УДК 004.42:004.415.2

А. С. Галіброда
В. В. Кабачій

СТВОРЕННЯ REST API ДЛЯ ТАСК-МЕНЕДЖЕРА: ПРАКТИЧНІ РЕЗУЛЬТАТИ ТА АНАЛІЗ

Вінницький національний технічний університет

Анотація

У дослідженні представлено результати розроблення REST API для системи управління завданнями, спрямованої на забезпечення ефективної командної та індивідуальної роботи користувачів. Створення такого програмного продукту потребує застосування сучасних архітектурних підходів, що гарантують масштабованість, надійність, безпечний доступ до даних та можливість подальшого розширення функціональності. У роботі проаналізовано принципи REST-архітектури, зокрема визначення ресурсів, узгодженість використання HTTP-методів та стандартизованість обміну даними у форматі JSON.

Особливу увагу приділено реалізації трирівневої архітектури застосунку, що включає контролери, сервіси та репозиторії, кожен з яких виконує окрему роль у загальній структурі системи. Такий підхід сприяє підвищенню модульності та зручності супроводу. Як основу фреймворку обрано Spring Boot, що дозволило мінімізувати конфігураційні витрати та забезпечити швидку інтеграцію з інструментами Spring Security і Spring Data. Окремо розглянуто питання безпеки, зокрема застосування JWT-токенів для автентифікації та авторизації користувачів без використання серверних сесій.

На основі проведеного аналізу та практичної реалізації побудовано концептуальну архітектуру REST API таск-менеджера, придатну до масштабування, адаптації та інтеграції з веб- або мобільними клієнтськими застосунками. Запропонований підхід забезпечує високу гнучкість системи та її відповідність сучасним вимогам до серверних застосунків.

Ключові слова: REST API, Spring Boot, таск-менеджер, архітектура, JWT, безпека, Java, PostgreSQL.

Abstract

This paper presents the results of developing a REST API for a task management system aimed at supporting effective team-based and individual workflows. Building such a software solution requires the use of modern architectural approaches that ensure scalability, reliability, secure data access, and the ability to extend system functionality in the future. The paper analyzes key principles of REST architecture, including resource identification, consistent use of HTTP methods, and standardized data exchange in JSON format.

Special attention is given to the implementation of a three-tier application architecture consisting of controllers, services, and repositories, each performing a distinct role within the overall system structure. This approach enhances modularity and simplifies system maintenance. Spring Boot was chosen as the primary framework, enabling reduced configuration overhead and seamless integration with Spring Security and Spring Data. The study also addresses security aspects, particularly the use of JWT tokens for user authentication and authorization without relying on server-side sessions.

Based on the conducted analysis and practical implementation, a conceptual architecture of the task manager REST API has been developed. It is suitable for scaling, adaptation, and integration with web or mobile client applications. The proposed approach ensures high system flexibility and compliance with modern requirements for server-side applications.

Keywords: REST API, Spring Boot, task manager, architecture, JWT, security, Java, PostgreSQL.

Вступ

У сучасному програмному забезпеченні веб-застосунки стали невід’ємною частиною як особистої, так і професійної діяльності. Вони охоплюють широкий спектр функцій – від соціального спілкування до управління ресурсами, проєктами та завданнями. Особливої популярності набули системи управління задачами, або таск-менеджери, що допомагають користувачам організовувати власну діяльність, встановлювати дедлайни, контролювати прогрес виконання завдань і координувати командну роботу. Їх застосування підвищує індивідуальну ефективність, а в корпоративному

середовищі сприяє оптимізації бізнес-процесів, покращенню комунікації та забезпеченню прозорості проєктної діяльності.

З ускладненням вимог до програмних рішень зростає потреба у створенні таких серверних інтерфейсів, які здатні масштабуватися, забезпечувати високий рівень безпеки, бути надійними й легко інтегруватися з іншими компонентами цифрової інфраструктури. У цьому контексті особливе значення має розробка REST API – центральної частини застосунку, що відповідає за обробку запитів, доступ до даних і виконання бізнес-логіки. REST-архітектура завдяки уніфікованим підходам до роботи з ресурсами, сумісності з HTTP-протоколом та широкій підтримці в сучасних веб-фреймворках залишається одним із найефективніших рішень для побудови API.

Актуальність теми дослідження полягає у необхідності створення надійного, масштабованого та зручного у використанні серверного API, який може ефективно поєднувати клієнтські інтерфейси – як веб, так і мобільні – із бекендом, що виконує ключову логіку управління задачами. При цьому важливими є не лише якісна реалізація функціональності, а й обґрунтований вибір архітектурних рішень, що визначають стабільність, продуктивність, безпеки та подальшу розширюваність системи.

Метою цієї роботи є дослідження архітектурних підходів до створення REST API для таск-менеджера, аналіз сучасних технологій і патернів, що застосовуються під час реалізації подібних систем, а також обґрунтування вибору фреймворку Spring як технологічної основи для побудови прототипу серверної частини.

Дослідження

Сучасні інформаційні системи управління завданнями потребують високої продуктивності, гнучкості та захищеності, оскільки працюють у багатокористувацькому середовищі та обробляють критичні бізнес-дані. У рамках проведеного дослідження було сформовано методичні та технологічні засади створення REST API для системи управління завданнями, орієнтованого на масштабоване використання та можливість інтеграції з веб- і мобільними клієнтськими застосунками. Основну увагу приділено питанням безпеки, організації зберігання даних, оптимізації швидкодії та забезпечення структурованої взаємодії між елементами системи.

Одним із ключових напрямів дослідження стала **реалізація механізмів автентифікації та авторизації**, оскільки саме вони визначають можливість безпечного доступу до серверних ресурсів. Для цього було проаналізовано сучасні підходи до побудови безстанових систем безпеки та впроваджено модель, засновану на JSON Web Token (JWT). Така модель дозволяє виконувати автентифікацію користувача один раз, після чого клієнт взаємодіє із сервером, передаючи криптографічно підписаний токен у заголовку кожного запиту. Проведені експерименти підтвердили ефективність даного підходу: сервер не зберігає сесійний стан, що забезпечує легку горизонтальну масштабованість та усуває вразливість, пов’язану з викраденням сесій [1]. Крім того, використання BCrypt для хешування паролів знижує ризик компрометації облікових даних у випадку витоку бази даних.

Дослідження також було спрямоване на **формування ефективної моделі зберігання даних**, здатної підтримувати складні взаємозв’язки між користувачами, списками To-Do та задачами. На основі вимог системи побудовано реляційну структуру бази даних PostgreSQL, що забезпечує підтримку транзакційності, цілісності зв’язків та оптимізованого доступу до даних. Для автоматизації взаємодії з базою даних застосовано Spring Data JPA, що дозволяє генерувати SQL-запити на підставі сигнатур методів та усуває потребу в ручному написанні значної частини коду [2]. Це знижує ймовірність помилок, скорочує час розробки та забезпечує прозорість роботи із сховищем.

Важливою складовою дослідження стала **оцінка продуктивності REST API**. Для цього було проведено серію експериментів, спрямованих на вимірювання часу відповіді для ключових операцій: реєстрації користувача, автентифікації, отримання списку завдань, створення та оновлення задач. Аналіз результатів показав, що більшість запитів обробляються в середньому за 13–30 мс, що відповідає вимогам інтерактивних веб-застосунків. Найбільш тривалою операцією виявилася авторизація (≈249 мс), що пояснюється інтенсивними криптографічними операціями при формуванні токена, що підтверджує дані аналогічних досліджень [3]. Усі вимірювання проводилися в однорівневому середовищі без додаткового балансування навантаження, тому очікується, що оптимізація інфраструктури може додатково покращити продуктивність.

Для забезпечення надійності розробленого рішення проведено **модульне та інтеграційне тестування**. Модульні тести із застосуванням JUnit дали змогу перевірити коректність роботи бізнес-логіки у відриві від інфраструктурних компонентів. Інтеграційні ж тести, виконані в середовищі

Postman і Spring Test, дозволили перевірити повні сценарії роботи: реєстрацію користувача, створення To-Do, додавання та зміну завдань, контроль доступу між ролями й обробку помилкових запитів. Результати підтвердили узгодженість роботи контролерів, сервісного шару та системи безпеки, а також коректність обробки одночасних запитів [4].

У дослідженні також проаналізовано **структуру архітектури серверного застосунку**, що ґрунтується на трирівневій моделі, яка вважається стандартом у сучасному проєктуванні вебсервісів. Контролерний рівень відповідає за маршрутизацію та обробку HTTP-запитів, сервісний рівень реалізує бізнес-логіку, а репозиторний рівень – комунікацію з базою даних. Така структуризація забезпечує високу модульність, зменшує зв'язність між компонентами, спрощує тестування та підвищує можливість подальшого масштабування системи. Впровадження DTO-класів дозволило розділити внутрішню доменну модель від формату передачі даних, що підвищує безпеку та захищає внутрішню структуру системи від некоректного втручання з боку клієнтів [5].

Висновок

У дослідженні представлено результати створення REST API для системи управління завданнями, що орієнтована на стабільність, гнучкість та подальше розширення функціональності. Реалізована логіка API охоплює ключові операції з користувачами, списками завдань і задачами, забезпечуючи передбачувану структуру ресурсів і коректну обробку HTTP-запитів.

Особливу увагу приділено захисту даних: механізм автентифікації та авторизації на основі JWT, поєднаний із хешуванням паролів через BCrypt, забезпечує надійний контроль доступу та відповідає сучасним вимогам безпеки веб-сервісів. Для зберігання даних обрано PostgreSQL, що гарантує ефективну роботу з реляційними структурами та можливість обробки складних вибірок.

Модульні та інтеграційні тести підтвердили коректність роботи API і сталість функціоналу за різних сценаріїв використання. Проведений аналіз продуктивності засвідчив стабільний час відповіді навіть за інтенсивної взаємодії з базою даних, що вказує на придатність рішення до масштабування.

Застосування Spring Boot, Spring Security та Spring Data JPA оптимізують процес розробки, зменшують кількість конфігураційного коду та забезпечують надійний фундамент для розширення системи. Запропоноване API може слугувати основою для подальшої побудови повноцінного таск-менеджера з веб-або мобільним інтерфейсом.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Curity. JWT Security Best Practices [Електронний ресурс] / Michał Trojanowski. – Режим доступу: https://curity.io/resources/learn/jwt-best-practices/ – Дата звернення: 15.11.2025.
2. Spring. Spring Data JPA [Електронний ресурс] / Spring. – Режим доступу: https://spring.io/projects/spring-data-jpa – Дата звернення: 15.11.2025
3. Microsoft. Best practices for RESTful web API design [Електронний ресурс] / Microsoft. – Режим доступу: https://learn.microsoft.com/en-us/azure/architecture/best-practices/api-design – Дата звернення: 15.11.2025.
4. Postman. What is Postman? [Електронний ресурс] / Postman. – Режим доступу: https://www.postman.com/product/ – Дата звернення: 15.11.2025.
5. Okta. Data Transfer Object DTO Definition and Usage [Електронний ресурс] / Okta. – Режим доступу: https://www.okta.com/en-gb/identity-101/dto/ – Дата звернення: 15.11.2025.

Галіброда Анатолій Сергійович – студент групи 1ІСТ-24м, кафедра автоматизації та інтелектуальних інформаційних технологій, факультет інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, Вінниця. e-mail: galibrodaa@gmail.com.

Науковий керівник: ***Кабачій Владислав Володимирович*** – к. т. н., доцент кафедри Автоматизації та інтелектуальних інформаційних технологій, Вінницький національний технічний університет, м. Вінниця, e-mail: vkabachiy@gmail.com

Halibroda Anatolii Serhiiovych – student Group 1IST-24m, Department of Automation and Intelligent Information Technologies, Faculty of Intelligent Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, Ukraine, e-mail: galibrodaa@gmail.com

Supervisor: **Kabachii Vladyslav Volodymyrovych**, PhD in Technical Sciences, Associate Professor of the Department of Automation and Intelligent Information Technologies, Vinnytsia National Technical University, Vinnytsia, Ukraine, e-mail: vkabachiy@gmail.com