| 2:59:21 | 175 |
| 2:59:23 | 192 |
| 2:59:25 | 175 |
| 2:59:27 | 164 |
| 2:59:29 | 178 |
| 2:59:31 | 165 |
| 2:59:33 | 182 |
| 2:59:35 | 174 |
| 2:59:38 | 568 |
| 2:59:39 | 173 |

Таблиця М.7 – Дані третього сценарію, сесії з навантаженням 100 паралельних запитів

| Час заміру | Середній час очікування, ms | Кількість оброблених запитів, rps |
|---|---|---|
| 17:04:18 | 419 | 1 |
| 17:04:19 | 847,83 | 6,5 |
| 17:04:20 | 2416,53 | 10 |
| 17:04:21 | 2988,38 | 14 |
| 17:04:22 | 4147,82 | 13,66 |
| 17:04:23 | 4985,21 | 12,66 |
| 17:04:24 | 5743,56 | 11,33 |
| 17:04:25 | 6996,35 | 13,33 |
| 17:04:26 | 7518,87 | 13,66 |
| 17:04:27 | 7062,73 | 15,66 |
| 17:04:28 | 7341,89 | 13 |
| 17:04:29 | 7396,47 | 14,33 |
| 17:04:30 | 7098,73 | 14,33 |
| 17:04:31 | 6904,73 | 16,33 |
| 17:04:32 | 7027,08 | 14 |
| 17:04:33 | 7481 | 12,33 |
| 17:04:34 | 7339,77 | 11,66 |
| 17:04:35 | 7516,17 | 11,66 |
| 17:04:36 | 7449,64 | 12 |
| 17:04:37 | 7129,13 | 10,33 |
| 17:04:38 | 7524,28 | 12,33 |
| 17:04:39 | 7857,42 | 12,66 |
| 17:04:40 | 9417,77 | 14,33 |
| 17:04:41 | 8450,25 | 12,33 |
| 17:04:42 | 7813,38 | 12,66 |
| 17:04:43 | 7397,83 | 12,33 |
| 17:04:44 | 7808,86 | 13 |
| 17:04:45 | 7711 | 13 |
| 17:04:46 | 7581,5 | 13,66 |
| 17:04:47 | 6949 | 13,66 |
| 17:04:48 | 6503,12 | 15 |
| 17:04:49 | 6525,07 | 15,33 |
| 17:04:50 | 6042,5 | 15,33 |
| 17:04:51 | 5658,88 | 15 |
| 17:04:52 | 5112,91 | 13,66 |
| 17:04:53 | 4767,53 | 14,66 |
| 17:04:54 | 5196 | 13,33 |
| 17:04:55 | 4094,56 | 15,66 |
| 17:04:56 | 4264 | 14,33 |
| 17:04:57 | 3813,31 | 15,66 |
| 17:04:58 | 3246,47 | 14,66 |
| 17:04:59 | 3203,8 | 17 |
| 17:05:00 | 2091,36 | 16,33 |
| 17:05:01 | 1488,72 | 17,33 |
| 17:05:02 | 1129,43 | 15,33 |
| 17:05:03 | 713,5 | 13,33 |
| 17:05:04 | 905,56 | 12,66 |
| 17:05:05 | 651 | 11,33 |
| 17:05:06 | 482,33 | 11,66 |
| 17:05:07 | 786,15 | 10,66 |
| 17:05:08 | 667,2 | 10,66 |
| 17:05:09 | 1193,55 | 11,33 |
| 17:05:10 | 664,7 | 10,33 |
| 17:05:11 | 661,83 | 11 |
| 17:05:12 | 357,83 | 9,33 |
| 17:05:13 | 2897,93 | 11 |
| 17:05:14 | 354 | 9,66 |
| 17:05:15 | 2099,86 | 12,33 |
| 17:05:16 | 1848 | 10 |
| 17:05:17 | 5915,57 | 12 |
| 17:05:18 | 343,7 | 10,66 |
| 17:05:19 | 4032,33 | 10 |
| 17:05:20 | 2700,06 | 11 |
| 17:05:21 | 449,82 | 11,33 |
| 17:05:22 | 1339 | 13,66 |
| 17:05:23 | 373,36 | 11,66 |

Таблиця М.8 – Дані третього сценарію, сесії з незначним навантаженням з впливом паралельної сесії зі 100 паралельними запитами

| Час заміру | Час очікування, ms |
|---|---|
| 3:03:29 | 313 |
| 3:03:30 | 172 |
| 3:03:32 | 182 |
| 3:03:34 | 185 |
| 3:03:36 | 164 |
| 3:03:38 | 174 |
| 3:03:40 | 180 |
| 3:03:42 | 190 |
| 3:03:44 | 185 |
| 3:03:46 | 178 |
| 3:03:48 | 185 |
| 3:03:50 | 181 |
| 3:03:52 | 243 |
| 3:03:55 | 199 |
| 3:03:57 | 189 |
| 3:03:59 | 309 |
| 3:04:01 | 332 |
| *3:04:09* | *6765* |
| *3:04:18* | *7446* |
| *3:04:19* | *7145* |
| *3:04:22* | *7656* |
| *3:04:24* | *7414* |
| *3:04:25* | *7231* |
| *3:04:27* | *7047* |
| *3:04:29* | *6698* |
| *3:04:30* | *6242* |
| *3:04:32* | *5561* |
| *3:04:34* | *5466* |
| *3:04:35* | *4588* |
| *3:04:36* | *4159* |
| *3:04:38* | *3682* |
| *3:04:39* | *3254* |
| *3:04:41* | *2833* |

| Час заміру | Час очікування, ms |
|---|---|
| *3:04:42* | *1865* |
| *3:04:43* | *1266* |
| *3:04:45* | *608* |
| *3:04:47* | *508* |
| *3:04:49* | *568* |
| 3:04:51 | 334 |
| 3:04:53 | 318 |
| 3:04:55 | 368 |
| 3:04:57 | 276 |
| 3:04:59 | 228 |
| 3:05:01 | 236 |
| 3:05:03 | 297 |
| 3:05:05 | 267 |
| 3:05:07 | 329 |
| 3:05:09 | 364 |
| 3:05:11 | 302 |
| 3:05:13 | 422 |
| 3:05:15 | 367 |
| 3:05:16 | 164 |
| 3:05:19 | 194 |
| 3:05:20 | 197 |
| 3:05:22 | 189 |
| 3:05:24 | 177 |
| 3:05:26 | 167 |
| 3:05:28 | 186 |
| 3:05:30 | 207 |
| 3:05:32 | 179 |
| 3:05:34 | 176 |
| 3:05:36 | 227 |
| 3:05:38 | 215 |
| 3:05:40 | 195 |
| 3:05:42 | 211 |
| 3:05:44 | 228 |