УДК 004.41:004.738

А. М. Обертинський[1]
В. І. Перепелиця[1]
А. О. Пашкевич[1]

ПЕРЕВАГИ МІКРОСЕРВІСНОГО ПІДХОДУ ПРИ ПРОЄКТУВАННІ СУЧАСНИХ ЦИФРОВИХ ОНЛАЙН-БІБЛІОТЕК

[1]Вінницький національний технічний університет

Анотація

У даній роботі досліджується питання проєктування та реалізації архітектури вебплатформи «Онлайн-бібліотека» на основі мікросервісного підходу з використанням технологій .NET (C#) та React (JavaScript). Проведено порівняльний аналіз класичної монолітної та мікросервісної архітектур з точки зору масштабованості, швидкодії обробки запитів до каталогу книг та надійності систем авторизації. Обґрунтовано вибір мікросервісної моделі для організації інтерактивного доступу користувачів до цифрового контенту, що забезпечує високу відмовостійкість системи та гнучкість розгортання окремих функціональних модулів.

Ключові слова: онлайн-бібліотека, мікросервісна архітектура, моноліт, .NET Core, React, авторизація, каталог книг.

Abstract

This paper investigates the design and implementation of the "Online Library" web platform architecture based on a microservice approach using .NET (C#) and React (JavaScript) technologies. A comparative analysis of classic monolithic and microservice architectures is conducted in terms of scalability, query performance for the book catalog, and authorization system reliability. The choice of a microservice model for organizing interactive user access to digital content is justified, ensuring high system fault tolerance and flexibility in deploying individual functional modules.

Keywords: online library, microservice architecture, monolith, .NET Core, React, authorization, book catalog.

Вступ

Ефективна організація доступу до інформаційних ресурсів у сучасних умовах потребує впровадження інтерактивних та високопродуктивних веб-платформ. Роль цифрових онлайн-бібліотек стрімко зростає, оскільки користувачі очікують миттєвого доступу до великих каталогів літератури, персоналізованих рекомендацій та надійних механізмів автентифікації. Дослідження в галузі сучасних вебтехнологій підтверджують, що успіх таких платформ залежить від гнучкості та стійкості їхньої програмної архітектури [1].

Проте процес розробки та супроводу великих бібліотечних систем стикається з низкою технічних викликів: необхідністю підтримки високої швидкості пошуку в каталогах, забезпеченням безпеки персональних даних під час реєстрації та авторизації, а також горизонтальним масштабуванням при пікових навантаженнях [2]. Для вирішення цих завдань традиційно використовувалися монолітні архітектури, які сьогодні поступово поступаються місцем розподіленим мікросервісним рішенням на основі об’єктно-орієнтованого програмування.

Сучасні розробники часто постають перед вибором між монолітом та мікросервісами. Наприклад, монолітна архітектура простіша у початковому розгортанні, але зі зростанням бази книг та кількості користувачів стає громіздкою та складною для оновлення. Натомість мікросервісний підхід дозволяє

розділити систему на незалежні сервіси (сервіс авторизації, сервіс каталогу книг тощо), які комунікують між собою через API, що значно підвищує загальну відмовостійкість застосунку [3].

Метою дослідження є обґрунтування доцільності та порівняльний аналіз монолітної та мікросервісної архітектур для оптимізації процесів розробки вебплатформи «Онлайн-бібліотека» в середовищі Visual Studio 2026.

Результати дослідження

Ефективність функціонування онлайн-бібліотеки безпосередньо залежить від обраної структури компонентів застосунку, яка має гарантувати мінімальний час відгуку клієнтської частини (React) на запити до серверних модулів (C#). У ході дослідження встановлено, що розподіл системи на окремі мікросервіси дозволяє ізолювати критично важливі процеси, такі як реєстрація, авторизація користувачів, а також створення, оновлення та перегляд каталогу видань.

Першою розглянутою моделлю є класична монолітна архітектура, де інтерфейс користувача, логіка обробки даних та доступ до бази даних об’єднані в один великий програмний вузол (рис. 1). У контексті бібліотеки це означає, що збій у модулі генерації статистики чи рецензій може призвести до повної зупинки роботи всього сервісу каталогу або авторизації [4].

Рисунок 1 - Монолітна структура компонентів застосунку

Головною перевагою монолітної архітектури є проста та швидка взаємодія між компонентами, оскільки всі модулі працюють у межах одного процесу. Це спрощує розробку, тестування та розгортання системи. Проте така архітектура має суттєвий недолік – складність масштабування окремих модулів. Наприклад, каталог книг, який отримує найбільше запитів, неможливо масштабувати незалежно від інших частин системи, тому доводиться дублювати весь застосунок [3].

Другим архітектурним рішенням, реалізованим у межах проєкту, є мікросервісна архітектура (рис. 2). У цій моделі система поділяється на окремі незалежні сервіси, кожен з яких відповідає за власну бізнес-функцію та має окрему базу даних. Такий підхід дозволяє незалежно масштабувати найбільш навантажені модулі, підвищує відмовостійкість системи та спрощує командну розробку. Водночас мікросервісна архітектура потребує складнішої інфраструктури та організації взаємодії між сервісами.

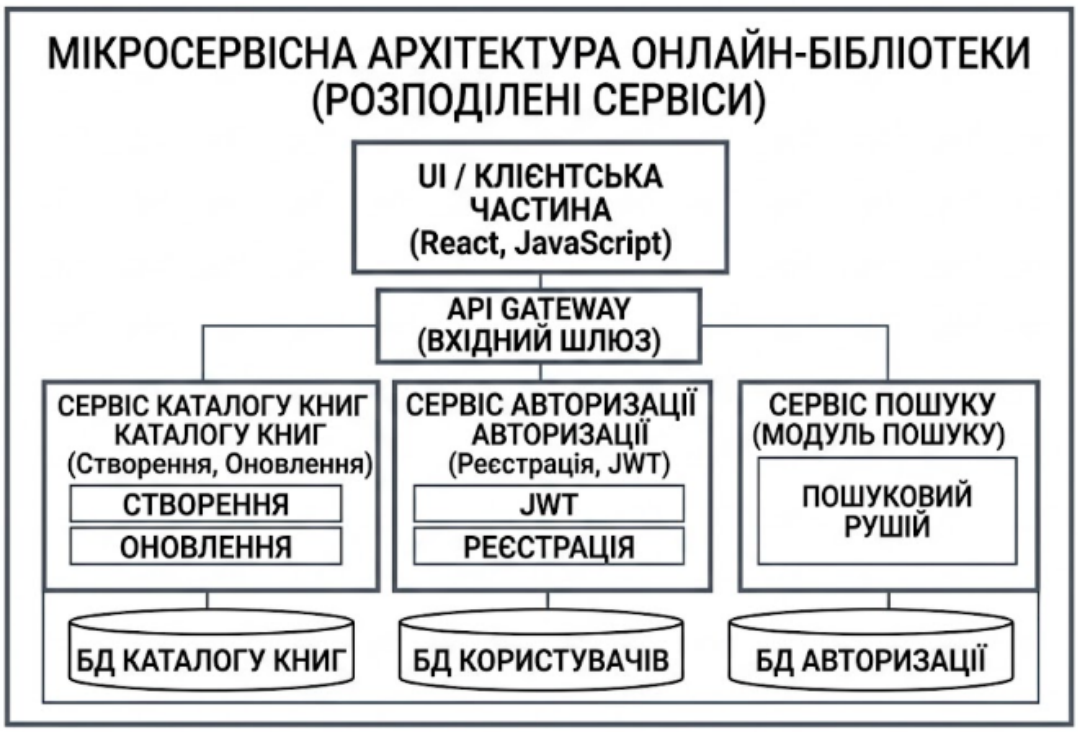

Рисунок 2 - Мікросервісна архітектура вебплатформи «Онлайн-бібліотека»

Використання мікросервісів є доцільним для систем, де критично важливо забезпечити безперебійну роботу інтерфейсу користувача. Наприклад, завдяки відокремленню модуля автентифікації (JWT-токени), процес авторизації виконується незалежно від навантаження на пошуковий рушій каталогу. Клієнтська частина застосунку, побудована на JavaScript (React), взаємодіє з мікросервісами через єдиний шлюз (API Gateway), що спрощує маршрутизацію запитів та підвищує безпеку [5].

Проведений порівняльний аналіз та побудовані послідовні діаграми алгоритмів роботи авторизації дозволили визначити ключові переваги мікросервісного підходу для онлайн-бібліотеки. Основним параметром порівняння стала гнучкість розгортання: мікросервіси дозволяють розробникам оновлювати функціонал створення та редагування картки книги без необхідності перезавантажувати модулі реєстрації користувачів [6].

З точки зору підтримки коду в середовищі Visual Studio 2026, використання принципів об’єктно-орієнтованого програмування в мікросервісах забезпечує високий рівень інкапсуляції та чистоти архітектури. Проте проектування такої системи потребує складнішої логіки для забезпечення консистентності даних між сервісами та налаштування надійної мережевої взаємодії.

Підводячи підсумок порівняння, можна стверджувати, що для сучасних інтерактивних вебплатформ з великою кількістю користувачів пріоритетною є мікросервісна архітектура. Вона дозволяє скоротити час відгуку інтерфейсу та гарантує стабільну роботу системи навіть при масових зверненнях до сховища книг [7].

Висновки

Результати проведені в межах розробки вебплатформи «Онлайн-бібліотека» підтверджують, що вибір мікросервісної архітектури є ключовим фактором для створення масштабованого та стійкого програмного продукту. Порівняльний аналіз показав, що відокремлення модуля авторизації та каталогу книг у самостійні сервіси на C# дозволяє оптимізувати навантаження на сервер та забезпечити безперебійний інтерактивний доступ користувачів до книг через клієнтську частину на React. Впровадження мікросервісного підходу підвищує відмововідмову стійкість системи, унеможливлюючи критичні збої

всього застосунку через помилки в окремих модулях. Перспективи подальших досліджень полягають у розробці механізмів динамічного кешування каталогу книг та інтеграції системи з хмарними сервісами для автоматичного масштабування платформи під час пікових навантажень.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Newman S. Building Microservices: Designing Fine-Grained Systems. – 2nd ed. – O'Reilly Media, 2021. – 614 p.
2. Fowler M. Microservices: a definition of this new architectural term. URL: https://martinfowler.com/articles/microservices.html
3. Трофименко О. Г., Прокоп Ю. В. Веб-технології та проектування веб-систем: навч. посіб. – Одеса: Фенікс, 2019. – 246 с.
4. Richter J. CLR via C#. Pro-Programmer. – 4th ed. – Microsoft Press, 2012. – 896 p.
5. Microsoft Docs. Architectural styles: Microservices. URL: https://learn.microsoft.com/en-us/azure/architecture/guide/architecture-styles/microservices
6. Бублик В. В. Об'єктно-орієнтоване програмування: підручник. – К.: ІТ-книга, 2015. – 632 с.
7. Evans E. Domain-Driven Design: Tackling Complexity in the Heart of Software. – Addison-Wesley, 2003. – 560 p.

Обертинський Андрій Миколайович – студент кафедри комп'ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, e-mail: obertynskiya@gmail.com.

Перепелиця В'ячеслав Ігорович – доктор філософії, асистент кафедри «Комп'ютерних наук», e-mail: pvi_92@ukr.net; Вінницький національний технічний університет, Вінниця.

Пашкевич Анна Олександрівна – студентка кафедри комп'ютерних наук, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, e-mail: annapawkevich@gmail.com

Obertynskyi Andrii Mykolayovych – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: obertynskiya@gmail.com.

Perepelytsia Viacheslav Ihorovych – PhD, assistant of the Department of Computer Science, e-mail: pvi_92@ukr.net; Vinnytsia National Technical University, Vinnytsia.

Pashkevych Anna Oleksandrivna – student of Computer Science Department, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: annapawkevich@gmail.com