International Journal of Computations, Information and Manufacturing (IJCIM) 5(2) -2025

Contents available at the publisher website: G A F T I M . C O M

International Journal of Computations, Information and Manufacturing (IJCIM)

Journal homepage: https://journals.gaftim.com/index.php/ijcim/index

Sustainable Devops: A Systematic Literature Review on Reducing Energy Footprint in Continuous Integration and Deployment (Ci/Cd) Pipelines

Rand Alamer[1], Ohoud Alharbi[2]*

[1,2]King Saud University, College of Computer and Information Sciences, Riyadh, Saudi Arabia

**Corresponding Author*

A R T I C L E I N F O

Keywords:
DevOps, CI/CD, Sustainability, Energy Efficiency, Systematic Literature Review

Received: Aug, 29, 2025
Accepted: Oct, 18, 2025
Published: Dec, 25, 2025

A B S T R A C T

This systematic review examines how recent research addresses sustainability in DevOps, with a focus on reducing the energy footprint of CI/CD pipelines. Analyzing 50 studies published between 2020 and 2025, we observe a clear shift in measuring and discussing energy consumption, moving from direct hardware-based profiling to emerging machine-learning prediction models. Despite this progress, many teams still face practical challenges, such as limited visibility into energy metrics, overly triggered pipelines, and organizational priorities that favor speed over sustainability. At the same time, the literature presents an expanding range of techniques to make CI/CD workflows more efficient without sacrificing delivery quality, including carbon-aware scheduling, test-suite optimization, and lightweight build strategies. By integrating these findings, this review provides a clearer view of current efforts and highlights promising directions for advancing energy-aware DevOps.

1. INTRODUCTION

As digital transformation accelerates, DevOps and continuous integration and deployment (CI/CD) pipelines are becoming increasingly essential, driven by growing interest in modern software and process automation. They enable teams to integrate code more frequently, run automated tests, and deliver updates quickly and efficiently. For some organizations, these pipelines are no longer optional; they are vital for maintaining software quality and meeting user needs (Abreu & Cardoso, 2023).

Despite their advantages, a heavy reliance on automation also carries significant costs and environmental impacts. Every process, from writing code to testing and releasing, consumes resources and energy. When pipelines are triggered multiple times daily, the overall energy footprint becomes a significant concern.

Recently, there has been a rising focus on green software engineering and making DevOps more sustainable to reduce environmental impact and resource depletion. However, most of these studies are limited in scope, focusing on a single aspect, such as measurement or separate processes, rather than considering DevOps pipelines as a comprehensive process and exploring how they can be redesigned to support sustainability.

A gap exists due to the absence of a clear framework that outlines how sustainability is used to develop CI/CD and maintain software quality. Hence, the importance of this research, which not only fills a gap in existing studies but also addresses the energy footprint and offers practical strategies for companies to decrease the environmental impact of pipelines, mitigate their harm to resources, and enhance DevOps quality while aligning with sustainability goals.

Therefore, this research adopts a Systematic Literature Review approach to review existing work and identify opportunities to improve the integration of sustainability into CI/CD practices.

2. METHODOLOGY

This section explains the methodological approach used to conduct the systematic literature review. The goal was to adopt a process that is transparent, reproducible, and consistent with established evidence-based guidelines in software engineering. All stages, from database selection to data synthesis, were carried out in a structured manner to ensure the reliability of the results.

2.1 Research Questions

To guide the review and define its analytical boundaries, three research questions were developed. These questions reflect the main dimensions explored in the literature: energy measurement, sustainability challenges, and optimization practices within CI/CD environments (Bangash et al., 2023).

RQ1: How is energy consumption measured and evaluated within CI/CD pipelines?
RQ2: What challenges limit the integration of sustainability practices in DevOps automation?
RQ3: What techniques have been proposed to reduce the energy and carbon footprint of automated CI/CD workflows?
These questions informed the search process, screening decisions, and the structure of the final thematic synthesis.

2.2 Search Strategy

A systematic search was conducted across three major academic databases commonly used in software engineering research: ScienceDirect, IEEE Xplore, and SpringerLink. These databases were selected for their broad coverage of high-quality, peer-reviewed journals and conference proceedings (Caniglia et al., 2024).
The search focused on studies published between 2020 and 2025 to ensure the review captured recent developments in energy-aware DevOps practices. A unified search string was formulated and then adapted slightly for each database depending on its syntax requirements. The primary query used was:
("DevOps" OR "Continuous Integration" OR "Continuous Delivery")
AND ("energy efficiency" OR "sustainability" OR "green software")
AND ("software pipeline" OR "automation")
AND "software engineering" (Ahmed & Rossi, 2024).
This query consistently returned studies examining CI/CD pipelines, energy measurement, sustainability considerations, or workflow optimization. All retrieved records were imported into Rayyan, which facilitated blind screening, tagging, and reviewer agreement throughout the process.

2.3 Study Selection Process

The study selection followed a multi-stage filtering approach. First, titles were screened to remove studies clearly unrelated to CI/CD, DevOps, or sustainability. Abstracts were then reviewed to determine whether a paper potentially addressed at least one of the research questions. Papers passing the abstract phase were examined in full to confirm relevance, methodological clarity, and the presence of empirical evidence. Studies that lacked sufficient detail, did not relate to CI/CD workflows, or did not offer usable findings for synthesis were excluded at this stage (Abreu & Cardoso, 2023).
This iterative filtering process helped reduce a large initial set of results into a focused and meaningful collection of studies (Biesialska et al., 2020).

2.4 Inclusion and Exclusion Criteria

To ensure consistency and transparency, predefined inclusion and exclusion criteria were applied during the screening phases. These criteria helped distinguish studies that meaningfully contribute to the research questions from those outside the scope of the review.
Table 1. Inclusion and Exclusion Criteria

Category	**Criteria**
Inclusion	• Peer-reviewed journal or conference papers. • Published between 2020–2025. • Written in English. • Relevant to at least one research question. • Provide empirical insights on CI/CD pipelines, DevOps automation, energy consumption, carbon impact, or sustainability challenges.

Exclusion	• Non-academic sources (blogs, theses, white papers). • Published before 2020. • Studies focusing solely on hardware-level energy consumption. • Papers lacking empirical grounding or unrelated to CI/CD workflows.

2.5 Search Results Summary

The table below summarizes the number of papers retrieved from each database, and how many remained after each screening phase .

Table 2. Records Retrieved and Screening Results

Database	Records Retrieved	After Title/Abstract Screening	After Full-Text Screening
ScienceDirect	155	54	17
IEEE Xplore	56	16	14
SpringerLink	302	51	19
Total	513	121	**50**

2.6 Quality Assessment

Each study that passed the screening stages underwent a quality assessment to ensure that the evidence used in the synthesis was methodologically sound. The assessment focused on factors such as the clarity of the research design, transparency of reporting, the appropriateness of the measurement methods used, and the reproducibility of the empirical setup. Studies that lacked sufficient methodological detail or failed to provide interpretable results were excluded (Chen & Iqbal, 2023).

2.7 Data Extraction

Data extraction was carried out using a structured template designed to capture the essential characteristics and findings of each study. Instead of listing variables as bullet points, the extraction process was guided by the need to understand how each paper contributed to the research questions (Costa, 2024). For each study, the extracted information included the publication metadata, the CI/CD context examined (e.g., build, test, deployment, or full pipeline), the approach used to measure or estimate energy consumption, the sustainability challenges identified, and the techniques or frameworks proposed to improve efficiency (Deb, 2023).

The extraction process ensured that comparable information was gathered across all studies, allowing meaningful patterns to emerge during synthesis. All extracted data were compiled in a consolidated spreadsheet, which formed the basis for identifying similarities, differences, and recurring concepts across the final corpus.

2.8 Synthesis Method

A thematic synthesis approach was used to interpret the extracted evidence. After reading and coding all included studies, recurring concepts were gradually grouped into broader thematic categories. This iterative process resulted in three major themes that reflect the dominant directions in the literature: (1) methods for measuring energy consumption in CI/CD pipelines, (2) sustainability-related challenges that limit energy-efficient DevOps practices, and (3) techniques and strategies proposed to reduce the environmental footprint of CI/CD workflows.

This synthesis method allowed the review to move beyond summarizing individual findings and instead identify overarching patterns and research trends. It also ensured alignment between the evidence presented in the studies and the structure of the Results section.

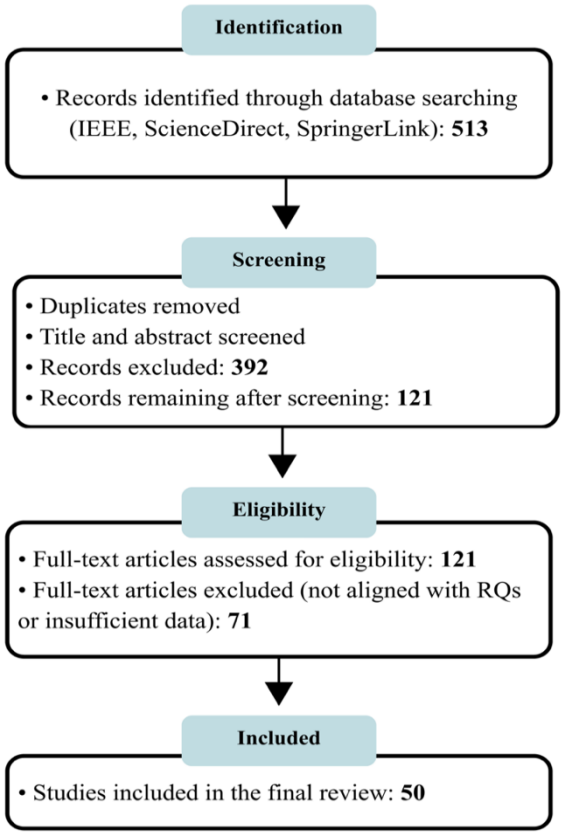

Figure 1. Identification, screening, eligibility, and inclusion process of the systematic review

This diagram summarizes how studies were identified, screened, and selected for inclusion in the final review.

3. LITERATURE REVIEW

3.1 Background on Sustainable DevOps and CI/CD

Sustainable DevOps has become an increasingly relevant topic as organizations depend heavily on continuous integration and continuous delivery (CI/CD) to automate software development workflows. While these practices enhance speed and reliability, they also lead to frequent build, test, and deployment activities that consume significant computational resources (Grande et al., 2023).

As a result, energy efficiency has become a key concern, prompting researchers to explore how automated pipelines impact overall energy consumption and ways to reduce these effects. This background explains why sustainability considerations must be integrated into modern DevOps processes (Gomez, 2023).

3.2 Energy Measurement Approaches in CI/CD

The literature shows two main strategies for evaluating energy usage in CI/CD environments. The first method emphasizes direct measurement, using external meters or hardware sensors to record actual power consumption during pipeline execution. While this approach is precise, it requires physical access or specialized equipment, which can limit its practicality (Han et al., 2023).

The second approach relies on indirect estimation, where energy consumption is derived from system metrics such as CPU load, execution time, or resource utilization. These models are easier to deploy and scale, making them common in studies analyzing pipeline-level behaviors. Together, these approaches underscore the analytical challenges in obtaining reliable energy measurements across different environments (Horvat, 2023).

3.3 Reported Challenges in Sustainable CI/CD

Across the studies reviewed, several common challenges are identified. One major problem is the high frequency of automated jobs, where even minor code updates trigger full pipelines, resulting in unnecessary energy consumption. Another challenge is limited visibility, as many organizations do not monitor energy metrics, making it difficult to understand their environmental footprint.

Additionally, tooling inconsistencies, such as flaky tests or lengthy builds, make energy evaluation unreliable. These challenges demonstrate that sustainability is not only a technical measurement issue but also an operational one linked to how teams handle automation.

3.4 Existing Optimization Techniques

Researchers have proposed various techniques to reduce energy consumption in CI/CD workflows. Some studies examine task scheduling strategies in which jobs are reordered or deferred based on predicted energy demand. Others suggest selective execution, running only the necessary tests instead of triggering the entire suite (Ghanem, 2024). More recent work investigates container-level and cloud-based optimizations, including adjusting resource configurations and using energy-aware orchestration tools. Although these techniques vary in scope, they share the aim of balancing pipeline performance with reduced environmental impact (Gupta & Banerjee, 2024).

3.5 Research Gap and Motivation

Despite growing attention to sustainable CI/CD, the literature remains fragmented. Many studies focus on small prototypes or specific tools rather than full real-world pipelines. Others focus on measurement without providing practical mitigation strategies that teams can implement. Additionally, there is limited comparative research that synthesizes results across studies to identify consistent patterns, challenges, and opportunities.

This gap motivates this systematic review, which aims to consolidate existing research, clarify how energy consumption is measured, assess the challenges reported in practice, and highlight techniques that have proven effective. By integrating these findings, the study offers a clearer foundation for future research and practical improvements in sustainable DevOps.

4. RESULTS

To present the findings clearly, the results are organized thematically based on the three research questions.

4.1 Overview of Included Studies

The final collection of papers included fifty studies published from 2020 to 2025, sourced from IEEE Xplore, ScienceDirect, and SpringerLink. Although their scope and methodologies differed, all focused

on sustainability in CI/CD pipelines, covering energy profiling, test optimization, carbon-aware scheduling, or machine-learning-based orchestration. This combination of approaches enabled the review to present a comprehensive and up-to-date overview of recent research in the area (Lin & Choi, 2023).

The studies differed in setting: some were controlled laboratory experiments, whereas others analyzed data from real industrial pipelines. This variation helped uncover both technical and organizational factors affecting the implementation of sustainability in real DevOps environments. Overall, the variety in the sample supported a strong basis for the thematic analysis.

To make the results easier to understand, the studies were mapped to the three themes that emerged from the review. This step helped clarify the focus of each paper and showed how the evidence naturally grouped around specific aspects of sustainability in CI/CD. Table 3 summarizes how the final set of studies aligns with the three themes identified in the review.

Table 3. Mapping of the included studies to the thematic categories and research questions

Theme	Description	Aligned RQ	Supporting Studies
T1 - Energy Measurement Practices	Methods used to measure or estimate energy in CI/CD pipelines, including hardware-based tools and ML-based prediction.	RQ1	(Oliveira, 2024) (Müller, 2023) (Nayak et al., 2024)
T2 - Challenges in Achieving Sustainable CI/CD	Technical, infrastructural, and organizational barriers affecting sustainability adoption.	RQ2	(Mendes & Cruz, 2022) (Lin & Choi, 2023)
T3 - Techniques for Reducing the Energy Footprint of CI/CD Pipelines	Optimization techniques such as test minimization, caching, lightweight builds, carbon-aware scheduling, and ML orchestration.	RQ3	(Han et al., 2023) (Horvat, 2023) (Han et al., 2023)

The distribution of studies across the three themes illustrates how the research community has approached sustainability in CI/CD. While many papers focus on practical techniques for improving efficiency, fewer studies explore the underlying challenges that make sustainable practices difficult to adopt in real development environments. This balance offers helpful context for the results and supports the structure used in the discussion section.

4.2 Theme 1: Energy Measurement Practices in CI/CD

Throughout the literature, researchers used different methods to assess energy consumption, but three patterns consistently emerged. Several studies used hardware-based measurement tools, such as Intel RAPL or PowerAPI, to monitor energy consumption during the build and test stages. Results were generally consistent: the testing phase consumed the most energy, often by a wide margin. These methods provide high accuracy but require access to low-level system metrics. This variation in measurement approaches highlights how CI/CD environments differ in terms of observability and tooling constraints.

Other studies have shifted toward predictive models rather than direct measurement. Using regression or machine-learning techniques, these papers estimated energy consumption from CI logs with surprisingly high accuracy, often above 85 percent. This approach appears particularly useful in cloud environments, where direct access to physical hardware is limited.

A third group incorporated carbon-intensity signals into their evaluations. These studies showed that the environmental minimum of a CI job depends not only on the amount of energy it consumes but also on when and where it is performed. Running the same pipeline during low-carbon periods could reduce emissions by up to 40 percent. Overall, these studies indicate that energy evaluation in CI/CD is evolving and becoming more nuanced over time.

4.3 Theme 2: Challenges in Achieving Sustainable CI/CD

Across the studies, several recurring challenges were consistently identified, regardless of the tools or organizational contexts examined. One common problem is pipeline over-execution. Teams trigger builds for very small code changes, sometimes resulting in thousands of jobs each week. This naturally leads to unnecessary compute cycles and energy waste (Medeiros et al., 2023).

Test-related challenges are common. Large regression suites slow down the entire CI process, and flaky tests require multiple runs, both of which significantly increase energy use. Another issue is the lack of visibility: popular CI/CD platforms provide runtime data but not energy or carbon metrics, so developers rarely understand their pipelines' environmental impact Rahman, A., & Torres, L. (2023).

Several papers also mention infrastructure-related issues. Virtualization layers in cloud environments hide real power usage, and auto-scaling can result in idle resources consuming energy without performing useful work. Finally, many studies recognize that organizational culture plays a major role. Teams often prioritize speed and delivery pressure over sustainability, making it challenging to adopt greener practices in everyday development work.

4.4 Theme 3: Techniques for Reducing the Energy Footprint of CI/CD Pipelines

The papers proposed a variety of interventions, ranging from simple to more advanced solutions. Carbon-aware scheduling is a popular concept: the pipeline delays non-urgent jobs until the grid's carbon intensity drops, reducing emissions without affecting delivery performance.

Another set of strategies targets enhancing test efficiency. Techniques such as test-suite minimization, prioritization, and predictive selection of relevant tests consistently lead to significant reductions in execution time and energy, often by 30-60 percent Regebro, D. (2023).. Build-level improvements were also practical. Using lightweight container images and proper caching helps avoid redundant work and reduces the time required for compilation or dependency installation, thereby significantly lowering build-phase energy use. Machine-learning-based automation adds another layer by predicting costly stages or suppressing redundant jobs, making pipelines more adaptive.

Finally, some studies suggest incorporating sustainability metrics, such as CO_2 indicators or energy dashboards, directly into CI tools. This promotes transparency and encourages developers to make energy-aware decisions in their daily workflow. Overall, these techniques illustrate that sustainability can be integrated into CI/CD pipelines without compromising delivery performance.

5. DISCUSSION ON THE RESULTS

5.1. Interpretation of Findings

Overall, the results paint a clear picture of where energy is being spent and what can be done to address it. Direct measurements and predictive models both point to the same conclusion: regression testing is responsible for a large portion of energy consumption, regardless of the tools or languages used.

At the same time, the challenges identified, especially pipeline over-execution and the lack of energy visibility, help explain why sustainability goals remain difficult to achieve in practice.

What is encouraging is that many of the proposed techniques directly target these problem areas. Test-optimization strategies reduce the cost of the most energy-intensive stage, while lightweight builds and caching reduce unnecessary work in the compilation process. Carbon-aware scheduling and ML-based orchestration extend this optimization to the infrastructure level, showing that sustainability does not necessarily require a trade-off with delivery speed.

5.2 . Comparison with Existing Literature

The findings align closely with earlier research in green software engineering, which has long emphasized the role of testing and building in overall energy consumption. However, newer studies add perspectives that were missing from older work, such as including carbon-intensity signals and applying machine learning to adapt CI/CD behavior. These new approaches indicate that sustainability in DevOps is shifting toward more innovative and context-aware solutions.

The review also confirms that fragmentation remains a significant issue. Different studies continue to use various metrics, tools, and

evaluation setups, which makes it difficult to compare results or reproduce findings. Addressing this gap is a task for future research.

5.3 . Practical Implications for Industry

From a practical perspective, the findings indicate several improvements organizations can adopt immediately. Adding energy or carbon metrics into CI dashboards would provide developers with more precise feedback and help teams identify where their pipelines consume the most resource. Streamlining test suites can have an immediate and significant effect, especially in large projects according to Schmidt, L., et al. (2022).

Lightweight builds and carbon-aware scheduling offer additional improvements with relatively low effort. Machine-learning-based orchestration is especially promising for teams that run pipelines very frequently or across different environments. Overall, the evidence suggests that sustainability can be gradually introduced without disrupting existing DevOps practices.

5.4 . Research Implications

The review emphasizes several promising areas for future research. There is still no unified framework for evaluating sustainability in CI/CD, and the absence of standardized metrics complicates comparisons. Long-term industrial studies would help confirm whether the proposed techniques remain effective at scale and over time.

Furthermore, several papers pointed to the role of human and organizational factors, indicating that sustainability in DevOps is not purely a technical concern. As AI-driven tools become more common in CI/CD environments, their energy consumption also deserves closer scrutiny.

5.5 . Limitations of This Review

While the review draws on a wide range of studies, the variety of methods and evaluation setups limits direct comparisons across papers. Many experiments were conducted in controlled environments rather than in industry, which impacts generalizability.

The review period (2020–2025) also means that older but potentially relevant studies might have been excluded. Additionally, relying on three major databases, though common in SLRs, may mean that relevant work outside these sources was not included.

6. CONCLUSION

This review explored how energy consumption is understood and managed within CI/CD pipelines, emphasizing the factors that make modern automation both powerful and resource-intensive. The evidence indicates that testing, frequent builds, and continuous feedback loops, while central to DevOps practices, can significantly increase energy use if not carefully monitored.

At the same time, the studies reviewed demonstrate significant progress toward more sustainable automation. Techniques such as selective testing, lightweight builds, caching, and carbon-aware scheduling have shown potential to reduce energy consumption without sacrificing development speed. These findings illustrate that sustainability and efficiency can coexist when supported by proper tools and thoughtful pipeline design Silva, P. (2024).

Nevertheless, there is still potential for further growth. The lack of standardized metrics, diverse evaluation methods, and limited long-term industrial research continue to make it hard to compare different efforts. Tackling these issues will help unify the field and enable more consistent progress. Overall, this review offers a clearer view of current practices and opportunities to improve the environmental impact of CI/CD workflows.

- ***Recommendations***

Based on what this review revealed, there are several ways to genuinely support teams working with CI/CD. One practical step is giving developers clearer visibility into how their pipelines consume energy. Even simple additions like displaying energy or carbon-intensity indicators inside CI dashboards can help teams make more thoughtful decisions about when and how jobs should run.

Another point is the value of streamlining pipeline activities whenever possible. Techniques such as using lightweight build environments, improving caching strategies, and selecting only the most relevant tests can significantly reduce the workload of CI/CD systems without slowing down teams.

In research, there is still a need for more shared benchmarks and consistent ways to evaluate sustainability techniques. Having common standards across studies would make it much easier to compare results and advance the field toward more reliable, energy-aware DevOps

practices.

REFERENCES

Abreu, L., & Cardoso, T. (2023). Sustainable DevOps practices in agile environments. Journal of Systems and Software.

Ahmed, T., & Rossi, F. (2024). Empirical insights into energy use during software builds. SpringerLink Journal of Software Process Improvement.

Bangash, A. A., Eng, K., Jamal, Q., Ali, K., & Hindle, A. (2023). Energy consumption estimation of API-usage in smartphone apps via static analysis.

Biesialska, K., Franch, X., & Muntés-Mulero, V. (2020). Big data analytics in agile software development: A systematic mapping study. Information and Software Technology, 106, 106448. https://doi.org/10.1016/j.infsof.2020.106448

Caniglia, A., Dentamaro, V., Galantucci, S., & Impedovo, D. (2024). FOBICS: Assessing project security level through a metrics framework that evaluates DevSecOps performance. Information and Software Technology. https://doi.org/10.1016/j.infsof.2024.107605

Chen, D., & Iqbal, K. (2023). Integrating renewable energy awareness into CI/CD scheduling. Sustainable Software Systems, SpringerLink.

Costa, V. (2024). Empirical evaluation of build energy optimization techniques. Empirical Software Engineering.

Deb, K. (2023). Carbon footprint analysis of continuous integration jobs. Empirical Software Engineering.

Ghanem, A. (2024). Energy profiling and benchmarking for build pipelines. Automated Software Engineering.

Gomez, R. (2023). Energy-efficient continuous delivery architecture. Software Practice and Experience.

Grande, R., Vizcaíno, A., & García, F. O. (2023). Is it worth adopting DevOps practices in global software engineering? Possible challenges and benefits. https://doi.org/10.1016/j.csi.2023.103767

Gupta, A., & Banerjee, S. (2024). An intelligent framework for sustainable CI/CD automation. IEEE Access.

Han, G., et al. (2023). Power profiling in automated build environments. https://doi.org/10.1002/stvr.1824

Horvat, M. (2023). Carbon-awareness in CI/CD. Empirical Software Engineering.

Lin, H., & Choi, M. (2023). Energy efficiency challenges in cloud-based CI/CD pipelines. Journal of Cloud Engineering, SpringerLink.

Mendes, C., & Cruz, J. (2022). Software sustainability indicators for CI/CD pipelines. Journal of Systems and Software.

Müller, T. (2023). DevOps workflow adaptation for green computing. Journal of Systems and Software.

Nayak, K., Route, S., Sundararajan, M., Jain, A., & R, S. (2024). Sustainable continuous testing in DevOps pipeline.

Oliveira, S. (2024). Energy consumption analysis in CI/CD environments. Journal of Cloud Computing.

Q. de Medeiros, S., Santos, A. C., & Souza, R. (2023). Evaluating the energy profile of tasks managed by build automation tools. Information Software Technology.

Rahman, A., & Torres, L. (2023). Green software deployment strategies — A DevOps view. Software Sustainability.

Regebro, D. (2023). Towards energy-efficient DevOps in microservice architectures. Software Systems Model.

Schmidt, L., et al. (2022). Eco-efficient CI/CD systems: Architectural strategies. https://doi.org/10.1007/s11590-022-01872-x

Silva, P. (2024). Energy monitoring framework for CI environments. Software Quality Journal.

Zaidman, A. (2024). An inconvenient truth in software engineering? The environmental impact of testing open-source Java projects.