

РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗОВАНОГО ТЕСТУВАННЯ REST APIS У ПАПЛАЙНАХ CI/CD

Давиденко Микола Олегович

Вінницький національний технічний університет, студент кафедри
програмного забезпечення

Хошаба Олександр Мирославович

Вінницький національний технічний університет, к.т.н., доцент кафедри
програмного забезпечення, pzmag2022@gmail.com

DEVELOPMENT OF A SOFTWARE TOOL FOR AUTOMATED TESTING OF REST APIS IN CI/CD PIPELINES

Davydenko Mykola Olehovych

Vinnitsia National Technical University, student, Department of Software

Khoshaba Oleksandr Myroslavovych

Vinnitsia National Technical University, Ph.D., Associate Professor, Department
of Software, pzmag2022@gmail.com

Integrating Continuous Integration and Continuous Deployment (CI/CD) pipelines has become pivotal for ensuring rapid and reliable software delivery in contemporary software development. Within this framework, the testing of RESTful APIs emerges as a critical component, necessitating automation to maintain efficiency and accuracy. This study presents the

development of a software tool designed to automate the testing of REST APIs within CI/CD processes. The tool aims to streamline the testing workflow, reduce manual intervention, and enhance the detection of potential issues early in the development cycle. Employing scripting techniques and integrating with existing CI/CD platforms, the tool facilitates automated test case generation, execution, and reporting. The implementation leverages industry-standard practices and tools, ensuring compatibility and scalability. Preliminary evaluations indicate a significant improvement in testing efficiency and reduced deployment errors. The findings underscore the importance of automated API testing in modern software development and the role of specialised tools in achieving this objective.

Keywords: REST API, CI/CD, automated testing, software tool, DevOps, test automation, continuous integration.

The increasing complexity of software ecosystems and the rapid pace of software delivery cycles have led to the widespread adoption of Continuous Integration and Continuous Deployment (CI/CD) practices. In this environment, maintaining the reliability and quality of software applications is contingent upon robust and repeatable testing mechanisms. RESTful APIs are pivotal as the glue that connects distributed services among the various components in a typical software architecture. Ensuring the correctness and performance of these APIs has become an indispensable part of modern DevOps workflows.

While still prevalent in smaller or legacy systems, manual API testing is no longer sufficient in fast-paced development pipelines. The shift towards agile methodologies and microservice architectures further underscores the need for test automation. A particular challenge arises in aligning API test automation with the CI/CD process to ensure that automated pipelines immediately

validate codebase changes. This paper presents the development of a dedicated software tool to automate the testing of REST APIs within CI/CD workflows, designed to be lightweight, extensible, and integrable with existing DevOps toolchains.

The tool was designed to satisfy several key requirements: seamless integration with popular CI/CD platforms such as Jenkins, GitLab CI, and GitHub Actions; support for automated generation of test suites from OpenAPI specifications; the ability to handle various authentication mechanisms; detailed logging and reporting capabilities; and support for negative and boundary testing. The development process followed an iterative, test-driven methodology. Python was selected as the primary language due to its strong API interaction and testing ecosystem, particularly libraries such as requests, unittest, and pytest.

A central architectural component of the tool is a test orchestrator module, responsible for parsing API specifications, dynamically generating test cases, and invoking them by pipeline events. Another key feature is the environment abstraction layer, which allows tests to be executed against different deployment environments (e.g., staging and production mirrors) without modification of the test logic. Test results are exported in JUnit-compatible formats to ensure compatibility with existing CI dashboards and reporting tools.

Experiments were conducted using real-world APIs in a controlled CI/CD pipeline setup to evaluate the tool's effectiveness. The APIs were selected based on varying complexity, authentication types (OAuth2, API keys), and operation counts. Baseline metrics were collected using traditional manual testing scripts. The results demonstrated a significant reduction in test execution time, improvement in test coverage, and faster feedback loops to developers. Integration into CI/CD pipelines immediately detects breaking changes and misconfigurations that would otherwise go unnoticed until later stages of development or deployment.

The evaluation also highlighted the tool's scalability. In projects with frequent commits and large numbers of pull requests, the tool could execute hundreds of tests per hour with minimal

configuration overhead. Furthermore, automated testing led to consistent results free from human error and interpretation bias. Developers noted the improvement in confidence when merging code, as regressions were caught early and automatically.

Several challenges were addressed during the tool's development. These included managing dynamic test data, mocking third-party services, and dealing with flaky tests arising from network instability or stateful APIs. These were mitigated by including data fixtures, retry logic, and state cleanup routines. Emphasis was also placed on ensuring that the tool did not slow down CI pipelines, which was achieved through parallel execution and lightweight containers.

In the broader DevOps and software reliability engineering context, the proposed tool contributes to the growing infrastructure-as-code and test-as-code paradigms ecosystem. It embodies the principle of shift-left testing by enabling developers to detect API-related issues early in the development lifecycle. Moreover, the approach supports continuous testing, where every code change is automatically subjected to regression and performance evaluation without human intervention.

Future work involves extending the tool's capabilities to support GraphQL APIs, incorporating machine learning techniques for intelligent test case generation, and enhancing the observability of test results through integration with monitoring and alerting systems. Plans also include releasing the tool as an open-source project to foster community adoption and feedback.

In conclusion, automating REST API testing within CI/CD pipelines is crucial to achieving reliable, scalable, and maintainable software delivery practices. The proposed tool provides a comprehensive solution that reduces testing time, improves test accuracy, and seamlessly integrates with DevOps workflows. Its application in real-world scenarios has demonstrated measurable benefits in productivity, code quality, and deployment confidence.

References

1. Kim, M., Stennett, T., Shah, D., Sinha, S., & Orso, A. (2023). Leveraging Large Language Models to Improve REST API Testing. arXiv preprint arXiv:2312.00894. <https://arxiv.org/abs/2312.00894>
2. Bardakci, T., Demeyer, S., & Beyazit, M. (2025). Test Amplification for REST APIs Using "Out-of-the-box" Large Language Models. arXiv preprint arXiv:2503.10306. <https://arxiv.org/abs/2503.10306>
3. Stallenberg, D., Olsthoorn, M., & Panichella, A. (2021). Improving Test Case Generation for REST APIs Through Hierarchical Clustering. arXiv preprint arXiv:2109.06655. <https://arxiv.org/abs/2109.06655>
4. Karlsson, S., Causevic, A., & Sundmark, D. (2019). QuickREST: Property-based Test Generation of OpenAPI-Described RESTful APIs. arXiv preprint arXiv:1912.09686. <https://arxiv.org/abs/1912.09686>
5. Kim, M., Xin, Q., Sinha, S., & Orso, A. (2022). Automated Test Generation for REST APIs: No Time to Rest Yet. arXiv preprint arXiv:2204.08348. <https://arxiv.org/abs/2204.08348>