УДК 004.422.8

В. А. Шмалюх
О. С. Козловський
І. В. Богач

РОЗРОБКА БЕКЕНД-СИСТЕМ НА БАЗІ МОНОЛІТНОЇ АРХІТЕКТУРИ З ПЕРСПЕКТИВОЮ ПЕРЕХОДУ ДО МІКРОСЕРВІСНОЇ АРХІТЕКТУРИ

Вінницький національний технічний університет

Анотація

У цій статті розглядаються методи розробки бекенд-систем на початкових стадіях із використанням монолітної архітектури з перспективою подальшого переходу до мікросервісної архітектури. Особлива увага приділяється перевагам і недолікам монолітної архітектури, а також основним принципам та методам, які дозволяють забезпечити безболісний перехід до мікросервісів у майбутньому.

Ключові слова: бекенд, монолітна архітектура, мікросервіси, програмна архітектура, масштабованість.

Abstract

This article discusses the methods of developing backend systems at the initial stages using a monolithic architecture with the prospect of transitioning to a microservices architecture. Special attention is given to the advantages and disadvantages of monolithic architecture, as well as the key principles and methods that enable a smooth transition to microservices in the future.

Keywords: backend, monolithic architecture, microservices, software architecture, scalability.

Вступ

Монолітна архітектура є традиційним підходом до розробки програмного забезпечення, що полягає у створенні єдиного цілісного додатку, в якому всі компоненти інтегровані та працюють разом. Незважаючи на певні недоліки, монолітна архітектура має свої переваги, особливо на початкових стадіях розробки проекту. Водночас, мікросервісна архітектура стає дедалі популярнішою завдяки своїй гнучкості, масштабованості та можливості незалежного розвитку окремих компонентів системи. Метою даної роботи є аналіз можливостей розробки бекенду із використанням моноліту з перспективою подальшого переходу до мікросервісів.

Результати дослідження

Монолітна архітектура має кілька важливих переваг, які роблять її привабливою для початкових стадій розробки. По-перше, вона спрощує розробку та розгортання додатків. Всі компоненти розробляються та розгортаються як єдиний додаток, що значно спрощує процеси розробки, тестування та розгортання. По-друге, менша кількість сервісів та компонентів полегшує управління додатком, моніторинг та логування. Це забезпечує просте управління додатком на початкових стадіях розробки. По-третє, відсутність необхідності у міжсервісних комунікаціях дозволяє розробці проходити швидше, що особливо важливо на початкових етапах проекту.

Однак, монолітна архітектура має свої обмеження. По-перше, зростання додатку може призводити до проблем із продуктивністю та масштабованістю, оскільки всі компоненти виконуються в одному середовищі. Це може обмежувати можливість масштабування додатку. По-друге, внесення змін до однієї частини додатку вимагає перерозгортання всього додатку, що може викликати простої та проблеми з доступністю. Це може бути серйозною проблемою для великих додатків. По-третє, зі збільшенням розміру додатку зростає складність його підтримки та внесення змін, що може негативно вплинути на продуктивність команди розробників.

Для забезпечення безболісного переходу до мікросервісної архітектури в майбутньому, необхідно врахувати кілька ключових принципів під час розробки монолітного додатку. Перш за все, необхідно забезпечити модульність розробки додатку. Додаток має складатися з чітко визначених модулів та компонентів, що можуть бути легко відокремлені у самостійні сервіси. Це дозволить забезпечити гнучкість та можливість масштабування додатку у майбутньому. По-друге, необхідно використовувати

чітко визначені API для взаємодії між модулями. Це дозволить легко замінити внутрішні виклики на зовнішні сервіси у майбутньому. По-третє, логіка кожного модуля повинна бути максимально інкапсульована, що дозволить уникнути тісної залежності між модулями. По-четверте, необхідно використовувати підходи до масштабування бази даних, що дозволять розподілити дані між різними сервісами у майбутньому.

Важливим аспектом підготовки до переходу на мікросервіси є чітке визначення API для взаємодії між модулями. Використання добре спроектованих API дозволяє забезпечити стандартизований спосіб комунікації між компонентами системи, що значно спрощує процес інтеграції та забезпечує можливість легкої заміни або модифікації окремих сервісів у майбутньому. Крім того, чіткі API сприяють підвищенню рівня безпеки та надійності системи, оскільки дозволяють краще контролювати доступ до ресурсів та даних.

Інкапсуляція логіки кожного модуля є ще одним важливим принципом, який забезпечує успішний перехід до мікросервісної архітектури. Вона передбачає максимальне відокремлення логіки кожного модуля від інших компонентів системи, що дозволяє уникнути тісних залежностей та забезпечує більшу гнучкість при зміні або оновленні окремих частин додатку. Це також сприяє підвищенню рівня повторного використання коду та зменшує ризик виникнення помилок при внесенні змін.

Особлива увага приділяється масштабуванню бази даних, яке є критичним аспектом при переході до мікросервісної архітектури. Використання підходів до горизонтального масштабування, таких як шардінг або реплікація даних, дозволяє розподілити навантаження між різними сервісами та забезпечити високу продуктивність системи навіть при значному збільшенні обсягів даних. Це також дозволяє забезпечити високу доступність та надійність системи, оскільки відмова одного сервісу не призводить до зупинки всієї системи.

Висновки

Розробка бекенду на початкових стадіях із використанням монолітної архітектури має свої переваги, особливо в частині швидкості розробки та простоти управління додатком. Проте, з часом можуть виникнути проблеми з масштабованістю та підтримкою. Підготовка до можливого переходу на мікросервісну архітектуру дозволяє зберегти гнучкість та забезпечити можливість масштабування системи в майбутньому. Правильне використання модульності, чітких API та інкапсуляції логіки є ключовими елементами успішного переходу від моноліту до мікросервісів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Newman, S. (2019). *Monolith to Microservices: Evolutionary Patterns to Transform Your Monolith*. O'Reilly Media, Inc. [Електронний ресурс] – Режим доступу до ресурсу: https://www.oreilly.com/library/view/monolith-to-microservices/9781492047831/.
2. Richardson, C. (2018). *Microservices Patterns: With examples in Java*. Manning Publications. [Електронний ресурс] – Режим доступу до ресурсу: https://www.manning.com/books/microservices-patterns.
3. Taft, D. K. (2020). *The Enterprise Path to Service Mesh Architectures: Delivering on the Next-Generation Microservices Vision*. O'Reilly Media, Inc. [Електронний ресурс] – Режим доступу до ресурсу: https://www.oreilly.com/library/view/the-enterprise-path/9781492055959/.
4. Wolfe, J. (2017). *Cloud Native: Using Containers, Functions, and Data to Build Next-Generation Applications*. O'Reilly Media, Inc. [Електронний ресурс] – Режим доступу до ресурсу: https://www.oreilly.com/library/view/cloud-native/9781491964641/.
5. Redmond, A., & Wilson, A. (2019). *Software Architecture Patterns for Serverless Systems: The Good, the Bad, and the Ugly*. O'Reilly Media, Inc. [Електронний ресурс] – Режим доступу до ресурсу: https://www.oreilly.com/library/view/software-architecture-patterns/9781492046902/.

Шмалюх Владислав Анатолійович *– студент групи 1ІСТ-23м, кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: zskat02@gmail.com.*

Козловський Олександр Сергійович *– студент групи 1ІСТ-22В, кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, факультет ФІІТА, Вінницький національний технічний університет, Вінниця, e-mail sk@vin.ua.*

Богач Ілона Віталіївна *– к.т.н., доцент кафедри автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м.Вінниця, e-mail: ilona.bogach@gmail.com*

Shmaliukh Vladyslav Anatoliyovych *– student of 1IST-23M group, Department of Automation and Intelligent Information Technologies, Faculty of Intelligent Information Technology and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: zskat02@gmail.com.*

Kozlovskyi Oleksandr S. *- student of the 1IST-22b group, Faculty of Intelligent Information Technologies and Automation, Vinnitsa National Technical University, Vinnitsa, e-mail: sk@vin.ua.*

Bogach Ilona Vitaliivna *– Associate Professor of Automation and Intelligent Information Technologies, Faculty of Computer Systems and Automatics Vinnytsia National Technical University, Vinnytsia, e-mail: ilona.bogach@gmail.com.*