УДК 681.12

М. М. Шейка[1]

МІКРОСЕРВІСНА АРХІТЕКТУРА: ПЕРЕВАГИ, ВИКЛИКИ ТА НАЙКРАЩІ ПРАКТИКИ

[1] Вінницький національний технічний університет.

Анотація

У статті досліджено мікросервісну архітектуру як сучасний підхід до розробки програмного забезпечення. Проаналізовано її переваги, зокрема гнучкість і масштабованість, а також виклики, пов'язані з координацією сервісів. Запропоновано рекомендації щодо використання шаблонів проектування та інструментів для ефективного впровадження мікросервісів..

Ключові слова: мікросервіси, архітектура програмного забезпечення, масштабованість, шаблони проектування, розподілені системи.

Abstract

The article explores microservice architecture as a modern approach to software development. It analyzes its advantages, such as flexibility and scalability, and challenges related to service coordination. Recommendations for using design patterns and tools for effective microservice implementation are proposed.

Keywords: microservices, software architecture, scalability, design patterns, distributed systems.

Вступ

Сучасні вимоги до програмного забезпечення, такі як висока доступність, швидке розгортання та можливість масштабування, спричинили зростання популярності мікросервісної архітектури. Цей підхід передбачає розбиття складних додатків на невеликі, незалежні сервіси, які взаємодіють через API. За словами Сема Ньюмана, мікросервіси дозволяють створювати системи, що легко адаптуються до змін [1].

Метою роботи є дослідження особливостей мікросервісної архітектури, аналіз її переваг і недоліків, а також визначення практичних підходів до її реалізації.

Результати дослідження

Мікросервісна архітектура ґрунтується на принципах модульності та автономності. Кожен сервіс відповідає за певну функцію, що дозволяє розробникам зосередитися на конкретних завданнях. Мартін Фаулер підкреслює, що мікросервіси сприяють гнучкості системи за рахунок зменшення залежностей між компонентами [3].

Основні три переваги мікросервісної архітектури це – гнучкість розробки, масштабованість та швидке розгортання. Команди можуть використовувати різні технологічні стеки для окремих сервісів, що сприяє інноваціям [2], система може масштабувати окремі сервіси залежно від навантаження, що економить ресурси [1], а незалежність сервісів дозволяє частіше випускати оновлення без впливу на всю систему [3]. Ці переваги зробили мікросервіси популярними серед таких компаній, як Netflix, Amazon і Spotify, які використовують їх для створення високонавантажених систем.

Проте мікросервіси мають і недоліки. Одним із головних викликів є складність координації між сервісами. Наприклад, забезпечення консистентності даних у розподілених системах вимагає спеціальних підходів, таких як шаблон «Event Sourcing» [2]. Інший виклик — це потреба в надійних інструментах моніторингу, оскільки велика кількість сервісів ускладнює відстеження помилок [1]. Для ефективного впровадження мікросервісів рекомендується використання шаблонів проектування, автоматизація та моніторинг. Шаблони, такі як «API Gateway», спрощують маршрутизацію запитів, а «Circuit Breaker» підвищує стійкість системи до збоїв [2]. Інструменти, такі як Kubernetes і Docker, полегшують розгортання та управління сервісами [1]. Системи, такі як Grafana, дозволяють відстежувати продуктивність і виявляти проблеми [3].

Практичним прикладом є Amazon, який використовує мікросервіси для управління різними частинами своєї платформи, що забезпечує високу надійність і швидке масштабування та Netflix, який

використовує мікросервіси разом із інструментами AWS для забезпечення високої доступності своїх сервісів.

Висновок

Мікросервісна архітектура пропонує значні переваги для розробки сучасного програмного забезпечення, зокрема гнучкість і масштабованість. Водночас вона потребує ретельного планування для подолання викликів, пов’язаних із координацією та моніторингом. Використання шаблонів проектування та автоматизованих інструментів може значно полегшити впровадження мікросервісів. У майбутньому цей підхід, ймовірно, залишатиметься ключовим для створення високонавантажених систем.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Newman, S. Building Microservices: Designing Fine-Grained Systems (2nd ed.). O'Reilly Media, 2021, pp. 22, 67, 203.
2. Richardson, C. Microservices Patterns: With Examples in Java. Manning Publications, 2018, pp. 92, 167, 189.
3. Fowler, M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002, pp. 112, 124, 215.

Шейка Михайло Максимович – студент групи 3ПІ-23б, факультет інформаційних технологій та комп’ютерної інженерії, Вінницький Національний Технічний Університет, Вінниця, e-mail: sheikamihailo2006@gmail.com

Sheika Mykhailo Maksymovych – student of group 3PI-23b, faculty of information technology and computer engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: sheikamihailo2006@gmail.com