Розробка та аналіз продуктивності з використанням логів, трейсів та метрик серверного додатку системи оцінки психологічного стану користувача

Владислав Сусідко[1], магістр (ORCID: 0009-0002-8652-8393),
Юлія Рябчун[1], д-р філ.(ORCID: 0000-0002-8320-4038)

[1] *Київський національний університет будівництва і архітектури, м.Київ, Україна*

АНОТАЦІЯ

Робота присвячена розробці та аналізу продуктивності серверного додатку для оцінки психологічного стану користувача. Для реалізації бек-енду використано монолітну архітектуру, .NET та C#, а для API – GraphQL. Дослідження включає комплексний аналіз продуктивності системи за допомогою логів, трейсів та метрик. Зокрема, було встановлено, що час виконання запиту становить 143 мілісекунди , а навантаження на процесор – 20%. Інфраструктура була розгорнута в Microsoft Azure з використанням Terraform. Аналіз підтвердив високу ефективність, надійність та масштабованість системи.

Ключові слова: оцінка психологічного стану, продуктивність, NET, C#, GraphQL, Microsoft Azure, Terraform, монолітна архітектура.

1. ВСТУП

У наш час, де психологічне благополуччя відіграє важливу роль у визначенні якості життя, все більшу увагу привертають інструменти для спостереження та аналізу психологічного стану. У цьому контексті, представлена робота зосереджена на розробці серверного додатку системи оцінки психологічного стану користувача, який має забезпечити ефективну та надійну взаємодію з даними. Однак, лише функціональна розробка є недостатньою; для критично важливих систем, що обробляють чутливі дані та вимагають швидкої реакції, першочергове значення має її продуктивність. Саме тому, невід'ємною частиною даного дослідження є всебічний аналіз продуктивності розробленого додатку. Для досягнення цієї мети буде використано комплексний підхід, що включає збір та аналіз логів, трейсів та метрик. Цей підхід дозволить не лише виявити потенційні вузькі місця та оптимізувати роботу системи, але й забезпечити високий рівень надійності, масштабованості та швидкої реакції в умовах реального навантаження.

2. МЕТА РОБОТИ

Метою даної роботи є дослідження розробки серверного додатку системи оцінки психологічного стану користувача та аналіз його продуктивності з використанням сучасних інструментів моніторингу, таких як логи, трейси та метрики, для виявлення потенційних вузьких місць, проведення оптимізації та забезпечення високого рівня ефективності, надійності, масштабованості та швидкодії системи в умовах реального навантаження.

3. ДОСЛІДЖЕННЯ

3.1. Розробка додатку

Система оцінки психологічного стану складається з клієнтської частини, серверної частини та бази даних. Її бек-енд розробка має на меті створити надійне програмне забезпечення, що відповідає за обробку даних, отриманих від користувача, та надання результатів оцінки. Основний функціонал системи включає реєстрацію, проходження тесту та надання відповідних результатів та рекомендацій користувачу.

Процес розробки системи розпочався з вибору технологій та побудови архітектури. Було обрано монолітну архітектуру, оскільки вона є простішою для розгортання та керування на початковому етапі проєкту, а також має легший процес налагодження коду. Для реалізації бек-енду було використано C# у середовищі .NET, а для проєктування API системи обрано технологію GraphQL. Такий підхід дозволив розробити ефективну та масштабовану серверний додаток, який відповідає всім вимогам, включаючи безпеку, стабільність та ефективність.

Інфраструктурна частина додатку була розроблена з використанням хмарного провайдера Microsoft Azure, оскільки він найкраще інтегрується з .NET платформою, обраною для розробки серверної частини. Для автоматизації розгортання інфраструктури було застосовано підхід Infrastructure as Code (IaC) за допомогою Terraform, що дозволяє уникнути людських помилок та забезпечує консистентність середовищ.

3.2. Діагностування запитів за допомогою трейсингу

Трейсинг – це метод спостереження за виконанням запитів або процесів у розподілених системах, який дозволяє відстежити шлях окремої операції через усі сервіси, компоненти та залежності. На відміну від метрик чи логів, трейсинг надає деталізовану інформацію про те, як саме запит проходить через систему, які вузли він зачіпає, скільки часу витрачається на кожен етап і де виникають затримки чи помилки. Це особливо важливо для мікросервісних архітектур, де один запит може проходити через десятки чи сотні різних сервісів, і без трейсингу зрозуміти причину проблеми або повільної роботи стає майже неможливо.

На рисунку 1 зображений трейс виконання запиту до серверноъ частини додатку. Час всього запиту зображений на часовій діаграмі і складає 143 мілісекунди, що є дуже гарним результатом. Запит складається з декількох вкладений запитів, які робить серверна частина: запити до бази даних на отримання інформації та запит до серверу авторизації.

3.3. Аналіз продуктивності з використанням метрик

Метрики відображають стан та поведінку системи у певний момент часу або на протязі певного періоду. Вони є однією з ключових складових комплексного моніторингу, оскільки дозволяють кількісно оцінювати продуктивність, навантаження та загальне здоров'я інфраструктури чи окремих компонентів. За допомогою метрик можна відстежувати такі параметри, як завантаження процесора, використання оперативної пам'яті, кількість запитів до бази даних або час відповіді сервісу. Регулярний збір та аналіз метрик допомагає виявляти аномалії, прогнозувати можливі збої та оптимізувати роботу системи.

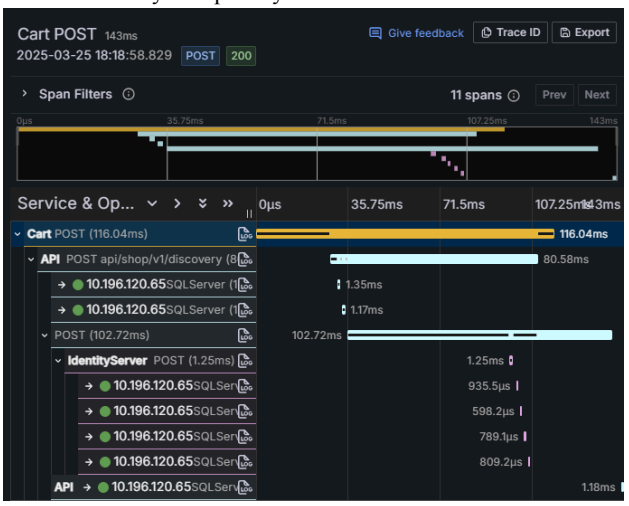

Рисунок 1. Трейс виконання запиту до серверної частини додатку

На рисунку 2 зображена метрика використання центрального процесора серверу, на якому виконується додаток. Стани *privileged*, *user*, *dpc* та *interrupt* являють собою навантаження процесу в різних режимах роботи, та в сумі показують навантаження на процесор, яке складає 20% на момент скріншоту графіка, що є нормою. *Idle* показує простій процесору, тобто обернене значення до навантаження.

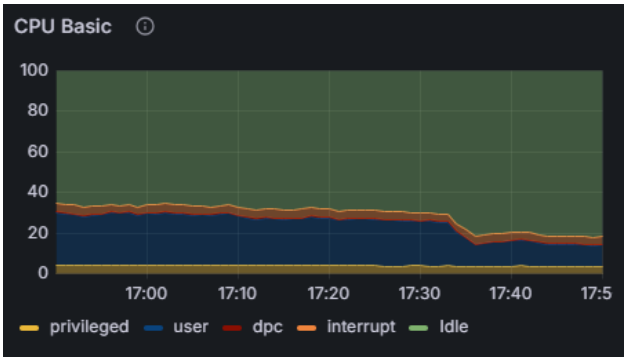

Рисунок 2. Метрика використання центрального процесору серверу

3.4. Використання логів для відлагодження додатку

Логи це структуровані або неструктуровані текстові записи, які фіксують події та стани системи або її окремих компонентів під час роботи. Вони є одним із основних джерел інформації для розуміння поведінки програмного забезпечення, виявлення помилок і проведення пост-інцидентного аналізу. Логування дозволяє розробникам і системним адміністраторам отримувати історичний контекст виконання системи, бачити послідовність дій та операцій, що відбулися в певний момент часу. Це незамінний інструмент для діагностики проблем, аналізу продуктивності та забезпечення стабільності роботи сервісів.

На рисунку 3 видно логи про завершені запити до серверної частини додатку. Кожний запис також містить додаткову інформацію, яку можна переглянути відкривши конкретний запис. Додаткова інформація допоможе швидше знайти та усунути джерело проблеми.

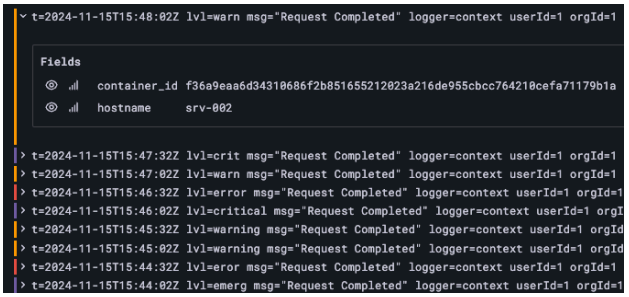

Рисунок 3. Логи серверного додатку

4. ВИСНОВКИ

Проведена робота показує успішну розробку серверного додатку, який ефективно виконує свої функції. Комплексний аналіз продуктивності з використанням логів, трейсів і метрик виявився ефективним для моніторингу та оптимізації системи. Результати аналізу підтвердили, що додаток має високу швидкодію (143 мс на запит) та стабільність (20% навантаження на ЦП). Це підтверджує, що система відповідає вимогам щодо надійності, масштабованості та швидкої реакції.

Список літератури

[1] Len B., Paul C., Rick K. Software Architecture in Practice (SEI Series in Software Engineering), 3rd Edition, 2012, pp. 73-90.

[2] Burns B., Grant B., Oppenheimer D., Brewer E., Wilkes J. Designing Distributed Systems: Patterns and Paradigms for Scalable, Reliable Services. Addison-Wesley Professional, 2018.

[3] Hohpe G., Woolf B. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley, 2004.

[4] Nygard M. Release It!: Design and Deploy Production-Ready Software. Pragmatic Bookshelf, 2007.

[5] Prometheus Authors. Prometheus: Monitoring system & time series database. URL: https://prometheus.io, Accessed 2024.

[6] Jaeger Authors. Jaeger – A Distributed Tracing System. URL: https://www.jaegertracing.io, Accessed 2024.

[7] Elasticsearch Authors. ELK Stack – Elasticsearch, Logstash, Kibana. URL: https://www.elastic.co/what-is/elk-stack, Accessed 2024.