National College of Ireland

Navigating AI Integration - From Theory to Practice: A Scoping Review of AI Integration Frameworks for DevOps

MSc Research Project
MSc in Cloud Computing

Salmaan Ali
Student ID: 22187448

School of Computing
National College of Ireland

Supervisor: Dr. Diego Lugones

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Salmaan Ali
Student ID:	22187448
Programme:	MSc in Cloud Computing
Year:	2024
Module:	MSc Research Project
Supervisor:	Dr. Diego Lugones
Submission Due Date:	16/09/2024
Project Title:	Navigating AI Integration - From Theory to Practice: A Scoping Review of AI Integration Frameworks for DevOps
Word Count:	XXX
Page Count:	25

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	16th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Navigating AI Integration - From Theory to Practice: A Scoping Review of AI Integration Frameworks for DevOps

Salmaan Ali
22187448

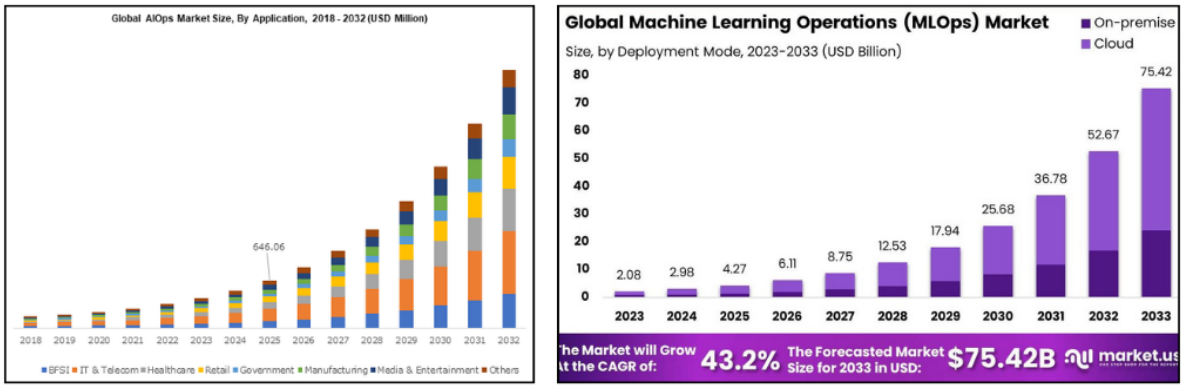
1 Abstract

The integration of Artificial Intelligence (AI) into DevOps workflows has dawned a paradigm shift that promises improvements across all stages of a Continuous Integration/ Continuous Deployment (CI/CD) routine. Despite the potential benefits, this unification faces challenges like integration complexity, data privacy concerns, ensuring the reliability of AI-generated outputs, bias mitigation, and the need for new skill sets. This scoping review examines the current state of AI integration in DevOps by synthesising findings from, after applying inclusion and exclusion criteria, 28 recent research articles, offering a panoramic view of AI-enhanced DevOps. Employing the Preferred Reporting Items for Systematic Reviews and Meta-Analyses extension for Scoping Reviews (PRISMA-ScR) guidelines, this article analyses how AI techniques such as Large Language Models, Machine Learning and, Natural Language Processing are transforming various DevOps stages such as development, testing, security, and monitoring. This analysis reveals that AI integration can significantly boost productivity, with some studies reporting up to 65% more requirements implemented and a 70% reduction in development time. Additionally, this study provides a practical implementation guide and a phased roadmap for organisations looking to harness AI in their DevOps workflows. This scoping review aims to bridge the gap between academic research and industry practice by identifying critical knowledge gaps and proposing future research directions to fully unlock the potential of AI in DevOps.

2 Introduction

The integration of Artificial Intelligence (AI) into the DevOps methodologies has transformed software development. The integration with AI seeks to address the ever-increasing complexity of software development and the never-ending demand for greater efficiency in development processes. As an increasing number of organisations are starting to embrace AI solutions, such as Large Language Models (LLMs), in order to optimize their workflows and enhance their throughput for product delivery, the synergy between AI and DevOps emerges as a critical area of investigation. The study into this integration becomes necessary largely due to the integration of AI into DevOps being fraught with challenges that may include AI expertise gap within DevOps teams, security and ethical

concerns, as well as resistance to transitioning into AI-optimised environments due to potential disruptions.



(a) (b)

Figure 1: (a) Global market size of AIOps, by GMinsights (2022)
(b) Global Marlet Size of MLOps, by Market.us (2023)

This presents a research problem that needs to be investigated: a deep dive into currently available frameworks that facilitate smooth integration of AI in DevOps by allowing organisations to overcome existing barriers in adoption and implementation. The significance of this research stems from its potential to address these challenges by exploring how integration frameworks contribute to the practical implementation of AI within DevOps environments and what factors should be considered in their adoption and effective utilisation. Moreover, despite recognising the potential of AI in software development, many businesses struggle to use it constructively due to a lack of technical know-how, as stated by Battina (2021). This research aims to mitigate these issues by providing a comprehensive review of available AI integration frameworks, identifying successful frameworks, knowledge gaps, and best practices. The research is motivated by the following question: How do integration frameworks contribute to the practical implementation of AI integration in DevOps, and what factors should be considered in their adoption and use? This study analyses and synthesises current literature on the topic, aiming to offer insights that guide organisations looking into AI-integrated DevOps practices, bridge knowledge gaps, and propose actionable strategies for effective utilisation of AI/ML in the DevOps paradigm.

The integration of Artificial Intelligence (AI) and Machine Learning (ML) into DevOps has proven to revitalise the software development by introducing novel operational methodologies that, as shown in Figure 1, are among the most trending workflows for software development. With the advent of social media and consumer electronics adoption being greater than ever before, organisations are feeling the necessity for agile, intelligent and efficient development pipelines to deliver solutions that meet this explosive growth of data. The advent of AIOps and MLOps as software development disciplines, aims to bridge the gap between fast-paced development cycles and the data-driven insights provided by AI/ML. They address the critical need for cohesive workflows that are not only scalable, reliable and secure within production environments, but also accelerate the deployment of solutions. Through an analysis of the key works that have paved the way for the integration of AI/ML within the DevOps framework, this literature review explores this emerging field and aims to recommend solutions to improve security, optimize

performance, and spur innovation in software development and deployment processes.

3 Related Works

Several recent studies have discussed the integration of AI into DevOps workflows and offered their unique perspective and insights. This section breaks down three research articles that complement and contrast this research.

Mboweni et al. (2022) conducted a systematic review of peer reviewed conference papers and journal articles, published between 2015 to 2022, about Machine Learning DevOps. Their aim was to identify the state of the art and identify knowledge gaps in the literature. They examined over 60 studies that dealt with ML-DevOps, also referred to as MLOps.

They found that there is a lack of studies that primarily investigate MLOps. Most studies focused on clarifying what is meant by MLOps and while there is commonality in MLOps definitions, a lack of standardisation leads to misconception and ambiguity. They also concluded that there is no common understanding among scholars and experts on how MLOps should be implemented and regulated across industries.

This systematic review aligns with my study in recognising the growing importance of integrating AI/ML into DevOps workflows. However, the focus of the review conducted by Mboweni et al. (2022) is specifically ML-DevOps, whereas my study presents a wider perspective, encompassing various AI techniques beyond just machine learning. Additionally, their work goes further and provides concrete implementation strategies for organisations.

The paper by Ali and Puri (2024) explores the integration AI techniques in DevOps practices and the advantages that follow, such as enhanced efficiency, automation, and decision-making capabilities. The authors provide a comprehensive review of existing literature and present a case study outlining the practical application of AI in a real-world DevOps environment.

They concluded that AI can greatly enhance various aspects of DevOps, including CI/CD pipelines, incident management and resource optimisation. They underline the significance of addressing ethical concerns and challenges in AI integration. The authors propose a phased approach to implementing AI in DevOps workflows.

While this study shares common ground with my review in its approach. However, my work provides a more detailed exploration of various AI techniques and their application across different stages of DevOps processes and the tools and frameworks that can be implemented.

Lu et al. (2022) present a research roadmap, produced by conducting a Systematic Literature Review (SLR), regarding the responsible use of AI. Their work focuses on setting up multi-level governance and development processes that establish responsible AI practices. They promote building "responsible-AI-by-design" into AI systems.

They propose a "three-pronged" approach: governance perspective, process perspective, and system perspective and emphasise the importance of ethical considerations in AI development and deployment. The authors outline research hurdles in domains such as requirements engineering for responsible AI and continuous monitoring of AI outcomes.

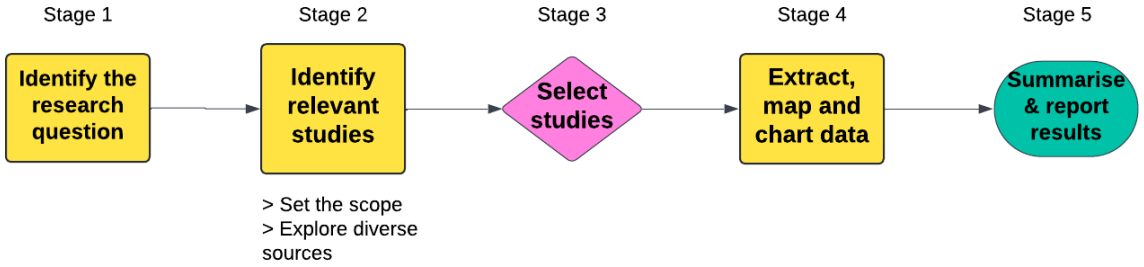


Figure 2: Stages of Scoping review

Lu et al. (2022)'s SLR that assessed 159 studies, focuses mainly on responsible AI in software development, which is in contrast to my study that targets the integration of AI in DevOps workflows. While there are significant overlaps in the ethical aspects discussed, my work provides a more detailed insight into the practical implementation of AI in DevOps workflows, while also addressing similar concerns about responsible AI development.

These studies clearly showcase the growing interest and importance of AI in DevOps. Mboweni et al. (2022) highlighted the need for standardisation in ML-DevOps (or MLOps) while Ali and Puri (2024) provided some practical understanding on AI in DevOps workflows. Lu et al. (2022) presented a broad perspective on the responsible use of AI in software development. Although these studies offer valuable insights, this research builds upon them and other studies by providing a more holistic view of the current scenario of AI integration into DevOps. This research discusses a wider range of AI techniques, highlights the benefits and challenges, and also offers detailed implementation strategies.

4 Methodology

This study undertakes a scoping review to collect and analyse currently available literature on the convergence of Artificial Intelligence and DevOps. This study follows the Preferred Reporting Items for Systematic Reviews and Meta-Analyses extension for Scoping Reviews (PRISMA-ScR) guidelines was developed by an expert panel of 24 members and 2 research leads to help readers as well as researchers develop a deeper understanding of relevant terminology, key items, and core concepts to report for scoping reviews, as stated by Andrea C. Tricco and Sharon E. Straus (2018). Various stages of the scoping review methodology are represented in Figure 2.

4.1 Research Method

To gather said studies, multiple search queries are made through various databases. These search queries consist of different combinations of keywords related to the research question. The articles discovered in the search results underwent title screening, and the relevant ones were downloaded. The downloaded articles are screened by reading the abstract, introduction and conclusions, so that irrelevant and duplicate articles can be

S.No.	Query
1	("AI" OR "Artificial Intelligence" OR "Machine Learning" OR "Deep Learning" OR "Large Language Models" OR "LLMs") AND ("DevOps" OR "DevOps workflow" OR "software development" OR "continuous integration" OR "continuous delivery")
2	("AI" OR "Artificial Intelligence" OR "Machine Learning" OR "Deep Learning" OR "Large Language Models" OR "LLMs") AND ("DevOps" OR "DevOps workflow") AND ("framework" OR "frameworks")
3	("AI frameworks" OR "Machine Learning frameworks" OR "Deep Learning frameworks" OR "LLMs frameworks") AND ("DevOps" OR "DevOps workflow")
4	("Large Language Models" OR "LLMs") AND ("DevOps" OR "DevOps workflow" OR "software development" OR "continuous integration" OR "continuous delivery")
5	("AI" OR "Artificial Intelligence" OR "Machine Learning" OR "Deep Learning" OR "LLMs") AND ("DevOps" OR "DevOps workflow") AND ("automation" OR "optimization" OR "efficiency")
6	("AI-driven" OR "Machine Learning-driven" OR "Deep Learning-driven" OR "LLMs-driven") AND ("DevOps" OR "DevOps workflow" OR "software development")

Table 1: Search queries

removed. The articles remaining after this step underwent Steps 4 and 5 as shown in Figure 2. The entire research method is summarised in Figure 3.

4.2 Search Queries and Databases

The search queries listed in Table 1 were run through four databases, namely Google Scholar, IEEE Xplore, ACM Digital Library and Springer. While Google Scholar was selected for its broad, interdisciplinary coverage, and its ability to index a wide range of academic publications, IEEE Xplore and ACM Digital library are the cornerstone databases here, since one is a leading resource for technical literature in engineering and technology while the other is a premier resource for computing literature, respectively. Springer is also a valuable addition for its extensive collection of scientific journals and conference proceedings.

4.3 Selection and Rejection Criteria

The following criteria were used to govern whether an article is included in the study or not.

Selection Criteria:

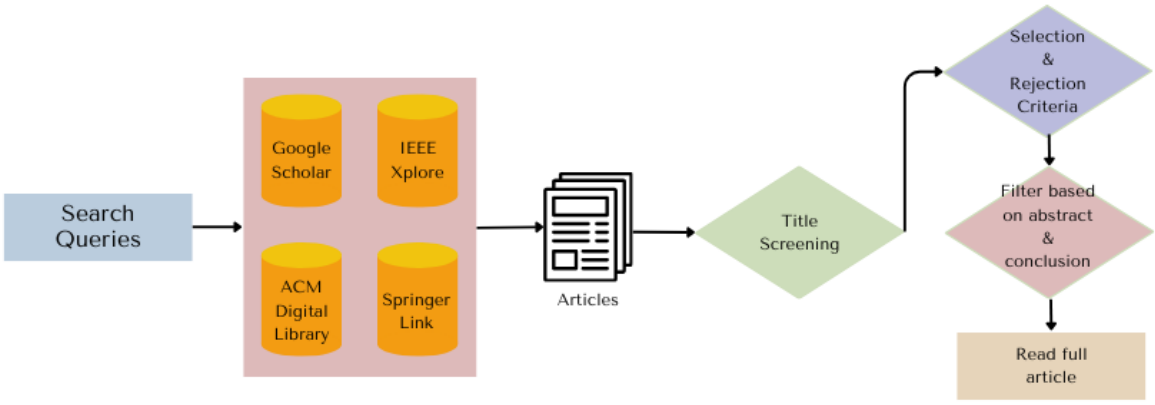


Figure 3: Research Methodology

- **Relevance:** The article explicitly talks about AI (including machine learning, deep learning, LLMs, etc.) integration into DevOps workflows or processes (planning, coding, building, testing, release, deployment, monitoring).
- **Publication Date:** Articles published in the last 3 years to ensure up-to-date research.
- **Language:** Articles published in English.
- **Availability:** Access to full-text of the article.

Rejection Criteria:

- **Relevance:** The article does not discuss AI, DevOps or the convergence of both these disciplines.
- **Publication Date:** Articles that were published before 2021.
- **Language:** Articles that are published in languages other than English.
- **Availability:** Articles that cannot be accessed in their entirety.
- Duplicate articles are also rejected.

Number of articles before and after applying the selection and rejection criteria are listed in Table 2.

5 Review Analysis and Results

5.1 Introduction

This sections provides the findings from 28 recent research articles that were section based on the selection and rejection criteria. It discusses the various AI techniques that are employed to aid DevOps practices, their impact on DevOps as well as the challenges and limitations that come with it.

Database	Articles Downloaded	Articles Selected
Google Scholar	15	7
IEEE Xplore	29	8
ACM Digital Library	21	9
Springer	18	4

Table 2: Number of articles before and after applying the selection and rejection criteria

5.2 AI Techniques in DevOps

5.2.1 Large Language Models (LLMs)

Many authors, such as Peng (2023), Sauvola et al. (2024), Wang and Chen (2023), Petrović (2023a), suggest that Large Language Models (LLMs) are currently the dominant AI technique being applied to DevOps, particularly for tasks such as code generation and code comprehension. Several studies have explored the usage of LLMs like GPT-3 and GPT-4 in different stages of the software development life cycle and the primary benefits of employing LLMs into DevOps workflows, as discussed by Peng (2023), include automated code generation, faster code comprehension and enhanced developer productivity.

However, Peng (2023) has also highlighted several challenges that plague the use of LLMs. These include the inability of LLMs to comprehend the overarching development process of complex systems, limitations in preciseness and abstract thinking capabilities and the potential for "hallucinations" i.e. generating non-existent information.

Furthermore, Liventsev et al. (2023) conducted a comprehensive study on autonomous program synthesis and repair utilising LLMs. Their study introduces SEIDR(Synthesize, Execute, Instruct, Debug and Rank), a novel framework to generate and debug code in Python and C++ using Codex-edit and GPT-3 models. They evaluated their framework using the Program Synthesis Benchmark 2 (PSB2), wherein SEIDR outperforms the PushGP baseline, solving 19 out of 25 problems with fewer than 1000 program executions. Their work contributes significantly to the field of generative programming, showcasing the potential of LLMs for efficient program synthesis.

5.2.2 Machine Learning and Predictive Analytics

Beyond LLMs, various machine learning techniques have also been implemented to DevOps workflows.

Traditional machine learning algorithms have demonstrated excellent ability in predicting effectiveness of DevOps implementations. A study by Kumar et al. (2023) utilised algorithms such as Support Vector Machines (SVMs), Artificial Neural Networks (ANNs) and Random Forest (RF) to analyse the impact of software practices on efficacy of DevOps workflows. Their study concluded That SVM with RBF kernel is the best suited algorithm for predicting DevOps effectiveness. Organisations can employ this data driven approach to evaluate and improve their DevOps workflows, utilising more targeted training and personalised resource allocation.

Anomaly detection and predictive analytics have surfaced as effective tools for identifying obstacles in DevOps workflows. Mohammed et al. (2024) proposed an AI-driven framework for Continuous Integration-Continuous Deployment that incorporates machine

learning models for predicting potential issue before they impact production systems. Similarly, Hausi A. et al. (2023) explored the use of LLMs and AIOps for continuous operations. Their approach proposes Reactive-Predictive-Proactive AIOps in combination with Digital Twins (virtual representations of specific entities, usually physical systems), highlighting how these technologies can enable more efficient and preemptive management of IT systems.

To enhance security measures, machine learning techniques have been implemented to automate threat detection, a crucial component of DevSecOps. Camacho (2024) proposed the integration of AI/ML into DevSecOps routines to automate threat detection and thereby enhance security measure throughout the software development life cycle. Their study underlined the capability of these techniques to reduce security incidents and improve overall system resilience.

Furthermore, the need for a consistent AI model for DevOps is attempted by Lyu et al. (2021), proposing a criteria for assessing consistency while also providing guidelines to derive interpretations that can be relied on. Their study not only highlights the advantages of implementing AI in DevOps, but also ensuring that its decision-making processes are transparent and trustworthy.

5.2.3 Natural Language Processing (NLP)

Natural Language Processing (NLP) techniques bridge the gap between human language and machine-readable instructions. The adoption of NLP techniques for DevOps has shown promising results in requirements engineering, code understanding, and testing phases.

Requirements Analysis and User Story Generation: NLP has proved to be crucial in automating and improving the requirements gathering process. Nasiri and Lahmer (2024) proposed an AI-driven methodology for refining and clustering agile requirements using NLP techniques. They employed BERT-based sentence transformers to calculate similarity between user stories, which enabled automated detection of duplicate requirements.

Dos Santos et al. (2024) explored AI-driven user story generation using both N-gram models and GPT-3. Their study was a comparison, generating context-based user stories, between the two models. They found that N-gram models showed higher semantic sensitivity while GPT-3 was able to produce more comprehensive user stories. Organisations may benefit from this application of NLP as it could significantly accelerate the requirements gathering process and improve consistency in user story writing across teams.

Code Documentation and Explanation: NLP techniques have also been employed to enhance code comprehension and documentation. MacNeil et al. (2023) conducted a study on using code explanations generated by LLMs in a web software development book. Their research aimed to understand the feasibility of AI-generated explanations to enhance student learning in software development courses. The study concluded that while students focused on all types of explanations, line-by-line explanations were visited most frequently (even though they were deemed least useful for learning). This demonstrates that employing NLP for code explanation could be useful for improving developer education and may introduce new training methods. It could also be utilised for enhancing code understanding, which could aid in maintainability.

Automated Test Case Generation: Multiple AI techniques have been employed to automate test case ganeration. NLP has demonstrated excellence in automating the process of test case generation, a crucial aspect of quality assurance in DevOps. Bayrı and Demirel (2023) investigated the impact of LLMs on software testing methodologies, particularly focusing on employing ChatGPT for generating unit tests. Their research concluded that ChatGPT can effectively generate executable test cases for a large percentage of Java classes, achieving considerable code coverage. This application of NLP could potentially reduce the manual effort and significantly speed us the testing process.

Meanwhile, Mehmood et al. (2023) explored the use of GitHub Copilot, an AI-powered programming assistant, for generating test cases. They studied how the quality of test cases generated by GitHub Copilot compared to manually written ones. Their findings stated that GitHub Copilot possessed the potential to diversify the range of test cases when provided with precise prompts.

5.3 Impact of AI on DevOps

5.3.1 Benefits of AI integration in DevOps

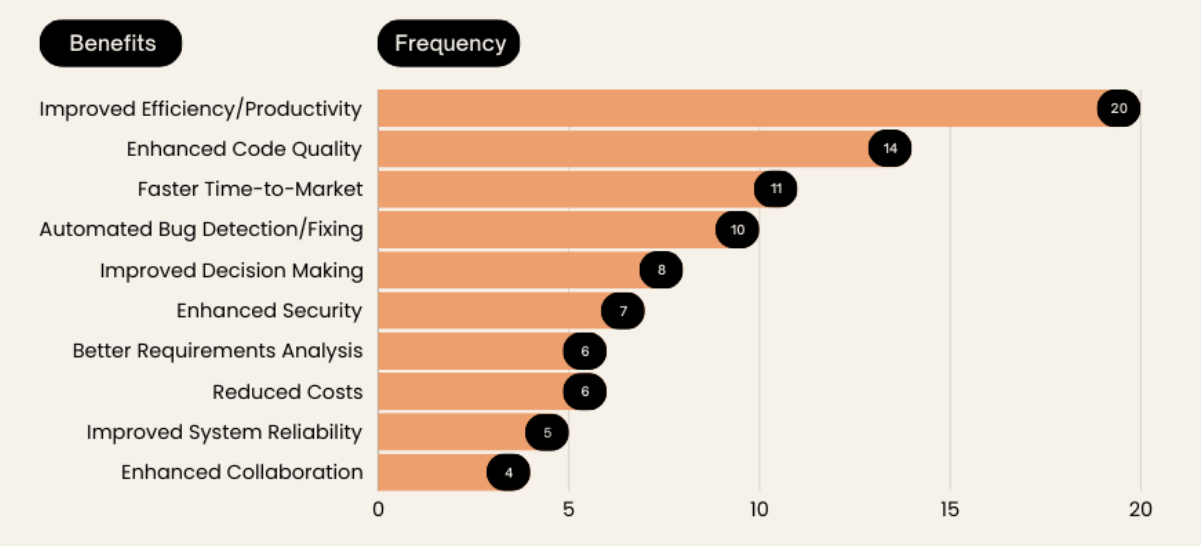


Figure 4: Frequency of mentioned benefits across reviewed articles

The integration of AI into DevOps practices offers a variety of benefits across various stages of the Software Development Life Cycle (SDLC). These benefits are the key towards popularity of AI adoption for DevOps workflows.

1. **Improved Efficiency and Productivity:** Multiple authors like Peng (2023), Sauvola et al. (2024) and Weber et al. (2024) reported significant improvements in software development speed and efficiency as a result of AI integration. Specifically, Weber et al. (2024) concluded that employing AI increased the number of requirements implemented by 65%. This substantial productivity gain was attributed to AI's coding tasks automation and code suggestion capabilities.

2. **Enhanced Code Quality and Consistency:** As discussed by Sauvola et al. (2024) and Du et al. (2024), employing AI-powered tools can improve code quality

as they can suggest best practices, identify potential bugs and ensure consistency in coding standards across different teams. Du et al. (2024) evaluated LLMs in class-level code generation and found that models such as GPT-3.5 and GPT-4 outperformed other models in generating correct and consistent class-level code.

3. **Faster Time-to-Market:** Studies by Sauvola et al. (2024) and Vemuri et al. (2024) have shown that AI integration can significantly reduce he time required to release new features and products to the market by automating various aspects of a CI/CD pipeline. Mohammed et al. (2024) also proposed an AI-driven framework for CI/CD, powered by predictive analytics that reduces the production overall cost while also accelerating the software delivery process.

4. **Proactive Issue Detection and Resolution:** Mohammed et al. (2024) and Hausi A. et al. (2023) concluded that AI-driven predictive analytics can assist in identifying potential issues before they hit the production systems, enabling developers to do proactive maintenance and hence reduce downtime. Hausi A. et al. (2023) discussed that LLMs can be employed for proactive operations which could offer faster incident detection and resolution.

5. **Improved Decision Making:** Data-driven insights provided by AI can be leveraged for decision making processes such as feature prioritization and resource allocation, as concluded by Kumar et al. (2023) and Vemuri et al. (2024). The study by Kumar et al. (2023) also demonstrated that machine learning models, specifically Support Vector Machines with RBF kernel are capable of predicting DevOps implementation effectiveness, potentially leading to better decision-making in DevOps adoption.

6. **Enhanced Security Practices:** AI integration into DevOps offers significant enhancement to security practices. Camacho (2024) discussed the how AI/ML practices have enhanced DevSecOps, offering automated thread detection and proactive identification of vulnerabilities across the SDLC.

7. **Automated Test Case Generation:** AI has proven to be of tremendous assistance in automating test case generation, reducing manual effort while improving test coverage. Bayrı and Demirel (2023) found that LLMs such as GPT-3.5 Turbo are capable of successfully generating executable test cases for a large percentage of Java classes.

These benefits can be mapped to certain stages of a DevOps pipeline, as illustrated in Figure 5. This demonstrates that even today, AI possess the capability to enhance the complete cycle of a DevOps workflow. As AI technologies continue to evolve, these benefits are likely to become even more pronounced, further transforming the landscape of software development and operations.

5.3.2 Challenges and Limitations

Despite the numerous benefits, AI integration in DevOps is filled with significant challenges and limitations that might be faced by organisations that are current and potential future adopters of AI frameworks for their workflows.

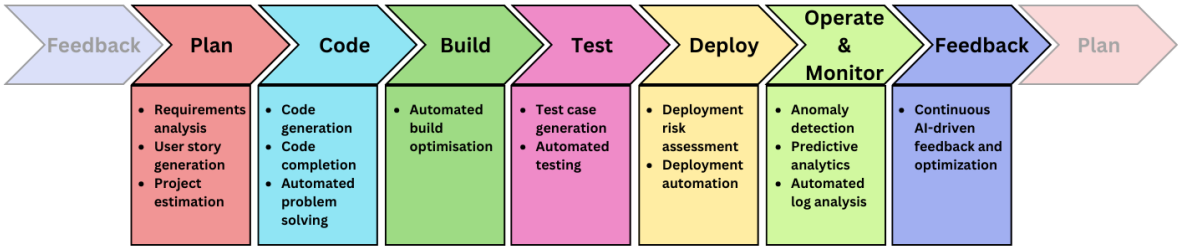


Figure 5: AI integration points in the DevOps pipeline

1. **Ethical Concerns and Bias:** Sauvola et al. (2024) emphasise that the use of AI in software development raises ethical concerns, especially regarding potential biases in AI generated code or decision making processes.They also highlight that AI models may amplify existing biases present in their training data. This could potentially lead to biased outcomes in software development or unfair decision making in DevOps processes.

2. **Data Privacy and Security:** Camacho (2024) argue that since AI systems require access to large amounts of data, ensuring the privacy and security of this data is a major concern. They point out that while AI-powered DevSecOps tools enhance security in many ways, these tools might also introduce new vulnerabilities. For instance, adversarial attacks could potentially manipulate AI models into making incorrect security assessments.

3. **Integration Complexity:** As discussed by Sauvola et al. (2024) and Rangaraju et al. (2023), integrating AI tools into existing DevOps workflows can be complex, requiring major changes to processes and even infrastructure. Rangaraju et al. (2023) further elaborate that AI-driven DevSecOps strategies often require sophisticated infrastructure for data collection, model training, and deployment. This can be especially challenging for organisations with legacy systems and/or limited resources.

4. **Skill Gap:** For effective integration of AI in DevOps, new skills and knowledge are required, which may not be readily available in many organisations, argue Sauvola et al. (2024) and Battina (2021). Further stated by Sauvola et al. (2024), the successful integration of AI into DevOps workflow requires a blend of software engineering, data science, and domain-specific expertise. Battina (2021) also discusses the evolving landscape of IT and DevOps in the United States, highlighting the growing demand for professionals with expertise in both DevOps and AI. Organisations may find this skill gap to be a significant barrier for AI adoption in DevOps practices.

5. **Over-reliance on AI:** Peng (2023) and Alenezi et al. (2022) discuss that over-relying on AI-generated solutions comes with a risk of reduction in human problem solving skills or overlooking important context that AI might miss. Peng (2023) raises concern towards the potential for "hallucinations" i.e. the generation of non-existent data by LLMs, which could introduce major errors if not managed. Alenezi et al. (2022) also states that loss of critical thinking and problem solving skills among development teams due to over-reliance on AI could be particularly

problematic when dealing with novel or complex issues that AI systems may not be sufficiently equipped to handle.

6. **Model Accuracy and Reliability:** In the context of AI in DevOps, another noteworthy challenge is to ensure the accuracy of AI models. Lyu et al. (2021) address this issue in their study on consistent interpretation of AIOps models. They emphasise that the randomness in model training can significantly impact interpretation consistency, and that only high-performing models ($AUC > 0.75$) yield more consistent interpretations.

7. **Continuous Learning and Adaptation:** DevOps requirement change rapidly and so do their environments. This raises another challenge, which is the need for continuous learning and adaptation of AI models. Hausi A. et al. (2023) discuss this in the context of proactive continuous operations, emphasising the need for AI systems to adapt to evolving IT environments and the new types of needs or problems that come within them.

In conclusion, organisations must carefully navigate these challenges to realise the full potential of AI integration into DevOps. Addressing ethical concerns, ensuring data privacy and security, managing integration complexity, bridging the skill gap, balancing AI assistance with human expertise, and ensuring model accuracy and adaptability are necessary steps towards successfully leveraging AI in DevOps workflows.

5.3.3 Impact on DevOps Stages

AI integration affects various stages of the DevOps pipeline. Table3 summarises the impact of AI on different DevOps stages based on the above discussion.

5.3.4 Readiness of AI integration in DevOps

Figure 6 represents the readiness of AI integration in DevOps via the CALMS[1] (Culture, Automation, Lean, Measurement, and Sharing) framework. The readiness level is represented on a scale of 1 to 5, where 1 is low readiness and 5 is high readiness, based on the insights from the reviewed literature.

1. **Culture : 1.5/5 - Lower readiness due to hurdles like skill gaps, resistance to AI adoption, ethical concerns, distrust and, need for embracing AI-human collaboration.**

    - On one hand Sauvola et al. (2024) discuss the lack and therefore the need for new skills and knowledge, on the other, Battina (2021) highlight the evolving skill requirement in the US IT industry due to the increasing AI adoption across organisations.
    - Peng (2023) raised ethical issues that might arise due to over-reliance on AI-generated code. Ethical issues are a major reason for organisations to distrust the use of AI.

[1] https://www.atlassian.com/devops/frameworks/calms-framework

DevOps Stage	AI Impact	Key Benefits	Challenges
Planning	• Automated requirements analysis • User story generation	• Improved efficiency • Better requirement clarity	Recording undocumented requirements
Development	• Code generation • Automated problem solving	• Increased productivity • Faster development cycles	Ensuring code quality and security
Testing	• Automated test case generation • Predictive analytics for bug detection	• Improved test coverage • Faster bug identification	Generating relevant test cases for complex scenarios
Deployment	• Automated deployment pipelines • Predictive analytics for deployment issues	• Faster, more reliable deployments • Proactive issue resolution	Ensuring consistency across environments
Monitoring	• Anomaly detection • Automated log analysis	• Faster incident detection • Improved system reliability	Handling false positives/negatives
Feedback	• Automated user feedback analysis • Predictive analytics for feature prioritization	• Better understanding of user needs • Data-driven decision making	Balancing automated insights with human expertise

Table 3: Impact of AI on DevOps Stages

- AI integration currently lacks the balance between AI capabilities and human expertise, as emphasised by Peng (2023), although work on exploring models for collaborative development between AI and humans is underway in various studies such as one by Cai et al. (2024).

2. **Automation : 5/5 - High readiness due to major strides in AI-driven automation across various DevOps stages such as code generation for development and testing.**

   - Sauvola et al. (2024), Mohammed et al. (2024) and, Vemuri et al. (2024) highlight the capabilities of AI to automate code generation, continuous integration and deployment, not just locally but also in cloud environments
   - LLMs are capable of advanced code production for development and testing as discussed by Peng (2023), Wang and Chen (2023), Bayrı and Demirel (2023) and Mehmood et al. (2023).

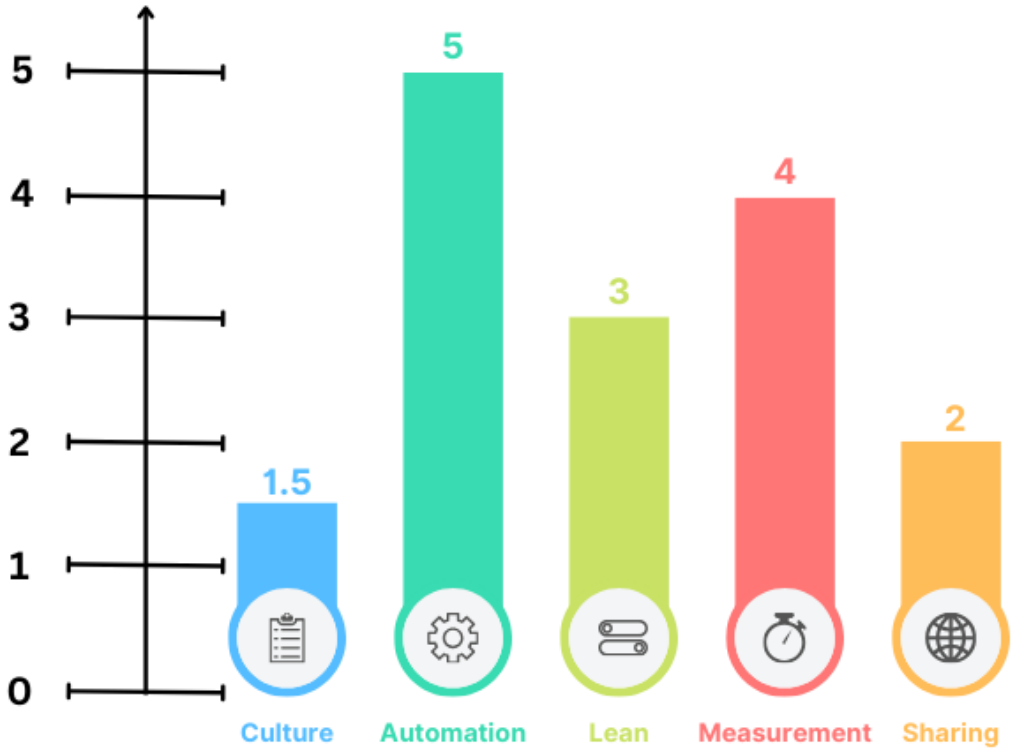


Figure 6: Bar chart representing the readiness of AI integration in DevOps using CALMS framework

3. **Lean : 3/5 - Moderate readiness due to potential for AI to increase efficiency and reduce redundancy in development processes, yet being afflicted by challenges concerning with integration itself.**

    - Contribution of AI in enhancing efficiency in software development processes and significantly improving productivity through assisted programming are discussed by Sauvola et al. (2024) and Weber et al. (2024), respectively.
    - Vemuri et al. (2024) explore how AI can streamline DevOps processes and eliminate bottlenecks, particularly in cloud environments.
    - As discussed by Sauvola et al. (2024) and Rangaraju et al. (2023), integrating AI into existing DevOps workflows is a tedious and complex process, requiring changes to pipelines and even infrastructures, which usually causes temporary damage to overall efficiency.

4. **Measurement : 4/5 - Strong readiness in AI-powered analytics and monitoring, with scope for enhancement in AI-driven metrics**

    - Mohammed et al. (2024) and Hausi A. et al. (2023) both discuss how AI applications offer advanced solutions for system monitoring and predictive analytics. They also highlight the use of AI for proactive issue identification and resolution.
    - Peng (2023) state that there is room for improvement in AI's ability to understand the development process on a wider scale that might lead to incongruous

code across applications. This indicates the need for better code quality metrics that monitor application over long term, which is also discussed by Huang et al. (2024).

5. **Sharing : 2/5 - Lower readiness in terms of knowledge sharing and collaborative measures surrounding use of AI in DevOps.**

    - Lyu et al. (2021) lay out the difficulties faced on achieving consistent interpretations of AI model decisions and how the lack of this comprehension makes it difficult sharing insights across teams.
    - Similarly, Kolltveit and Li (2023) highlight the hurdles faced in operationalising and explaining ML models in production environments.
    - The study on automated assistants that aid in coding and documentation done by Leung and Murphy (2023) suggests the potential for greater collaboration through AI tools. Petrović (2023b) explored the use of ChatGPT for supporting software development and indicated the possibility of improved sharing and collaboration through AI.

The CALMS framework, when employed to evaluate the impact of AI in DevOps, reveals not only the high readiness of technical aspects, but also hints at challenges in organisational and cultural preparedness. Using this analysis, future integration frameworks should address the lagging behind of organisational and cultural adaptation. To foster a culture of AI usage, organisations will benefit by adopting a phased plan, starting with smaller project so it's easier to gauge the value of AI in DevOps. Implementation accompanied by basic training on AI/ML concepts and workshops on ethical concerns can further reduce fear and resistance. In the early stages, gamifying the adoption process, like creating a point system for using AI tools or contributing to knowledge base, can make the process more engaging. Organisations can also develop metrics to tract team satisfaction and collaboration levels to boost a cultural shift.
Lack of knowledge sharing can be addressed by creating an internal wiki or knowledge base. Organising "tech talk" sessions where teams can share their experience with AI in DevOps can be beneficial. Also, internal mentorship programmes where AI-savvy teams or individuals can guide others and a push for participation in external conferences might improve knowledge sharing. Furthermore, the moderate score in Lean practices suggests that efficiency achieved by using AI can still be improved by focusing on end-to-end optimisation. These insights can be used by organisations and researchers alike for providing a more balanced approach to AI integration and possibly attain 5's in all aspect of CALMS framework.

6 Knowledge Gaps and Future Research Directions

Despite the significant progress made over the years, several noteworthy knowledge gaps remain and closing these gaps is crucial for continued evolution and effective implementation of AI in DevOps.

6.1 Long-term Impact on Software Quality

While AI tools have shown promise in improving short-term productivity, more research needs to be conducted on their long-term impact on software quality, maintainability and

technical debt.

Peng (2023) highlight that despite LLMs like GPT-4 have demonstrated tremendous ability in generating code quickly, their impact on overall system design and long-term maintainability is not well understood. While functional, AI-generated code poses a risk risk of not adhering to best practices for maintainability and scalability. Huang et al. (2024) further emphasise the need for studies on how AI-assisted coding impacts the accumulation of technical debt over time. They propose that while AI might handle immediate coding requirements, it could potentially introduce subtle issues that only surface in the long term.

AI-driven code generation and testing can introduce inefficient and insecure code and can reproduce biases present in training data. Overreliance can even lead to reduction in developer capability and vulnerabilities being overlooked. Future research should focus on long-term studies that track software projects using AI-assisted development over extended periods, juxtaposing them with traditionally developed projects in terms of metrics such as maintainability, bug rates and refactoring needs while exercising quality assurance processes like peer reviews.

6.2 Understanding of AI decisions in DevOps Contexts

As AI systems become more integrated into critical DevOps processes, there's a need for better clarity of AI decisions and recommendations.

Lyu et al. (2021) address this issue in their work on consistent interpretation of AIOps models. They highlight that the "black box" nature of many AI models can be problematic in scenarios where understanding the reasoning behind decisions is of importance. Their work suggests that only high-performing models ($AUC > 0.75$) yield consistent interpretations, emphasising the need for both accuracy and clarity in AI-powered DevOps systems.

Kolltveit and Li (2023) produced a systematic literature review on operationalizing machine learning models wherein they have explained the importance of model understandability in production environments. They convey that transparent AI techniques are especially crucial in industries with stricter regulations or when dealing with critical systems.

Future research should focus on developing AI models that not only perform well but also provide clear, understandable explanations for their decisions. This could involve techniques from the field of explainable AI (XAI)[2] adjusted specifically for DevOps.

6.3 Human-AI Collaboration Models

Peng (2023) discuss the limitations of current AI models in understanding the holistic development process of complex systems. They suggest that human-AI collaboration models are need to take advantage of both AI (in coding/automation tasks) and humans (in high-level design and problem solving).

Cai et al. (2024), in their work on federated learning for adapting LLMs to software development, highlight that collaborative learning approaches could potentially preserve data privacy while enabling collective improvements of AI models.

[2]https://www.ibm.com/topics/explainable-ai

Future research could explore novel interaction paradigms that allow intuitive collaboration between human developers and AI models to facilitate seamless automation, balanced with human oversight.

6.4 Cross-Domain Knowledge Transfer

Organisations looking into adopting AI for their DevOps workflows could benefit from AI models that can effectively transfer knowledge across different programming languages, frameworks, and problem domains.

de Oliveira and Castor (2024) conducted a study to aid experiments with LLMs in software development and they highlight the challenge of adapting AI models to diverse development contexts and programming languages.

In their study on fully autonomous programming with LLMs, Liventsev et al. (2023), briefly discuss the inability of current models to generalise across different programming paradigms and problem areas.

Different domain have different regulatory compliances such as HIPAA for Healthcare and GDPR for finance in the EU. Organisations would need to develop knowledge graphs that map the relationship between different domain-specific concepts and use these to train models/build tools that understand the different constraints of different domains and thus facilitate more accurate knowledge transfer. On the side of DevOps, organisations could develop standardised APIs for already existing domain specific tools so that the said tools can be used with existing DevOps pipelines.
In these contexts the following modifications can be done to the framework discussed in the next section. Organisations can establish a Regulatory Compliance Checker that verifies if the current practices meet the specific requirements of the target domain. There would also be a need for domain specific metrics during monitoring phase of the DevOps workflow. Lastly, organisations that work across domains would benefit from maintaining an internal repository that stores best practices, common patterns, and lessons learned from AI integration across different domains. Future frameworks could incorporate means for organisations from different domains to collaboratively train and improve AI models which would further facilitate cross domain knowledge sharing.

7 Practical Implementation of AI in DevOps: A Guide for Organisations

This study has explored various aspects of AI integration in DevOps, highlighting how their synergy offers numerous benefits but is also riddled with challenges. This section summarises key findings and provides a practical guide for organisations seeking to implement AI in their DevOps workflows.

7.1 Summary of Key Findings

1. AI techniques such as Large Language Models (LLMs) are being widely adopted across various DevOps stages, from planning to monitoring [Peng (2023), Sauvola et al. (2024), Wang and Chen (2023), Petrović (2023a)].

2. Predictive analytics powered by Machine Learning are supercharging DevOps by enhancing decision making, anomaly detection and proactive issue resolution [Kumar

et al. (2023), Mohammed et al. (2024), Hausi A. et al. (2023)].

3. Significant benefits of AI integration include improved efficiency, faster time-to-market, and automated bug detection [Weber et al. (2024), Du et al. (2024), Vemuri et al. (2024)].

4. Despite benefits, challenges remain in areas such as data privacy, integration complexity, and skill gaps [Sauvola et al. (2024), Rangaraju et al. (2023), Battina (2021)].

7.2 Implementation Guide for Organizations

Based on the findings discussed above and in section 5, presented below is a step-by-step guide for organisations looking to implement AI in their DevOps workflows. Figure 7 gives a visual representation of this guide in the form of a decision tree.

1. **Assess current DevOps Maturity:**
   - Evaluate existing DevOps practices using frameworks like CALMS (Culture, Automation, Lean, Measurement, Sharing).
   - Identify weak areas which could benefit the most from leveraging AI.

2. **Begin with high impact, low risk zones:** Proceed with AI implementation areas that offer clear benefits with relatively lower implementation risks, such as:
   - Code completion and suggestion tools, for e.g. GitHub Copilot [Leung and Murphy (2023), Mehmood et al. (2023)]
   - Automated test case generation [Bayrı and Demirel (2023), Mehmood et al. (2023)]
   - Log analysis and anomaly detection [Mohammed et al. (2024), Hausi A. et al. (2023)]

3. **Safeguard sensitive data:** Ensure robust data collection and implement data governance policies that prioritise privacy and address security concerns [Camacho (2024)].

4. **Choose appropriate AI Tools and Frameworks:**
   - Tool selection needs to be based on specific needs and existing technology stack while considering factors like integration capabilities, scalability and vendor support. Usually low-code and no-code tools can be easily integrated into most workflows [Waqas et al. (2024)].
   - For ML development and deployments, one can opt for Tensorflow or Keras. Popular cloud based-AI services include AWS Sagemaker and Google Cloud AI Platform [Mohammed et al. (2024), Camacho (2024)].

5. **Address the skill gap:**
   - Train existing staff on AI and ML concepts specifically nuanced for DevOps.

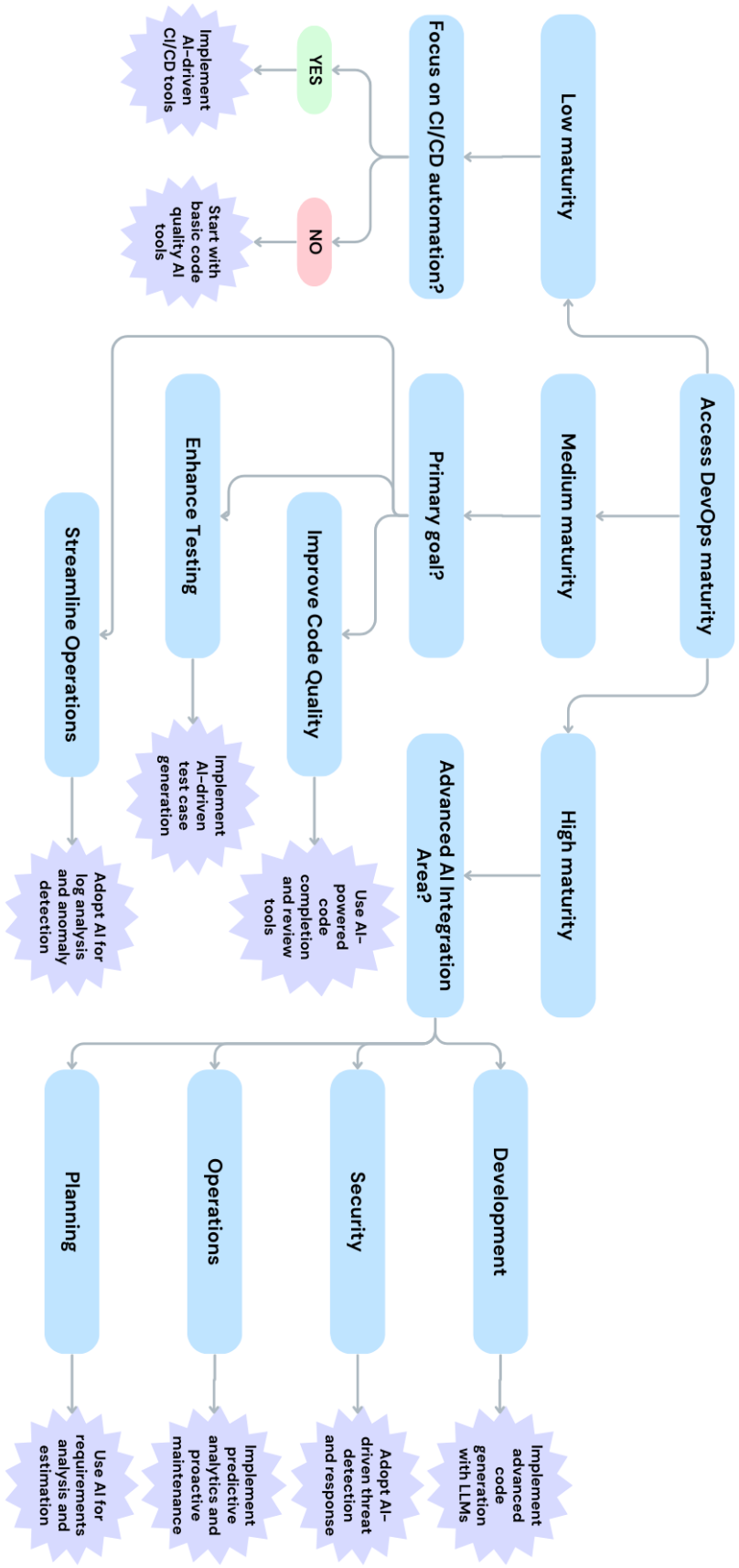


Figure 7: Decision Tree to guide organisations in adopting AI technologies based on their DevOps needs and maturity.

- If needed, hire specialists or partner with AI experts [Sauvola et al. (2024), Battina (2021)].

6. **Implement in phases:**
    - Choose a project in a non-critical area of the DevOps pipeline and start with it.
    - Gradually expand AI integration to other projects and stages of the pipeline, applying the intel gathered from the first implementation.
7. **Foster a Culture of AI Adoption:**
    - Encourage experimentation and learning within teams.
    - Promote collaboration between development and operations teams with AI/ML experts or teams [Peng (2023), Cai et al. (2024)].
8. **Monitor and Measure Impact:** Establish metrics such as performance improvements, cost savings, and team productivity and use them to evaluate the impact of AI integration on DevOps processes [Kumar et al. (2023), Vemuri et al. (2024)].
9. **Stay Informed and Adaptable:** AI technologies are evolving rapidly and so are the tools and methodologies that leverage them. It would be necessary to stay abreast of new developments in AI for DevOps and be prepared to adapt as new technologies and best practices emerge.

7.3 Potential Roadmap for AI Integration in DevOps

As described in the implementation guide above, Figure 8 gives an overview of a proposed roadmap for an organisation looking towards integrating AI into their DevOps workflows. By following this guide and roadmap, organisations can achieve their goal of systematically integrating AI while mitigating potential risks. It is imperative to understand that successful AI integration is a continuous process, requiring long-term commitment, constant learning and persistence to adapt to evolving technologies and best practices.

During Phase 3, certain qualitative and quantitative metrics can be established to measure cultural shift. Measuring Adoption Rate, i.e. the number of employees using AI in their daily workflows and the number of projects that incorporate AI, should be foremost. They could also have productivity metrics that measure the time taken to complete tasks, number of requirement fulfilled and the number of bugs fixed in each sprint, before and after AI integration. Innovation metrics such as number of new ideas or process improvements applied using insights provided by AI, could also be an indicator of cultural shift compared to pre-AI integration.
Qualitative metrics can include employee feedback surveys and AI training participation. Internal discussion facilitates knowledge sharing and this could be tracked to get an idea of cultural shift. On the contrary, tracking the number of support tickets related to AI tools can be indicative of resistance to change and a decrease in their numbers would mean a positive cultural shift. Finally, measuring customer satisfaction, especially in terms of quality and lower time to market would indicate successful human-AI collaboration.

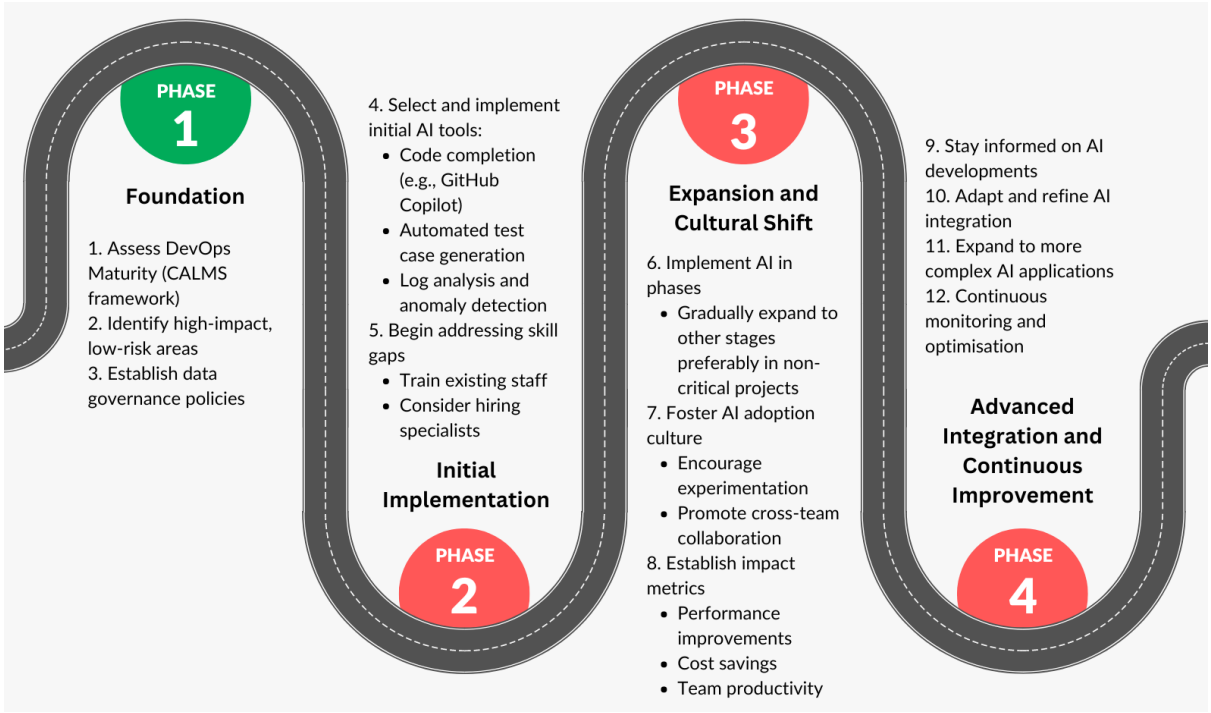


Figure 8: Proposed roadmap for AI integration in DevOps

8 Conclusion

The integration of AI into DevOps workflows represents a major enhancement in how software is developed, deployed, and maintained. This comprehensive review of 29 studies reveals that AI technologies have transformed DevOps with features such as code generation, automated bug detection, requirements analysis and proactive system monitoring.The research highlights that LLMs are the dominant technique for AI integration in DevOps, with 40% of applications being done with with LLMs. It is found that AI integration offers a 65% boost to productivity and results in enhanced code quality and faster time-to-market.

This analysis, framed within the CALMS framework reveals that technical aspects such as automation and measurement exhibit high readiness while cultural aspects like knowledge sharing face challenges. This highlights the need for integration frameworks that give equal importance to technical and human factors. The proposed roadmap combined with the decision tree provides a phased approach and that balances quick wins with long-term strategic goals, providing a practical guidance for organisations at various stages of AI adoption.

This study also identifies the need for future research into long-term impact of AI on software quality, explainable AI in DevOps contexts, and more effective human-AI collaboration models. As AI techniques continue to evolve, their use in DevOps will unlock more frontiers leading to a more nuanced software development process.

References

Alenezi, M., Zarour, M. and Akour, M. (2022). Can Artificial Intelligence Transform DevOps?, *arXiv preprint arXiv:2206.00225* .

URL: *https://arxiv.org/abs/2206.00225*

Ali, M. S. and Puri, D. (2024). Optimizing devops methodologies with the integration of artificial intelligence, *2024 3rd International Conference for Innovation in Technology (INOCON)*, pp. 1–5.

Andrea C. Tricco, PhD, M. E. L. M. W. Z. M. K. K. O. P. B. H. C. P. D. L. P. M. B. D. M. P. M. M. D. P. P. M. T. H. P. L. W. P. S. H. P. E. A. A. M. P. M. C. C. M. M. J. M. P. L. S. P. M. L. H. P. M. B. A. A. B. B. M. G. W. P. C. G. M. S. L. P. C. M. G. P. R. M. T. M. P. M. E. V. L. P. K. S.-W. M. P. J. M. M. T. C. P. M. T. M. P. M. and Sharon E. Straus, MD, M. (2018). Prisma extension for scoping reviews (prisma-scr): Checklist and explanation, *Annals of Internal Medicine* **169**(7): 467–473. PMID: 30178033.
URL: *https://doi.org/10.7326/M18-0850*

Battina, D. S. (2021). AI and DevOps in Information Technology and its Future in the United States, *IJRAR - International Journal of Research and Analytical Reviews (IJRAR)* **8**(1): 108–112.
URL: *http://www.ijrar.org/IJRAR21A1580.pdf*

Bayrı, V. and Demirel, E. (2023). Ai-powered software testing: The impact of large language models on testing methodologies, *2023 4th International Informatics and Software Engineering Conference (IISEC)*, pp. 1–4.

Cai, Z., Chen, J., Chen, W., Wang, W., Zhu, X. and Ouyang, A. (2024). F-codellm: A federated learning framework for adapting large language models to practical software development, *Proceedings of the 2024 IEEE/ACM 46th International Conference on Software Engineering: Companion Proceedings*, ICSE-Companion '24, Association for Computing Machinery, New York, NY, USA, p. 416–417.
URL: *https://doi.org/10.1145/3639478.3643533*

Camacho, N. G. (2024). Unlocking the potential of ai/ml in devsecops: Effective strategies and optimal practices, *Journal of Artificial Intelligence General science (JAIGS) ISSN:3006-4023* **3**(1): 106–115.
URL: *https://ojs.boulibrary.com/index.php/JAIGS/article/view/72*

de Oliveira, B. and Castor, F. (2024). Athenallm: Supporting experiments with large language models in software development, *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension*, ICPC '24, Association for Computing Machinery, New York, NY, USA, p. 69–73.
URL: *https://doi.org/10.1145/3643916.3644437*

Dos Santos, C. A., Bouchard, K. and Petrillo, F. (2024). Ai-driven user story generation, *2024 International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA)*, pp. 1–6.

Du, X., Liu, M., Wang, K., Wang, H., Liu, J., Chen, Y., Feng, J., Sha, C., Peng, X. and Lou, Y. (2024). Evaluating large language models in class-level code generation, *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 982–994.
URL: *https://doi.ieeecomputersociety.org/10.1145/3597503.3639219*

GMinsights (2022). Global aiops market revenue to cross $38 bn by 2032, Global Market Insights Inc. Available at: `https://www.gminsights.com/pressrelease/aiops-market` (Accessed: 02 July 2024).

Hausi A., M., Litoiu, M., Rivera, L. F., Rasolroveicy, M., Villegas, N. M., Tamura, G., Watts, I., Erpenbach, E. and Shwartz, L. (2023). Proactive continuous operations using large language models (llms) and aiops, *Proceedings of the 33rd Annual International Conference on Computer Science and Software Engineering*, CASCON '23, IBM Corp., USA, p. 198–199.

Huang, T., Sun, Z., Jin, Z., Li, G. and Lyu, C. (2024). Knowledge-aware code generation with large language models, *Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension*, ICPC '24, Association for Computing Machinery, New York, NY, USA, p. 52–63.
URL: *https://doi.org/10.1145/3643916.3644418*

Kolltveit, A. B. and Li, J. (2023). Operationalizing machine learning models: a systematic literature review, *Proceedings of the 1st Workshop on Software Engineering for Responsible AI*, SE4RAI '22, Association for Computing Machinery, New York, NY, USA, p. 1–8.
URL: *https://doi.org/10.1145/3526073.3527584*

Kumar, A., Nadeem, M. and Shameem, M. (2023). Machine learning based predictive modeling to effectively implement devops practices in software organizations, *Automated Software Engineering* **30**(2): 21.
URL: *https://doi.org/10.1007/s10515-023-00388-8*

Leung, M. and Murphy, G. (2023). On automated assistants for software development: The role of llms, *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, IEEE Computer Society, Los Alamitos, CA, USA, pp. 1737–1741.
URL: *https://doi.ieeecomputersociety.org/10.1109/ASE56229.2023.00035*

Liventsev, V., Grishina, A., Härmä, A. and Moonen, L. (2023). Fully autonomous programming with large language models, *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '23, Association for Computing Machinery, New York, NY, USA, p. 1146–1155.
URL: *https://doi.org/10.1145/3583131.3590481*

Lu, Q., Zhu, L., Xu, X., Whittle, J. and Xing, Z. (2022). Towards a roadmap on software engineering for responsible ai, *Proceedings of the 1st International Conference on AI Engineering: Software Engineering for AI*, CAIN '22, Association for Computing Machinery, New York, NY, USA, p. 101–112.
URL: *https://doi.org/10.1145/3522664.3528607*

Lyu, Y., Rajbahadur, G. K., Lin, D., Chen, B. and Jiang, Z. M. J. (2021). Towards a consistent interpretation of aiops models, *ACM Trans. Softw. Eng. Methodol.* **31**(1).
URL: *https://doi.org/10.1145/3488269*

MacNeil, S., Tran, A., Hellas, A., Kim, J., Sarsa, S., Denny, P., Bernstein, S. and Leinonen, J. (2023). Experiences from using code explanations generated by large

language models in a web software development e-book, *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSE 2023, Association for Computing Machinery, New York, NY, USA, p. 931–937.
URL: *https://doi.org/10.1145/3545945.3569785*

Market.us (2023). The future of mlops: Emerging trends and technologies, LinkedIn. Available at: `https://www.linkedin.com/pulse/future-mlops-emerging-trends-technologies-markets-us-wwtwf/` (Accessed: 03 July 2024).

Mboweni, T., Masombuka, T. and Dongmo, C. (2022). A systematic review of machine learning devops, *2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, pp. 1–6.

Mehmood, S., Janjua, U. I. and Ahmed, A. (2023). From manual to automatic: The evolution of test case generation methods and the role of github copilot, *2023 International Conference on Frontiers of Information Technology (FIT)*, pp. 13–18.

Mohammed, A. S., Saddi, V. R., Gopal, S. K., Dhanasekaran, S. and Naruka, M. S. (2024). Ai-driven continuous integration and continuous deployment in software engineering, *2024 2nd International Conference on Disruptive Technologies (ICDT)*, pp. 531–536.

Nasiri, S. and Lahmer, M. (2024). Ai-driven methodology for refining and clustering agile requirements, *2024 4th International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, pp. 1–7.

Peng, X. (2023). Software development in the age of intelligence: embracing large language models with the right approach, *Frontiers of Information Technology & Electronic Engineering* **24**(11): 1513–1519.
URL: *https://doi.org/10.1631/FITEE.2300537*

Petrović, N. (2023a). Chat gpt-based design-time devsecops, *2023 58th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST)*, pp. 143–146.

Petrović, N. (2023b). Machine learning-based run-time devsecops: Chatgpt against traditional approach, *2023 10th International Conference on Electrical, Electronic and Computing Engineering (IcETRAN)*, pp. 1–5.

Rangaraju, S., Ness, S. and Dharmalingam, R. (2023). Incorporating ai-driven strategies in devsecops for robust cloud security, *International Journal of Innovative Science and Research Technology (IJISRT)* **8**(11): 2359–2365.
URL: *http://www.ijisrt.com*

Sauvola, J., Tarkoma, S., Klemettinen, M., Riekki, J. and Doermann, D. (2024). Future of software development with generative ai, *Automated Software Engineering* **31**(1): 26.
URL: *https://doi.org/10.1007/s10515-024-00426-z*

Vemuri, N., Thaneeru, N. and Tatikonda, V. (2024). Ai-optimized devops for streamlined cloud ci/cd, *International Journal of Innovative Science and Research Technology* **9**: 7.

Wang, J. and Chen, Y. (2023). A review on code generation with llms: Application and evaluation, *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*, pp. 284–289.

Waqas, M., Ali, Z., Sánchez-Gordón, M. and Kristiansen, M. (2024). *Using LowCode and NoCode Tools in DevOps: A Multivocal Literature Review*, Springer Nature Switzerland, Cham, pp. 71–87.
URL: *https://doi.org/10.1007/978-3-031-50590-4*$_5$

Weber, T., Brandmaier, M., Schmidt, A. and Mayer, S. (2024). Significant productivity gains through programming with large language models, *Proc. ACM Hum.-Comput. Interact.* **8**(EICS).
URL: *https://doi.org/10.1145/3661145*