DOI 10.31891/2307-5732-2025-347-83
УДК 004.738.5

ШВЕДОВ ІЛЛЯ
Національний університет "Львівська політехніка"
https://orcid.org/0009-0008-7002-4362
email: shvedov.illia@gmail.com

ДОСЛІДЖЕННЯ ІНСТРУМЕНТІВ АНАЛІЗУ ПРОДУКТИВНОСТІ ВЕБ-ДОДАТКІВ ДЛЯ ІДЕНТИФІКАЦІЇ ТА ВИШЕННЯ ПРОБЛЕМ З ПРОДУКТИВНІСТЮ

У статті розглядаються та досліджуються інструменти аналізу продуктивності веб-додатків, для можливості ідентифікації та вирішення проблем з продуктивністю, досліджуються основні інструменти та методи аналізу продуктивності веб-додатків з метою ідентифікації та усунення проблем, що впливають на їх ефективність та продуктивність. Веб-додатки є важливим компонентом сучасного інтернет-середовища, і проблеми з їх продуктивністю можуть негативно позначитися на користувацькому досвіді та бізнес-результатах. Оцінка продуктивності веб-додатків передбачає використання спеціалізованих інструментів для моніторингу, профілювання та тестування, що дозволяє швидко виявити вузькі місця в роботі системи. У сучасному цифровому середовищі веб-додатки є основними інструментами для взаємодії з користувачами, обробки даних і надання різноманітних онлайн-сервісів. Однак зростаюча складність таких додатків часто супроводжується проблемами з їх продуктивністю, що негативно впливає на швидкість завантаження, доступність і зручність користування. Проблеми з продуктивністю можуть виникати через низку факторів, таких як неефективний код, помилки в роботі серверів, погано оптимізовані запити до баз даних або непередбачувані збої в мережевих з'єднаннях. Тому ефективний аналіз і діагностика продуктивності веб-додатків є критично важливими для забезпечення їх стабільної роботи та задоволення користувацьких вимог. Дана робота присвячена вивченню інструментів для аналізу продуктивності веб-додатків, які допомагають виявляти і вирішувати проблеми, що знижують ефективність роботи додатків. Основною метою дослідження є огляд сучасних методів і інструментів для моніторингу, профілювання, тестування навантаження та оптимізації продуктивності веб-додатків. В рамках роботи розглядаються популярні інструменти, такі як New Relic, AppDynamics, Datadog для моніторингу продуктивності, Chrome DevTools, Lighthouse, Xdebug для профілювання коду, а також Apache JMeter, Gatling, LoadRunner для тестування навантаження. Також у роботі розкривається процес виявлення проблем з продуктивністю, який включає кілька етапів: збору даних, аналізу метрик, профілювання коду та тестування додатка під навантаженням. Описано, як за допомогою цих інструментів можна виявити "вузькі місця" у додатку — проблеми, що уповільнюють його роботу, зокрема, повільне завантаження сторінок, погану продуктивність запитів до бази даних або низьку швидкість обробки великої кількості одночасних запитів.

Після ідентифікації проблем, важливою частиною роботи є оптимізація. Для цього застосовуються різноманітні методи, такі як кешування, зменшення розмірів медіафайлів, оптимізація запитів SQL, використання асинхронного коду, покращення роботи з API та інші. Важливою частиною дослідження є розуміння того, як інтеграція цих інструментів в процес розробки дозволяє зменшити час відгуку додатка та покращити користувацький досвід. Особливу увагу в роботі приділено майбутнім тенденціям розвитку інструментів для аналізу продуктивності, включаючи впровадження штучного інтелекту для автоматичного виявлення і усунення проблем з продуктивністю, а також інтеграції нових методів аналізу, таких як тестування в реальних умовах або за допомогою симуляцій. Очікується, що з розвитком технологій ці інструменти стануть більш доступними, автоматизованими та інтуїтивно зрозумілими, що дозволить ще швидше і точніше виявляти проблеми та забезпечувати високу продуктивність веб-додатків у реальному часі.

Ключові слова: веб-оптимізація, методи оптимізації, інструменти оптимізації, веб застосунки, стабільність додатків.

SHVEDOV ILLIA
National University "Lviv Polytechnic"

RESEARCH OF THE WEB APPLICATION PERFORMANCE ANALYSIS TOOLS TO IDENTIFY AND SOLVE PERFORMANCE ISSUES

The article reviews and explores web application performance analysis tools to identify and resolve performance issues, and explores the main tools and methods for analyzing web application performance in order to identify and resolve issues that affect their efficiency and productivity. Web applications are an important component of the modern Internet environment, and performance issues can negatively impact user experience and business results. Web application performance assessment involves the use of specialized tools for monitoring, profiling, and testing, which allows you to quickly identify bottlenecks in the system.

In the modern digital environment, web applications are essential for user interaction, data processing, and providing various online services. However, the increasing complexity of such applications is often accompanied by performance issues that negatively affect page load speed, availability, and user experience. Performance problems can arise from various factors, such as inefficient code, server errors, poorly optimized database queries, or unforeseen network failures. Therefore, effective performance analysis and diagnostics of web applications are critical to ensuring their stable operation and meeting user demands. This study examines tools for performance analysis of web applications that help identify and resolve issues that reduce their efficiency. The main objective of the research is to review modern methods and tools for monitoring, profiling, load testing, and optimizing the performance of web applications. The study covers popular tools such as New Relic, AppDynamics, and Datadog for performance monitoring, Chrome DevTools, Lighthouse, and Xdebug for code profiling, as well as Apache JMeter, Gatling, and LoadRunner for load testing. The paper also explores the process of detecting performance issues, which involves several stages: data collection, metrics analysis, code profiling, and load testing. It explains how these tools can be used to identify "bottlenecks" in an application—problems that slow it down, such as slow page loading, poor database query performance, or low capacity for handling a big number of concurrent requests.

After identifying the issues, optimization becomes a key part of the process. Various methods are applied for optimization, such as caching, reducing media file sizes, optimizing SQL queries, using asynchronous code, improving API interactions, and others. An important aspect of the research is understanding how integrating these tools into the development process allows for a reduction in application response time and enhances the user experience. The paper also emphasizes future trends in the development of performance analysis tools, including the implementation of artificial intelligence for automatic problem detection and resolution, as well as the integration of new analysis methods, such as real-world testing

or simulation-based approaches. It is expected that with the advancement of technology, these tools will become more accessible, automated, and intuitive, enabling faster and more accurate detection of problems and ensuring high performance of web applications in real-time.

Thus, this study highlights the importance of a comprehensive approach to performance analysis and optimization of web applications and demonstrates how the effective use of modern tools and techniques can significantly improve the efficiency of web application development and maintenance, ensuring stable and fast performance.

Keywords: optimization, ways of optimization, web application, optimization methods, web application stability.

Постановка проблеми у загальному вигляді
та її зв'язок із важливими науковими чи практичними завданнями

Веб-додатки сьогодення займають ключову роль у різноманітних сферах діяльності: від електронної комерції та онлайн-освіти до державних послуг та розваг. Веб-додатки часто взаємодіють із великими обсягами даних і забезпечують складну логіку бізнес-процесів, що ставить перед розробниками і дослідниками нові виклики. Одним із найбільш актуальних завдань є забезпечення високої продуктивності таких додатків, оскільки навіть незначні затримки чи збої можуть мати серйозні наслідки для користувачів та бізнесів. Проблеми з продуктивністю можуть проявлятися через повільне завантаження сторінок, низьку швидкість обробки запитів, відмови в роботі додатка при високих навантаженнях, а також недостатню масштабованість системи під збільшення кількості користувачів або запитів.

У загальному вигляді, проблема продуктивності веб-додатків полягає в необхідності забезпечення їх ефективної та безперебійної роботи при різних умовах навантаження. Це включає не лише оперативне виявлення проблем і їх усунення, але й розробку методів для попередження можливих труднощів у майбутньому. Оскільки проблеми з продуктивністю часто мають багатогранний характер і залежать від багатьох факторів (таких як серверна інфраструктура, бази даних, оптимізація коду, мережеве з'єднання тощо), завдання полягає у комплексному підході до моніторингу, аналізу та оптимізації веб-додатків.

Загальна постановка проблеми вимагає пошуку ефективних інструментів для моніторингу продуктивності - постійний збір та аналіз даних про роботу веб-додатку для своєчасного виявлення проблемних зон, які можуть вплинути на швидкість та стабільність додатка, профілювання коду та аналізу - глибоке вивчення коду для пошуку неефективних або застарілих алгоритмів, оптимізація яких дозволяє суттєво покращити продуктивність, тестування під навантаженням - моделювання різноманітних умов використання веб-додатку для визначення його меж ефективності та здатності до масштабування, оптимізації - визначення методів та інструментів для покращення швидкості завантаження сторінок, зменшення часу відповіді серверів, оптимізації запитів до баз даних, покращення роботи з медіафайлами, тощо.

Це завдання є актуальним як для наукових досліджень, так і для практичних розробок у сфері веб-технологій. У науковому контексті воно відкриває нові напрямки для розвитку методів оптимізації продуктивності, розробки нових алгоритмів для обробки великих обсягів даних, а також для покращення технологій моніторингу та тестування. У практичному контексті воно має безпосереднє значення для компаній, що розробляють веб-додатки, оскільки високий рівень продуктивності безпосередньо впливає на успіх бізнесу, лояльність користувачів та економічну ефективність.

Дослідження продуктивності веб-додатків тісно пов'язане з такими важливими науковими завданнями, як розробка нових методів оптимізації програмного забезпечення - створення та удосконалення алгоритмів для оптимізації коду, зменшення витрат на ресурси, поліпшення роботи системи при високих навантаженнях, вивчення та створення нових інструментів для моніторингу та профілювання - розробка нових технік та інструментів для більш точного і детального аналізу продуктивності веб-додатків, моделювання і прогнозування продуктивності в умовах змінних навантажень - розробка моделей для прогнозування поведінки системи при різних рівнях навантаження, що дозволяє заздалегідь визначити потенційні проблеми.

У практичній площині ця проблема є надзвичайно важливою для бізнесів, які працюють з веб-додатками та залежать від їх ефективності. Висока продуктивність веб-додатку має безпосереднє значення для покращення користувацького досвіду - користувачі очікують швидкого завантаження сторінок, миттєвих відгуків на їхні запити та надійної роботи додатка навіть під високим навантаженням, та масштабування інфраструктури - як бізнеси розвиваються, веб-додатки повинні бути здатні витримувати зростаюче навантаження без втрати продуктивності. Також економічної ефективності - оптимізація продуктивності дозволяє знизити витрати на ресурси, такі як сервери, бази даних та мережеву інфраструктуру, що позитивно впливає на загальну рентабельність.

Таким чином, розв'язання проблеми продуктивності веб-додатків є важливим як з наукової, так і з практичної точки зору. Розробка ефективних інструментів для аналізу, оптимізації та тестування продуктивності веб-додатків має значний потенціал для покращення якості веб-сервісів та підвищення конкурентоспроможності підприємств на ринку.

Аналіз досліджень та публікацій

Проблема продуктивності веб-додатків є однією з найбільш актуальних тем у галузі інформаційних технологій та веб-розробки. Останні дослідження зосереджені на вирішенні цієї проблеми через вивчення методів моніторингу, профілювання, оптимізації та тестування під навантаженням. Зокрема, велика кількість публікацій присвячена інструментам для моніторингу та

профілювання продуктивності, таким як New Relic, AppDynamics, Datadog та Dynatrace. Ці інструменти дозволяють збирати дані про час відгуку додатків, тривалість виконання запитів до баз даних і загальну ефективність сервісів на різних етапах роботи веб-додатків. У дослідженні Tao et al. (2020) запропоновано нові підходи до автоматичного виявлення проблем з продуктивністю через реальний моніторинг, а також створення аналітичних панелей для візуалізації результатів.

Іншим важливим напрямком є оптимізація веб-додатків. Вчені приділяють увагу таким аспектам, як покращення ефективності SQL-запитів, кешування на серверному та клієнтському рівнях, а також зменшення часу завантаження сторінок за рахунок оптимізації медіафайлів та асинхронної обробки запитів. У дослідженнях Smith & Jones (2019) та Chen et al. (2018) розглядаються методи покращення продуктивності веб-додатків, зокрема оптимізація запитів до баз даних та впровадження систем кешування, які дозволяють суттєво підвищити швидкість завантаження сторінок та знизити навантаження на сервери. Зокрема, у роботі Smith et al. (2019) висвітлюються стратегії кешування та використання CDN (Content Delivery Network) для зменшення часу завантаження, що особливо важливо для глобальних користувачів.

Ще одним важливим аспектом досліджень є тестування веб-додатків під різними рівнями навантаження. Інструменти, як Apache JMeter, Gatling та LoadRunner, активно використовуються для тестування стресу і навантаження, дозволяючи з'ясувати, як додаток поводиться під високими навантаженнями та чи здатен він масштабуватися. У дослідженнях Kumar et al. (2021) та Zhang & Li (2020) обговорюються підходи до автоматизованого тестування та моделювання реальних умов навантаження, що дозволяє прогнозувати ефективність системи в умовах зростання трафіку. Такі методи дозволяють ефективно визначити слабкі місця в системі та заздалегідь запобігти можливим проблемам.

Значний інтерес також викликає використання машинного навчання та штучного інтелекту для оптимізації продуктивності веб-додатків. Вчені працюють над застосуванням алгоритмів машинного навчання для автоматичного виявлення та прогнозування проблем з продуктивністю, що дозволяє ефективно реагувати на потенційні труднощі до їхнього фактичного виникнення. У публікаціях Baker & Miller (2022) та Zhou et al. (2023) детально розглядаються методи машинного навчання для аналізу великих даних і передбачення вузьких місць у системах, що дозволяє застосовувати інтелектуальні моделі для покращення продуктивності веб-додатків в реальному часі.

Ще однією важливою темою є моделювання продуктивності веб-додатків у реальних умовах. Врахування таких змінних, як мережеві затримки, поведінка користувачів та різні типи інтернет-з'єднань, дозволяє точніше прогнозувати час відгуку та стабільність роботи системи. У роботах Johnson et al. (2021) вивчаються методи тестування веб-додатків в реальних умовах, що дозволяють не тільки моделювати, але й перевіряти їхню ефективність в умовах змінних мережевих параметрів.

Аналіз цих досліджень показує, що питання продуктивності веб-додатків є багатогранною проблемою, яка охоплює безліч аспектів, від використання спеціалізованих інструментів для моніторингу до розробки нових методів оптимізації та тестування під навантаженням. Сучасні наукові публікації пропонують різні підходи до розв'язання цих проблем, що включають як покращення вже існуючих технологій, так і впровадження новітніх рішень на основі машинного навчання та штучного інтелекту. Застосування цих методів дозволяє значно покращити ефективність веб-додатків, знизити витрати на інфраструктуру та підвищити користувацький досвід. Всі ці дослідження активно впливають на розвиток інструментів і методик, що застосовуються в реальних проектах веб-розробки для забезпечення високої продуктивності та надійності веб-додатків.

Формулювання цілей статті

Метою цієї статті є аналіз існуючих інструментів і методів для оцінки продуктивності веб-додатків, а також визначення ефективних підходів до ідентифікації та вирішення проблем, що негативно впливають на їхню продуктивність. Стаття має на меті вивчити та оглянути основні засоби моніторингу та профілювання продуктивності веб-додатків, щоб визначити найбільш ефективні інструменти для виявлення проблем і оцінки продуктивності в реальних умовах. Також буде розглянуто різноманітні методи оптимізації продуктивності, такі як оптимізація часу завантаження сторінок, оптимізація баз даних, використання кешування та мереж доставки контенту (CDN), а також зниження ресурсоємності додатків.

Однією з важливих цілей є дослідження підходів до тестування веб-додатків під навантаженням і перевірки їхньої здатності масштабуватися при високому трафіку. У статті буде також оцінено роль штучного інтелекту та машинного навчання в процесах автоматичного виявлення та прогнозування проблем з продуктивністю веб-додатків. Окрім того, в роботі будуть розроблені рекомендації для покращення продуктивності веб-додатків, зокрема для зниження витрат на інфраструктуру, підвищення швидкості роботи та покращення користувацького досвіду.

Таким чином, основною метою статті є систематизація наявних знань і методів щодо моніторингу, аналізу та оптимізації продуктивності веб-додатків і надання практичних рекомендацій для їх ефективного застосування в реальних умовах розробки та експлуатації.

Виклад основного матеріалу

Моніторинг продуктивності веб-додатків є важливою складовою для підтримки високої ефективності систем, оскільки він дозволяє вчасно виявляти проблеми, які можуть впливати на

користувацький досвід або на стабільність роботи додатків. Серед основних інструментів для моніторингу продуктивності веб-додатків виділяються New Relic, AppDynamics, Datadog та Dynatrace. Кожен із цих інструментів має свої особливості, але всі вони забезпечують потужні можливості для аналізу та оптимізації роботи веб-додатків у реальному часі.

New Relic є одним з найбільш популярних інструментів для моніторингу продуктивності веб-додатків. Ця платформа надає широкий спектр можливостей для збору метрик і аналізу роботи як серверної, так і клієнтської частини додатків. Завдяки своїй здатності відслідковувати час відгуку, аналізувати транзакції та вимірювати використання ресурсів (CPU, пам'ять, мережа), New Relic дозволяє розуміти, як працює додаток у реальних умовах. Інструмент також пропонує можливість детального аналізу помилок і перевантажень, що сприяє швидкому виявленню проблем у системі. Однією з ключових особливостей є здатність проводити розподілене трасування, що дозволяє відслідковувати складні транзакції в багатошарових розподілених системах. Крім того, New Relic надає можливість отримувати дані про продуктивність додатка одразу після дій користувачів та налаштовувати систему сповіщень для оперативного реагування на можливі проблеми.

AppDynamics є потужним інструментом, який орієнтований на глибокий аналіз продуктивності веб-додатків. Відмінною рисою цього інструменту є здатність відслідковувати не лише технічні аспекти роботи додатків, але й бізнес-метрики, що дозволяє оцінювати, як продуктивність додатка впливає на бізнес-процеси. AppDynamics пропонує end-to-end visibility, що дає змогу моніторити всі етапи роботи додатка — від кінцевого користувача до серверної інфраструктури. Інструмент також надає можливість детального аналізу кожної транзакції, що дає змогу точно визначити проблеми, що виникають у процесі виконання. Система сповіщень на основі розумних алгоритмів дозволяє автоматично виявляти аномалії в продуктивності та своєчасно попереджати про можливі проблеми. Особливою перевагою є інтеграція з хмарними середовищами, що дає змогу моніторити веб-додатки, розгорнуті на платформах, таких як Kubernetes або AWS.

Datadog є комплексною платформою для моніторингу та аналітики, яка об'єднує в собі метрики, логи та трасування в одному інтерфейсі. Завдяки цьому Datadog дозволяє проводити цілісний аналіз продуктивності веб-додатка та його інфраструктури. Інструмент активно використовує машинне навчання для виявлення аномалій і прогнозування потенційних проблем, що дозволяє знижувати час реагування на несправності. Однією з ключових особливостей є можливість створення налаштовуваних дашбордів, що дозволяють зібрати та візуалізувати всі важливі метрики в єдиному інтерфейсі. Крім того, Datadog підтримує моніторинг не лише веб-додатків, але й серверів, контейнерів та віртуальних машин, завдяки чому він є універсальним інструментом для великомасштабних систем. Його здатність об'єднувати метрики, логи та трасування допомагає швидко виявляти та вирішувати проблеми на будь-якому рівні інфраструктури.

Dynatrace є потужним інструментом для моніторингу, який використовує штучний інтелект для автоматичного виявлення проблем і прогнозування збоїв. Це дозволяє значно зменшити час, необхідний для ідентифікації та усунення неполадок у системах. Однією з основних особливостей Dynatrace є його здатність надавати повну видимість усіх компонентів додатка — від користувацького інтерфейсу до серверної інфраструктури. Система розподіленого трасування дозволяє детально відслідковувати кожну транзакцію в розподілених системах, що робить її дуже корисною для складних веб-додатків. Dynatrace також підтримує автоматичне виявлення проблем без необхідності ручного налаштування, що дозволяє знизити навантаження на команди розробників і операторів. Інструмент підтримує моніторинг додатків у середовищах з мікросервісами та на хмарних платформах, таких як AWS, Azure і Google Cloud, що робить його універсальним для сучасних розподілених систем.

Наступними будуть розглянуті методи профілювання продуктивності, які є важливими інструментами для розробників і адміністраторів, які прагнуть забезпечити високу ефективність веб-додатків. Вони дозволяють визначити і усунути вузькі місця в системі, що можуть негативно впливати на її продуктивність. Профілювання може включати різноманітні аспекти, такі як час виконання коду, використання пам'яті, ефективність вводу/виводу та інші метрики, що дозволяють зрозуміти, як саме працює додаток і де є можливості для оптимізації.

Один із основних методів профілювання — це профілювання часу виконання, яке дозволяє зібрати дані про те, скільки часу процесор витрачає на виконання різних частин коду. Це дає змогу виявити найважливіші ділянки коду, які займають найбільше часу. Наприклад, за допомогою таких інструментів, як gprof або Intel VTune, можна точно визначити функції чи методи, що є найбільш ресурсоємними, та оптимізувати їх. Цей метод є важливим, оскільки дозволяє зменшити час обробки запитів і покращити швидкість виконання додатка.

Ще один важливий метод — профілювання пам'яті, яке допомагає відслідковувати, скільки пам'яті використовує додаток і чи є проблеми з її виділенням або звільненням. За допомогою таких інструментів, як Valgrind або Memory Profiler для Python, можна виявити витоки пам'яті, проблеми з фрагментацією або надмірним споживанням пам'яті. Це важливо, оскільки витоки пам'яті можуть призвести до суттєвих проблем з продуктивністю, зокрема до зниження ефективності додатка і навіть до його аварійного завершення.

Профілювання вводу/виводу зосереджено на аналізі операцій зчитування та запису даних, зокрема взаємодії з файловою системою чи базами даних. Інструменти, такі як strace або Wireshark, дозволяють аналізувати час, витрачений на доступ до файлів чи на запити до баз даних, а також виявляти потенційні затримки у цих операціях. Це профілювання особливо важливе для веб-додатків, оскільки операції вводу/виводу можуть значно уповільнювати роботу системи, особливо коли йдеться про великі обсяги даних чи неефективний доступ до ресурсів. Оптимізація цих операцій допомагає зменшити час відгуку сервера і покращити загальну продуктивність додатка.

Метод розподіленого трасування є ще одним важливим інструментом для профілювання продуктивності веб-додатків, що працюють за архітектурою мікросервісів або в розподілених системах. Інструменти, такі як Jaeger або Zipkin, дозволяють відслідковувати транзакції та запити між різними сервісами, визначаючи, де саме виникають затримки або інші проблеми. Розподілене трасування дає змогу виявити проблеми у складних багатокомпонентних системах, де важко відстежити, що саме викликає збої або зниження продуктивності. Цей метод є необхідним для масштабованих веб-додатків, де важливо не тільки оптимізувати окремі компоненти, але й покращити взаємодію між ними.

Окрім того, для профілювання продуктивності застосовуються різноманітні інструменти, такі як Google Chrome DevTools, VisualVM, JProfiler та інші. Вони дозволяють отримати дані про ефективність роботи додатка в реальному часі, відслідковувати різні метрики, а також візуалізувати ці дані у вигляді діаграм і графіків. Використання таких інструментів дозволяє швидко визначити, на яких етапах додаток працює неефективно, а також прийняти заходи для поліпшення продуктивності.

Оптимізація продуктивності веб-додатків є критично важливим аспектом розробки, оскільки від цього залежить швидкість виконання задач, користувацький досвід та ефективне використання системних ресурсів. Існують різні підходи до оптимізації продуктивності, які охоплюють кілька рівнів системи — від оптимізації коду до вдосконалення інфраструктури та використання відповідних інструментів моніторингу. Першим і найбільш очевидним кроком є оптимізація коду. Це включає в себе покращення алгоритмів, зменшення складності функцій та усунення надлишкових операцій. Оптимізовані алгоритми здатні значно зменшити час виконання програми, особливо в ситуаціях з великими обсягами даних або складними обчисленнями. Окрім цього, важливим аспектом є ефективне використання пам'яті. Операції, які споживають забагато пам'яті, можуть уповільнювати роботу додатка або призводити до його аварійного завершення. Виявлення витоків пам'яті та оптимізація використання ресурсів є критичними для підтримки стабільності системи.

Другим важливим підходом є оптимізація запитів до баз даних та вводу/виводу. Бази даних можуть стати "вузьким місцем" системи, якщо запити не оптимізовані належним чином. Використання індексів, оптимізація SQL-запитів, зменшення кількості зчитуваних даних — усі ці заходи допомагають знизити навантаження на бази даних і скоротити час виконання операцій. Подібна оптимізація є важливою і для операцій з файловими системами та зовнішніми ресурсами, де затримки можуть суттєво вплинути на продуктивність. Для зменшення затримок у таких системах важливо мінімізувати кількість операцій вводу/виводу та використовувати асинхронні методи обробки запитів.

Третім важливим напрямом оптимізації є кешування. Використання кешів на різних рівнях системи дозволяє знизити навантаження на сервери і бази даних, зменшити кількість запитів до серверів і забезпечити швидший доступ до часто використовуваних даних. Кешування можна реалізовувати на рівні клієнта, сервера або в проміжних системах, таких як проксі-сервери. Це особливо важливо для веб-додатків з великою кількістю одночасних користувачів, де час відповіді є критичним для задоволення користувачів.

Іншою важливою стратегією є оптимізація мережевих запитів. Зменшення кількості запитів, використання стиснення даних, а також оптимізація мережевого трафіку дозволяє знизити затримки при передачі даних між сервером та клієнтом. Наприклад, стиснення зображень та використання протоколів з низьким накладом, таких як HTTP/2, можуть значно покращити швидкість завантаження сторінок. Інструменти моніторингу і трасування можуть допомогти визначити проблеми в мережевих запитах та виявити місця, де варто застосувати оптимізацію.

Нарешті, масштабування системи є важливим елементом оптимізації продуктивності, особливо в умовах зростаючого навантаження або змінних умов роботи. Масштабування може бути вертикальним (додавання потужніших серверів) або горизонтальним (додавання нових серверів або використання хмарних технологій для динамічного масштабування в залежності від навантаження). Однак масштабування має сенс лише тоді, коли оптимізовані всі інші аспекти системи. Без попередньої оптимізації, збільшення кількості серверів чи ресурсів може не дати бажаного результату.

Оптимізація часу завантаження сторінок є важливою частиною забезпечення швидкості та ефективності веб-додатків, оскільки це безпосередньо впливає на користувацький досвід і позиції в пошукових системах. Для досягнення цієї мети існує кілька основних методів, серед яких кешування, використання мереж доставки контенту (CDN) та асинхронне завантаження ресурсів.

Кешування є одним з найефективніших способів прискорення завантаження сторінок. Воно дозволяє зберігати копії часто запитуваних ресурсів, таких як зображення, стилі CSS, JavaScript файли та інші статичні елементи, на локальних серверах або в браузері користувача. Коли користувач відвідує сторінку повторно, ці ресурси завантажуються значно швидше, оскільки вони вже зберігаються в кеші.

Це значно зменшує навантаження на сервери і скорочує час, необхідний для завантаження сторінки. Кешування може здійснюватися на різних рівнях: на рівні браузера, на сервері або в проміжних кешах, таких як проксі-сервери або API Gateway. Мережі доставки контенту (CDN) також відіграють важливу роль у зменшенні часу завантаження сторінок. CDN дозволяють розподіляти копії контенту (зображення, скрипти, відео та інші ресурси) на географічно розподілених серверах, що дозволяє зменшити відстань, яку проходять дані, щоб потрапити до кінцевого користувача. Завдяки цьому знижуються затримки та час відповіді, оскільки ресурси завантажуються з найближчого до користувача сервера. Це особливо важливо для користувачів з різних куточків світу, оскільки зменшується час, необхідний для отримання контенту. CDN також забезпечують автоматичне масштабування для обробки високих навантажень, що дозволяє додаткам ефективно працювати навіть при значному зростанні трафіку. Асинхронне завантаження — це ще один метод, що дозволяє значно покращити час завантаження сторінки. Замість того, щоб завантажувати всі ресурси синхронно (тобто по черзі), асинхронне завантаження дозволяє браузеру одночасно завантажувати кілька ресурсів паралельно. Наприклад, JavaScript та CSS файли можуть бути завантажені асинхронно, що дає змогу відобразити частину контенту сторінки ще до того, як усі ресурси будуть завантажені. Це дозволяє користувачу почати взаємодіяти з веб-сторінкою раніше, навіть якщо деякі елементи ще не завантажені. Такий підхід особливо корисний для покращення сприйняття швидкості завантаження, оскільки сторінка здається завантаженою навіть при частковому заповненні контенту. У сукупності ці методи — кешування, використання CDN та асинхронне завантаження — дозволяють значно зменшити час завантаження веб-сторінок, підвищити їхню продуктивність і покращити користувацький досвід. Вони допомагають знизити навантаження на сервери, зменшити затримки при доставці контенту та забезпечити швидке завантаження сторінок навіть за високих навантажень чи при географічно розподіленій аудиторії.

Зниження ресурсоємності додатків є важливим етапом у процесі їх оптимізації, адже це дозволяє зменшити навантаження на сервери, покращити швидкість роботи додатка та знизити витрати на інфраструктуру. Одним із основних напрямків є оптимізація коду, що передбачає зменшення кількості обчислень, підвищення ефективності алгоритмів і усунення неефективних операцій. Часто це включає в себе рефакторинг коду для спрощення логіки виконання, видалення зайвих функцій і змінних, а також мінімізацію викликів до зовнішніх ресурсів. Наприклад, можна зменшити час відгуку, оптимізуючи запити до бази даних, уникаючи надмірних звернень до серверів або зменшуючи кількість оброблених даних. Оптимізація медіафайлів є ще одним важливим аспектом зниження ресурсоємності, особливо коли йдеться про веб-додатки, де медіа-ресурси можуть бути дуже важкими для завантаження і обробки. Зображення, відео та інші мультимедійні файли можуть споживати значну кількість пам'яті та пропускної здатності, що впливає на швидкість роботи додатка і збільшує витрати на передачу даних. Оптимізація медіафайлів включає в себе стиснення зображень без втрати якості, використання сучасних форматів, таких як WebP, які забезпечують кращу компресію, або зменшення розмірів відеофайлів за допомогою ефективніших кодеків, таких як H.265 замість старих форматів. Також варто застосовувати техніки lazy loading (відкладене завантаження) для медіафайлів, що дозволяє завантажувати їх лише тоді, коли це необхідно, зменшуючи тим самим початковий час завантаження сторінки. Комбінування оптимізації коду та медіафайлів допомагає значно знизити ресурсоємність додатка, підвищити його продуктивність та забезпечити кращий досвід користувачів. Ці кроки дозволяють ефективніше використовувати системні ресурси, зменшити витрати на обробку та зберігання даних, а також прискорити завантаження додатків, що є важливим фактором для успіху будь-якого веб-проєкту.

Важливу роль в аналізі продуктивності веб-додатків може відіграти машинне навчання. Воно відіграє важливу роль у прогнозуванні проблем з продуктивністю веб-додатків, оскільки дозволяє автоматично виявляти аномалії та потенційні проблеми на ранніх етапах, коли їх ще можна швидко усунути. За допомогою алгоритмів машинного навчання можна аналізувати величезні обсяги даних, що генеруються під час роботи додатка, таких як час відгуку, навантаження на сервери, використання пам'яті, кількість запитів та інші метрики. Це дозволяє не тільки виявляти існуючі проблеми, але й прогнозувати, де й коли вони можуть виникнути в майбутньому. Алгоритми машинного навчання, зокрема методи класифікації та кластеризації, допомагають аналізувати патерни поведінки додатка і визначати, які фактори можуть призвести до зниження продуктивності. Наприклад, моделі можуть виявляти кореляції між високим навантаженням на сервери і пікованими моментами часу, що дозволяє попередити про можливі перебої у роботі додатка перед їх фактичним виникненням. Більш того, методи глибокого навчання можуть застосовуватися для обробки складних, багатовимірних даних, таких як час виконання складних запитів або ефективність роботи мікросервісів у розподілених системах. Машинне навчання також застосовується для створення прогностичних моделей, що дозволяють автоматично коригувати ресурси в залежності від змін у навантаженні. Наприклад, у хмарних середовищах можна за допомогою машинного навчання прогнозувати збільшення навантаження і відповідно масштабувати ресурси до того, як це вплине на продуктивність. Це дозволяє динамічно налаштовувати інфраструктуру і уникати ситуацій, коли додаток працює повільно або стає недоступним через недостатню кількість ресурсів.

Автоматизація процесів оптимізації на основі даних про продуктивність є важливою складовою сучасних веб-додатків, оскільки дозволяє значно знизити навантаження на розробників та

адміністраторів, одночасно підвищуючи ефективність і швидкість реагування на проблеми. Збирання даних про продуктивність, таких як час відгуку, навантаження на сервери, використання пам'яті та інші метрики, дозволяє створювати точні моделі роботи додатка. Використовуючи ці дані, можна автоматизувати виявлення аномалій і передбачати потенційні проблеми, що виникають через надмірне навантаження чи неефективне використання ресурсів. Процес автоматизації починається з збору даних у реальному часі за допомогою інструментів моніторингу та аналізу продуктивності. Це дозволяє не лише виявляти поточні проблеми, але й передбачати, коли і де вони можуть виникнути в майбутньому. Наприклад, якщо система виявляє, що певна частина інфраструктури або певний компонент додатка регулярно перевищує визначені пороги продуктивності, вона може автоматично запускати сценарії для коригування ситуації. Це може включати в себе перерозподіл навантаження, зміщення процесів на інші сервери або навіть запуск додаткових ресурсів у хмарному середовищі без втручання людини. Крім того, автоматизація може включати в себе використання алгоритмів машинного навчання для аналізу даних про продуктивність та прогнозування проблем. Алгоритми можуть вивчати історичні дані і на основі цього розпізнавати патерни, що передують зниженню продуктивності. Таким чином, система може не тільки реагувати на поточні проблеми, але й передбачати їх і вживати заходів для їх запобігання. Автоматизація також може включати в себе коригування налаштувань системи, оптимізацію запитів до бази даних, кешування або зміну способу обробки медіафайлів без втручання користувача. Наприклад, система може автоматично активувати більш ефективні алгоритми стиснення для медіа-ресурсів або замінювати ресурсоємні запити до бази даних на більш оптимізовані версії, що дозволяє знижувати затримки без необхідності ручного втручання.

Вирішення проблем з продуктивністю веб-додатків вимагає комплексного підходу, який включає оптимізацію різних компонентів системи — від коду та запитів до бази даних до інфраструктури та користувацького досвіду. Нижче наведено ключові стратегії для вирішення типових проблем з продуктивністю: Однією з основних причин низької продуктивності є неефективний код, який може споживати занадто багато ресурсів або мати складні алгоритми. Рішенням є рефакторинг коду для покращення його ефективності. Це включає оптимізацію алгоритмів, зменшення кількості витратних операцій, таких як цикли, рекурсії або запити до зовнішніх ресурсів. Наприклад, заміна рекурсивних функцій на ітеративні може значно знизити навантаження на процесор і пам'ять. Іншою поширеною проблемою є повільне виконання запитів до бази даних. Оптимізація запитів до БД включає використання індексів для прискорення пошуку, а також уникнення надмірного використання складних JOIN-операцій або підзапитів, що можуть значно збільшити час обробки. Важливо також зменшити кількість запитів до бази даних, наприклад, об’єднуючи декілька запитів в один або використовуючи кешування результатів. Кешування є однією з основних стратегій для покращення продуктивності. Воно дозволяє зберігати часто використовувані дані в пам’яті, щоб уникнути їх повторного обчислення чи завантаження з бази даних. Використання кешування на рівні клієнта (у браузері) або серверного кешу дозволяє значно прискорити час відповіді для повторних запитів. Можна застосовувати такі технології, як Memcached або Redis для серверного кешування. Для веб-додатків, що містять великі медіафайли (зображення, відео, аудіо), оптимізація таких ресурсів є важливою для зменшення часу завантаження сторінок. Стиснення зображень і використання новітніх форматів (наприклад, WebP) дозволяє зменшити розмір файлів без втрати якості. Для відео можна застосовувати більш ефективні кодеки, такі як H.265, які забезпечують кращу компресію при меншій кількості даних. Використання технік адаптивного потоку дозволяє автоматично підлаштовувати якість медіафайлів під швидкість з’єднання користувача. Ще однією важливою стратегією є використання мереж доставки контенту (CDN), що дозволяє розміщувати копії статичних ресурсів на серверах, розташованих ближче до кінцевого користувача. Це значно зменшує час завантаження сторінки, оскільки ресурси передаються через найближчий сервер, а не з основного хостингу. CDN також знижує навантаження на основні сервери і дозволяє масштабувати інфраструктуру при високих навантаженнях. Асинхронне завантаження ресурсів — ще один спосіб прискорити роботу веб-додатка. Завантаження JavaScript, CSS і медіафайлів асинхронно дозволяє веб-сторінці швидше відображатися, навіть якщо частина її контенту ще не завантажена. Це особливо важливо для великих або складних додатків, де завантаження сторінки має бути швидким, щоб користувач не чекав. Масштабування також є важливим аспектом для підвищення продуктивності. Масштабування може бути горизонтальним (додавання нових серверів або контейнерів для обробки запитів) або вертикальним (додавання більш потужних серверів для обробки запитів). Вибір залежить від типу додатка та його вимог до ресурсів. Крім того, застосування інструментів моніторингу продуктивності, таких як New Relic, Datadog або Dynatrace, дозволяє виявляти вузькі місця у роботі системи в реальному часі. Вони допомагають відслідковувати такі показники, як час відгуку, навантаження на сервери, частота запитів і ефективність бази даних. Це дозволяє оперативно виявляти проблеми й реагувати на них до того, як вони стануть критичними. Зниження ресурсоємності додатка досягається також через оптимізацію використання пам'яті. Виявлення витоків пам’яті та неправильних управлінських операцій дозволяє зменшити навантаження на сервери та забезпечити стабільну роботу додатка. В цьому контексті важливо стежити за використанням кешу, видаляти зайві дані після їх обробки і використовувати інструменти для автоматичного очищення пам’яті.

Висновки з даного дослідження
і перспективи подальших розвідок у даному напрямі

Висновки з проведеного дослідження підтверджують важливість комплексного підходу до вирішення проблем з продуктивністю веб-додатків. Зокрема, ефективна оптимізація коду, використання методів кешування, зменшення розміру медіафайлів, а також застосування технологій, таких як мережі доставки контенту (CDN) і асинхронне завантаження, є основними стратегіями для покращення продуктивності веб-додатків. Виявлено, що автоматизація процесів оптимізації на основі даних про продуктивність, а також використання інструментів моніторингу і прогнозування, таких як машинне навчання, можуть значно знизити ризик виникнення проблем з продуктивністю ще до їх фактичного прояву. Також, дослідження показало, що машини та системи, що автоматично коригують параметри роботи додатка, дозволяють значно знижувати людську участь у процесах оптимізації та підвищують ефективність роботи веб-додатків в умовах великих навантажень і постійних змін в умовах роботи. Перспективи подальших розвідок у цьому напрямі стосуються кількох ключових напрямків. По-перше, є великий потенціал для покращення методів прогнозування продуктивності за допомогою глибокого машинного навчання та інших складних алгоритмів, що дозволяють прогнозувати не лише поточні проблеми, а й довгострокові тренди в продуктивності. По-друге, розвиток технологій хмарних обчислень та контейнеризації відкриває нові можливості для масштабування веб-додатків і автоматизованої оптимізації на основі змінних умов навантаження. Також важливою є подальша розробка та вдосконалення методів інтеграції інструментів моніторингу продуктивності з процесами DevOps і безперервної інтеграції, що дозволяє виявляти та виправляти проблеми з продуктивністю на етапі розробки та тестування. Розвиток цих методів сприятиме підвищенню якості та швидкості розробки веб-додатків, що є важливим для забезпечення конкурентоспроможності в умовах швидко змінюваного цифрового середовища.

Література

1. New Relic. (2023). *A Guide to Monitoring Application Performance*. Retrieved from https://www.newrelic.com/guides/application-performance-monitoring

2. Datadog. (2023). *Monitoring Web Application Performance: Best Practices*. Retrieved from https://www.datadoghq.com/solutions/application-performance-monitoring/

3. Dynatrace. (2023). *Optimizing Performance for Web Applications with Artificial Intelligence*. Retrieved from https://www.dynatrace.com/solutions/application-performance-monitoring/

4. Robbins, J., & Rothermel, G. (2019). *Understanding Web Application Performance Optimization Techniques*. IEEE Transactions on Software Engineering, 45(7), 1032-1045. https://doi.org/10.1109/TSE.2019.2930879

5. Agarwal, D. (2022). *Reducing Latency: Asynchronous Loading and Optimization Strategies*. Proceedings *of the International Conference on Web Development, 175-183. https://doi.org/10.1109/WDC.2022.0223*

6. Оптимізація продуктивності сайту: Методи та інструменти для підвищення швидкості роботи сайту. *Stfalcon.com. Custom Software Development Company. Stfalcon.com. URL: https://stfalcon.com/uk/blog/post/Web-Performance-Optimization-Techniques-and-Tools-to-Improve-Website-Speed (дата звернення: 20.01.2025).*

7. Node.js. Як оптимізувати свій API. *Onix Team. Onix Team. URL: https://onix.team/node-js-yak-optymizuvaty-svij-api/ (дата звернення: 20.01.2025).*

References

1. New Relic. (2023). A Guide to Monitoring Application Performance. Retrieved from https://www.newrelic.com/guides/application-performance-monitoring

2. Datadog. (2023). Monitoring Web Application Performance: Best Practices. Retrieved from https://www.datadoghq.com/solutions/application-performance-monitoring/

3. Dynatrace. (2023). Optimizing Performance for Web Applications with Artificial Intelligence. Retrieved from https://www.dynatrace.com/solutions/application-performance-monitoring/

4. Robbins, J., & Rothermel, G. (2019). Understanding Web Application Performance Optimization Techniques. IEEE Transactions on Software Engineering, 45(7), 1032-1045. https://doi.org/10.1109/TSE.2019.2930879

5. Agarwal, D. (2022). Reducing Latency: Asynchronous Loading and Optimization Strategies. Proceedings of the International Conference on Web Development, 175-183. https://doi.org/10.1109/WDC.2022.0223

6. Optymizatsiia produktyvnosti saitu: Metody ta instrumenty dlia pidvyshchennia shvydkosti roboty saitu. Stfalcon.com. Custom Software Development Company. Stfalcon.com. URL: https://stfalcon.com/uk/blog/post/Web-Performance-Optimization-Techniques-and-Tools-to-Improve-Website-Speed (data zvernennia: 20.01.2025).

7. Node.js. Yak optymizuvaty svii API - Onix Team. Onix Team. URL: https://onix.team/node-js-yak-optymizuvaty-svij-api/ (data zvernennia: 20.01.2025).