DevSecOps: A Comprehensive Study and Hands-On Application for sustainability

[Mohith Krishna V , Madhav Ram, M Pranavkrishnan, Dr Tripty Singh][1], Adhirath Mandal[2], Dr Suman Chatterji[3]
[1]Dept. of Computer Science and Engineering
Amrita School of Computing, Bangalore, Amrita Vishwa Vidyapeetham, India
[2]Assistant Professor, Graphic Era (Deemed to be University), Dehradun, Uttarakhand, India
[3]Dept of Mechanical Enigneering, Konju National University, Cheonan, South Korea
bl.en.u4cse21125@bl.students.amrita.edu, bl.en.u4cse21117@bl.students.amrita.edu,
bl.en.u4cse21113@bl.students.amrita.edu, tripty_singh@blr.amrita.edu, adhirathmandal.me@geu.ac.in,
mrsumanmech@gmail.com

***Abstract*— This study applies a seamless integration of security protocols within the DevOps lifecycle, created through the intricate association of Continuous Integration/Continuous Deployment (CI/CD), robust security frameworks, vigilant monitoring via the GitOps methodology. This aims to simplify the workflow and automate the security and code reviews of the project and focus on development of features of the webapp. Our exploration unfolds through an exhaustive examination of existing literature and empirical analyses, delving deep into the challenges and advantages inherent in the adoption of a DevSecOps paradigm. This automated DevSecOps module is crafted for a modern software development infrastructure for small teams, fusing security practices within the realm of Infrastructure as Code (IaC). Performance measures, ranging from time efficiency assessments to a thorough survey of essential DevSecOps indications, are constantly monitored. Along with performance measures, sustainability measurements such as efficiency and resource management are tracked. This technique gives essential insights that will help guide future DevSecOps upgrades.**

Keywords— DevOps, DevSecOps, CI/CD, Security, GitOps, IAC, Scalable operations.

I. INTRODUCTION

The software development industry has seen a massive change as a result of DevSecOps becoming more mainstream and a wider acceptance of its principles. Collaboration, automation, deployment, and continuous integration are elements that DevOps, an abbreviation for development (Dev) and operations (Ops), combine to enhance software delivery. This has changed the ways in which businesses develop, test, and release software. The DevOps methodology, however, presents a serious challenge in terms of security

The idea behind DevSecOps is to bridge the gap that exists between existing DevOps methodologies and security has become the natural evolution of DevOps. An important reason for widespread acceptance of DevSecOps is the increased recognition of the vital role that security plays in today's digital ecosystems.

DevSecOps, which aims to bridge the gap between security and DevOps approaches, has evolved as a natural progression, acknowledging security's vital role in today's increasingly interconnected digital ecosystems. This shift in methods used for security means that companies rely less on reactive countermeasures against threats after deployment which can result in inefficiencies, delays and a higher risk of breaches as noted by Akanksha Gupta et al. [9].

The implementation of DevSecOps means that security becomes the topmost priority throughout the entire software development process. Cybersecurity specialists are always in close collaboration with the development teams throughout the DevSecOps cycle, incorporating security measures and features into the dev-cycle and addressing any security threats from the initial stages. Organisations encourage the use of coding techniques and teach engineers to follow security guidelines and best practices. Finding vulnerabilities in code and dependencies is made easier with the use of automated security testing technologies used in the integration and deployment parts of the DevSecOps cycle.

Environmental sustainability is emphasized by the DevSecOps cycle. This is made possible by using efficient development methods like resource optimization, reducing wastage and task automation. A great example of this is how carbon emissions and energy usage are reduced by the automation of various security processes and also the usage of containerization technology. Another aspect of sustainability that DevSecOps creates is economic sustainability, as resource utilization is optimized and costs of security breaches reduce as a direct effect of integrating security into the development process, DevSecOps is economically sustainable.

According to the IBM Cost of Data Breach Report 2021 [10], the global average cost of a data breach was $4.24 million, with healthcare, financial services, and technology sectors being the most targeted industries. Another study by Puppet [11] shows organizations that implement DevSecOps practices experience a 50% reduction in the time it takes to remediate security vulnerabilities compared to those that do not. Results of these studies show the necessity to make DevSecOps practices compulsory in the development cycle for any type of software.

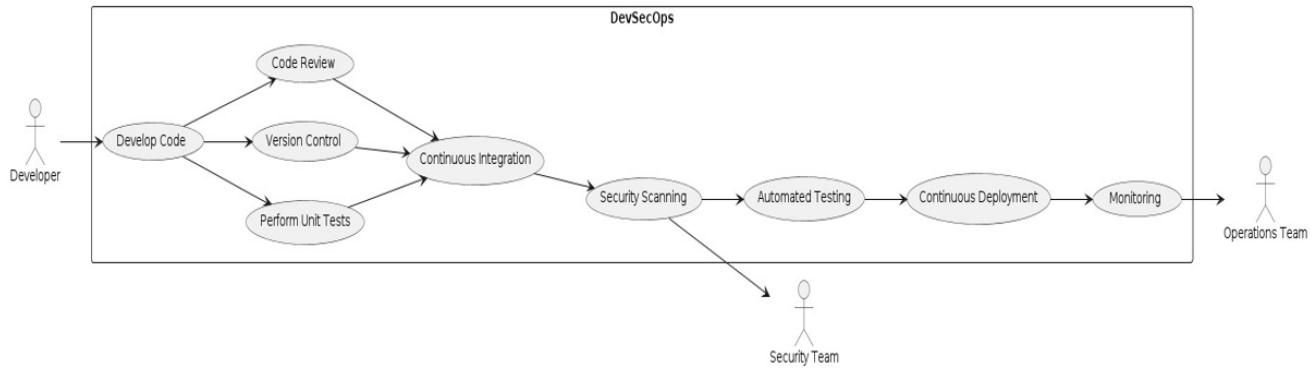

Fig 1. Use Case Diagram DevSecOps

Fig 1 explains the full overview of DevSecOps and results expected via a use case diagram, security and continuous monitoring being the pillars of the DevSecOps Framework.

In this research work, section II focuses on related works, section III will talk about the research gap that exists in this space currently. Section IV delves deep into the materials and methods used. Section V is our proposed methodology. Section VI are results from the research work and Section VII talks about conclusions drawn from the research work.

II. Literature Review

The study by Håvard Myrbakken et al. [1] gives a complete overview of DevSecOps, focusing on DevSecOps' meaning, definition, principles, challenges and also its benefits. Through a comprehensive review of works related to DevSecOps, the authors collect insights from various sources like white papers, articles and blogs. These sources examine the integration of Security processes into DevOps environments. Here, DevSecOps is defined as: 'the amalgamation of security practices and principles with DevOps methodologies emphasizing proactive security measures throughout the software development lifecycle.' The study also identifies shortcomings of adopting DevSecOps including cultural resistance and a lack of tools. Benefits of implementing DevSecOps practices such as reduced unplanned work and increased automation are also highlighted in the study. The study concludes by suggesting possibilities for future research, including surveys on organizational adoption of DevSecOps and further research into frameworks that help in implementing DevSecOps.

In this research [2], Jessica Diaz et al. investigated about integrating safety practices into DevOps environments such as IoT-centred systems. They point out difficulties that are caused by very much scattered and heterogeneous internet of things(IoT) systems which slows down the development process and complicates cyber security event monitoring too. The authors define DevSecOps as a culture that promotes collaboration among development, security and operation teams to ensure efficient delivery of business value. They suggest for a self-service cybersecurity monitoring activity called Fast & Continuous Feedback from Ops to Dev (F&CF) which they claim can help in fast identification and resolution of security problems. They design this activity using the Software & Systems Process Engineering Metamodel (SPEM) that allows setting up different types of service based on monitoring or alerting according to what suits each team involved in the project. The case study also shows how Monitoring as Code (MaC) has been implemented through Automation Infrastructure deployed via virtualization and containerization technology by the authors themselves. This method encourages cooperation between developers, operators and security personnel leading to adoption of DevSecOps culture because it enables continuous visibility into system behaviour throughout its lifecycle. The research concludes by talking about the implications of the method proposed, the work also creates an outline for future research and also talks about integration of advanced monitoring techniques using machine learning algorithms.

The study by Muhammad Azeem Akbar et al. [3] aims to answer the increasing need for including security in the DevOps methods necessary for ensuring secure software building. The research presents a thorough literature review; and a survey through questionnaires which look into what challenges are faced when implementing DevSecOps process and their priority. It also seeks to determine how these core categories of problems are related and rank them in order of importance so that people who work in this area as well as those who study it can know where to put more effort on their investigations. The reported difficulties include a variety of domains, including standards, safe code, and testing tools. The researchers emphasize the importance of overcoming these challenges in order to more successfully generate safe programmes in DevOps environments, focusing on the importance of safety at all phases of the software development lifecycle.

Tony Hsu et al. [4] proposed a technique for incorporating security into the DevOps workflow, as well as features that address the issues of protecting fast growing software.

An important part of DevSecOps is the process of setting up security targets and KPIs that match with various industry best practices, such as the OWASP SAMM (Software Assurance Maturity Model). This Framework helps use have a better understanding of an organisation's security measures. When KPIs and other aspects of security are mentioned clearly, it helps the developer track progress throughout the project. Microsoft Security Development Lifecycle (SDL) and SAMM offer an example of how businesses may create security assurance programmes that are customised to meet

their unique requirements and are both efficient and successful.

To develop secure and robust software systems, we need to establish secure approaches to coding and to achieve that DevSecOps promotes OWASP Code Review, CWE (Common Weakness Enumeration) and CERT (Computer Emergency Response Team) guidelines which are industry best practices. Secure approaches like this one helps organizations decrease the chances of security breaches and vulnerabilities in the applications that they send out.

Another important aspect of DevSecOps is security testing. This helps the project development team/organisation to identify and subsequently erase security threats early in the development cycle. This work offers various plans and strategies to conduct thorough checks in security. Automated security testing tools enable continuous integration pipelines to increease the pace of the testing process and guarantee resolution of security threats. Both incident response and security monitoring are essential components of a successful DevSecOps approach. The work covers incident response best practices in addition to planning, containment, detection and post-event analysis. By utilising security monitoring technologies like threat intelligence, security scanning, and logging to quickly identify and address security threats, organisations can lessen the impact that security incidents have on their systems.

The paper by Rakesh Kumar rt al. [5] gives an in-depth analysis of the ways hpe cloud technologies interact with Agile, DevOps and Open Source Software (OSS) to promote corporate agility and increasing application delivery speeds. The authors talk about DevSecOps as a cure to the issues after realizing the difficulty of realising these goals while ignoring security practices. One of the best contribution's is the creation of the ADOC conceptual security paradigm, which automates and seamlessly integrates security controls into the delivery workflow using OSS and cloud-native technologies like microservices, containers, and serverless computing. Accelerated development of secure, market-ready apps is the aim of the ADOC paradigm.

The research by Sakthiswaran Rangaraju et al. [6] examines methods to improve cloud security by integrating AI techniques into the DevSecOps framework. The success of these AI solutions for reducing risks is evaluated in the research by both quantitative and qualitative methods. The selection, implementation, and operationalization of AI-powered security solutions inside DevSecOps pipelines are stressed. We take special care to evaluate the security requirements and challenges of cloud environments to determine where AI-based strategies should be used. The study comprises a quantitative survey covering AI adoption and efficacy together with qualitative interviews that explore experiences with integrating AI, strategies employed, and effects on security outcomes. With AI-driven solutions for improved security, the study combines important DevSecOps concepts including shift-left security, automation, continuous monitoring, and compliance as code.

This paper by Saniora R. Duclervil et al. [7] starts out by recognising the importance of technology in the modern economy, especially its pervasive influence on day-to-day living, in the quickly changing field of software development. The authors stress the issues in efficiency, security, and maintenance that result from the proliferation of connected devices via the Internet of Things (IoT). The study aims to tackle the growing cybersecurity problems, particularly in the context of IoT devices and interconnected networks, by creating a new DevSecOps methodology. Continuous security measures are incorporated into the original DevOps framework through this agile-based software development process. The process entails a transition to agile development, with a focus on the ongoing evolution of product requirements. The rationale behind selecting DevSecOps as the recommended paradigm is supported, emphasising how important it is to include security early in the development cycle. The actual application of the proposed DevSecOps model, where security is integrated into the development lifecycle without any issues, is a project management website. The necessity to thoroughly incorporate security across all development phases is emphasised in the paper's conclusion, which also acknowledges project limits and suggests future directions for improving the model.

The technical report, written by Jose Morales et al. [8] and published by Carnegie Mellon University, provides a thorough implementation guide for DevSecOps (DSO) in defence and highly regulated domains, including complex networks of systems. The document covers particular issues in high-risk and systems-of-systems environments and describes the fundamentals, workings, and expected advantages of DSO. Preparation, Establishment, and Management—the three main stages of implementing DSO—are introduced. Making a plan for developing the DSO ecosystem and establishing realistic objectives are the main preparation tasks. Achieving an initial capability through establishment entails developing processes, automation, culture, and system design. Both ecosystem health and organisational performance are measured and tracked continuously in management. The paper places a strong emphasis on taking a balanced approach, trying to minimise needless rework while effectively evolving the ecosystem. Comprehensive activity summaries and references on technical topics, change management, and advanced DSO data are provided in the appendices. The abstract especially focuses on projects related to defence and regulated contexts as it emphasises the part played by DSO in the integration of development, security, and operations for quicker software delivery.

The emergence of DevSecOps has been addressed in this work done by Prates et al. [12]. This is the new pattern that combines security practices with the development cycles of soft wares using DevOps method. The main objective is to find out different measurements that can be used by institutions when implementing DevSecOps so as to monitor their effectiveness. Such investigation employs Multivocal Literature Review (MLR) thereby identifying various professional and academic perspectives about metrics for safety integration into development cycle called DevSecOps. Though it may seem like a fresh issue but still nine metrics

related to efficiency of DevSecOps have been established through this research. These indicators are critical for any organisation looking to assess its level of commitment to safeguarding software development processes through DevSecOps; hence, such organisations should not neglect them. Finally, this paper highlights the importance of further research in these areas while also suggesting potential avenues for future investigations, such as interviewing or surveying those involved in DevSecOps, so that proposed measures can be refined and extended based on their feedback.

The proposed intelligent traffic management system by Tripty Singh [13] uses CMDP (Constrained Markov Decision Process) and policy gradient methods for predicting congestion at intersections and suggests that security should be integrated into every aspect of the robotic system's development and deployment. This is achieved by incorporating security into development and operations is therefore aligned with DevSecOps principles. In order to make AI driven traffic management decisions possible reinforcement learning algorithms have to be employed which include CMDPs also known as policy gradients based on this strict safety measures need to be followed. Security monitoring must always remain continuous starting from real-time data processing through sensors and traffic signal timing decisions.

Tripty Singh et al. [14] discusses agricultural robots in closed fields and aligns with DevSecOps principles as it maintains security all across the creation of the robots, while the research work focuses on the design and implementation of farming robots, it emphasises on functionality, reliability and data security which are DevSecOps principles.

Dr. Tripty Singh et al. [15], explore the current design frameworks in several sectors, emphasizing the difficulties brought forth by fresh and machine-generated data. The integration of Apache Hadoop into contemporary data architecture is the main topic of discussion, along with its function in distributed computing and large-scale dataset querying for dependable and quick results. The authors discuss the exponential rise of data, projecting that it would reach 40 Zeta Bytes by 2020, with new data kinds responsible for a large share of this increase from 2.8 Zeta Bytes in 2012. The article highlights Hadoop's potential for managing unstructured and semi-structured data by offering distributed processing, storage, and scalable analytics.

III. Research Gap

The papers reviewed does not address the process of automated review of code and combining the GitOps workflow seamlessly along with the CI/CD and Software Development Lifecycle. This Research involves the creation of an automated DevSecOps module that seamlessly integrates security protocols within the DevOps lifecycle. This integration is achieved through the intricate association of Continuous Integration/Continuous Deployment (CI/CD), robust security frameworks, and vigilant monitoring via the GitOps methodology. Unlike traditional approaches, this method aims to simplify the workflow, automate security and code reviews, and prioritize the development of webapp features. It specifically tailors to modern software development infrastructure for small teams, embedding security practices within the realm of Infrastructure as Code (IaC). The unique aspect lies in the comprehensive exploration through literature examination and empirical analyses, uncovering challenges and advantages inherent in adopting a DevSecOps paradigm. The focus on performance metrics, including time efficiency assessments and critical DevSecOps indicators, provides invaluable insights for steering future enhancements in the implementation of DevSecOps.

IV. Methods and materials

This research implantation encompasses a secure approach to the typical Dev0ps workflow that ensures end to end security and maintainability. This is particularly of importance to smaller teams to ensure high throughput and allows to focus on the underlying software features rather than the workflow via automation. Collaboration with stakeholders to gather requirements for the web application, including functional and non-functional security requirements is key. Also identify regulatory compliance requirements, such as GDPR or HIPAA, if applicable.

We showcase this DevSecOps workflow on an example cloud native webapp. We can consider the following stack which is highly approachable and maintainable by small teams with minimal effort. Containerization and Orchestration using Kubernetes and Docker. GitHub Actions is used for the Continuous Integration and Continuous Deployment of the webapp seamlessly integrating the ever-evolving git commits. We build a middleware using OWASP ZAP and SonarQube to monitor and check for vulnerabilities and potential threats in the codebase. Threat modelling is performed to identify potential security threats and vulnerabilities in the web application architecture by utilizing techniques such as STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) to assess security risks. Enforce secure coding guidelines based on industry standards such as OWASP Secure Coding Practices, we also automate and provide a singular platform for human code reviews.

The implementation sets up a CI/CD pipeline to automate the build, test, and deployment processes. Integrates security testing tools into the pipeline to perform static code analysis, dependency scanning, and vulnerability assessment. Ensures that security checks are performed at every stage of the pipeline. Container security best practices are applied to secure Docker containers and Kubernetes clusters. Utilize tools such as Docker Bench for Security and Kubernetes Pod Security Policies to harden containerized environments. Implement a robust monitoring and verbose logging mechanisms to detect and respond to security incidents in real-time. Utilize tools like Prometheus and Grafana for monitoring and ELK stack (Elasticsearch, Logstash, Kibana) for centralized logging and log analysis. Automated thorough testing of the DevSecOps workflow, codebase etc. including functional testing, security testing, and performance testing. Utilize automated testing tools and frameworks to validate

the effectiveness of security controls and identify any gaps or vulnerabilities.

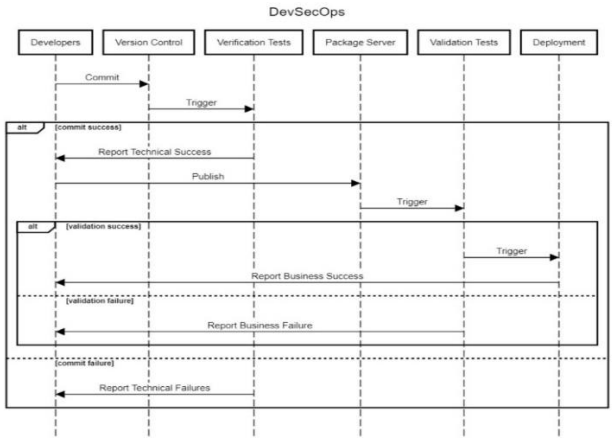

Fig. 2. UML Sequence Diagram for a typical DevSecOps pipeline

Fig.2. is the UML Sequence Diagram for DevSecOps. It starts with the developer committing code. The code is published to a package server only after verification. For a code to be released into production, validation tests have to be conducted to ensure that the code meets the business requirements. At every phase, success and failure scenarios are shown in the diagram, and the developer receives a report. This feedback report is crucial in DevSecOps to ensure consistent and secure code.

V. Proposed Methodology

The process starts with a requirements collection phase, which tries to define project specifications. A comprehensive approach is provided at each step of the development lifecycle, from thorough threat modelling to identify potential threats, strong secure coding practices, to the use of multimodal security scans based on SAST, SCA, and DAST, to identify and fix vulnerabilities as precisely as possible once the code is written and deployed within a testing environment. Finally, a range of tests, including functional testing, unit testing, penetration testing, and manual testing, are carried out to discover and rectify flaws or vulnerabilities in the security.

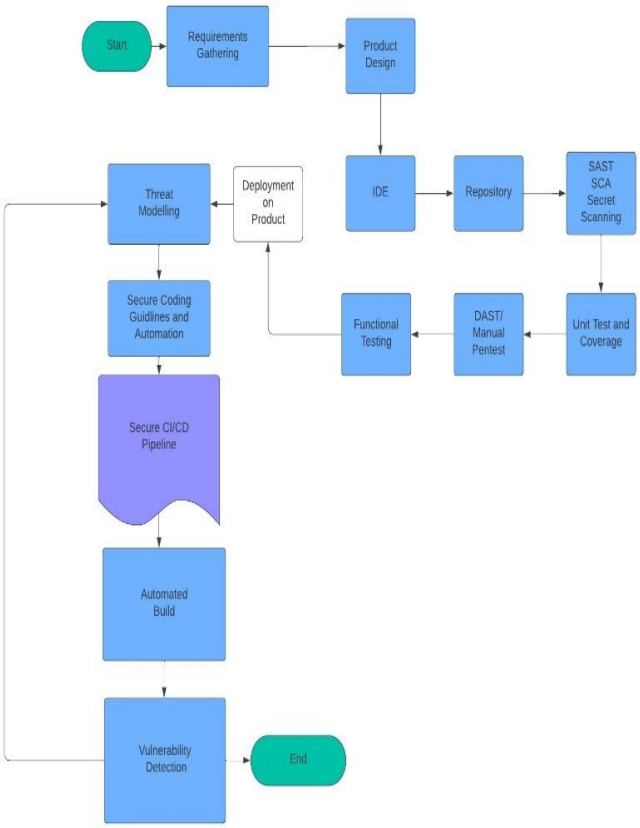

Fig. 3. Architecture of DevSecOps Lifecycle

A. Initial Setup and Deployment

Figure 3 shows launching an EC2 instance (T2 Large) and preparing the application setup. The code of the application then is cloned onto the EC2 instance. Then, the docker is setup on the EC2 instance which is followed by building and running the app on a docker container. Before we run the container; we get the API key necessary to run the application from TMDB (The Movie Database) and recreate the docker image with the API key.

B. Security

SonarQube and Trivy are run on the EC2 instance to scan for vulnerabilities. After installation, configure SonarQube to analyze the application code for quality and security issue and integrate SonarQube with the CI/CD pipeline.

C. Creating A CI/CD Pipeline

CI/CD pipeline with Jenkins for automation is set up by installing Jenkins on the preferred server or machine. Once installed, Jenkins is configured by installing necessary plugins such as Eclipse Temurin Installer for Java, SonarQube Scanner for code quality analysis, NodeJs Plugin for Node.js projects, and Email Extension Plugin for email notifications. After plugin installation, each tool is configured within Jenkins according to the project requirements, setting up JDK installations, SonarQube server connection, Node.js installations, and email configurations. Next, Jenkins webhook is created to trigger builds automatically. Finally, a

CI/CD pipeline in Jenkins is created, specifying stages for building, testing, deploying, and any other steps in the application deployment process. This pipeline will automate the entire software delivery process, from code changes to deployment, ensuring efficient and consistent releases.

D. Monitoring

The focus is currently on monitoring our application with Prometheus and Grafana. First, Prometheus, a popular open-source monitoring and alerting toolkit, is installed by creating a dedicated Linux user, downloading Prometheus binaries, and configuring the system unit file. Prometheus was later launched as a service, giving users access to its web interface to monitor metrics. In addition, Node Exporter is installed to collect system-level metrics and has been integrated with Prometheus for extensive monitoring. Grafana, a popular visualisation tool, is then installed on Ubuntu 22.04, allowing it to smoothly integrate with Prometheus. After adding the Grafana repository and installing the software, the Grafana service is enabled and launched. Grafana's web interface was used to connect to Prometheus as a data source and import a pre-configured dashboard for effective metric visualisation. With Prometheus and Grafana installed, the monitoring infrastructure is powerful and ready to ensure the application's health and performance.

E. Notification

Jenkins notification service has been configured.

F. Kubernetes

During the Kubernetes phase, a Kubernetes cluster with node groups is created to provide a scalable environment for application deployment and management. Additionally, Prometheus is used in monitoring a Kubernetes cluster in which Node Exporter is installed using Helm. Helm installs the Node Exporter upon adding the Prometheus Community Helm repository and creating a Kubernetes namespace. This component receives system-level metrics from the cluster nodes The Prometheus configuration (prometheus.yml) is modified with a new task for harvesting metrics from specific endpoints on the nodes. This enables effective monitoring of the cluster's health and performance. Finally, apps are deployed via ArgoCD, a Kubernetes-based continuous delivery solution.

G. Cleanup

Unused EC2 instances are terminated. This ensures that resources are not overutilized and are efficiently managed to reduce expenses.

VI. RESULTS

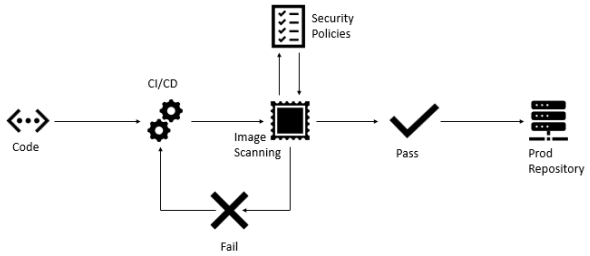

Fig. 3. Security Integration

Fig 3. describes the integration of security policies as part of the DevOps production cycle to make up a more resilient alternative compared to the current DevOps system.

This research explains the successful deployment of a web application within a cloud infrastructure. The research is focused on security, automation, and scalability and to build and enforce a solid CI/CD pipeline that promotes automated building, testing, and deployment of new features and updates to the web app. The web application and its dependencies are containerized and deployed through Docker, securely stored in Amazon Elastic Container Registry (ECR). This offers portability and consistency across multiple environments. Kubernetes, which is managed by Amazon Elastic Kubernetes Service (EKS), is used to efficiently orchestrate containerized components and offer autonomous scalability, self-healing, and high availability. Mechanisms of security throughout the pipeline are evaluated. These include Docker Security Scanning and Kubernetes security features such as Network Policies, Pod Security Policies, and RBAC for their effectiveness in protecting the web application from threats and unauthorized access. Finally, the research demonstrates a successful deployment of a secure, scalable, and resilient web application on the cloud adhering to the best practices in DevOps, cybersecurity, and cloud computing.

VII. FUTURE WORK AND CONCLUSIONS

The Primary focus of future research will be making the entire DevSecOps cycle more efficient. This means speeding up CI/CD pipelines and creating even tighter security measures and research could also focus on various cloud features to make applications more scalable and bringing costs down. Another area will be integrating Artificial Intelligence and Machine Learning that can help in improving the web application's reliability. Overall, more research on DevSecOps will create better cloud apps and software delivery systems. Future work on sustainability is also important which will prioritise sustainability and performance equally. While efficiency and resource management are important, measurements such as carbon footprint and water sage should also be considered in future development.

Another area where future work could focus on is creating more tools for DevSecOps. Works focusing on reducing

communication gaps and creating more secure coding standards along with more automation is also a possibility. However, despite these shortcomings DevSecOps has various positives which include less unplanned work, more automation, better cooperation, continuous improvement across operations, security teams and development.

Frameworks such as OWADP SAMM, incorporating security goals and KPIs. Implementation of security assurance programs and secure coding approaches. These are among the various tools that developers have to overcome certain challenges listed earlier and make DevSecOps easier to implement.

Sustainability is something that needs to integrated into DevSecOps to create efficient, robust and socially responsible development cycles. Organisations that prioritise resource efficiency, cost reduction, and ethical resource management can reduce waste and their carbon impact. This means reducing operating costs by utilising energy-efficient technologies and using various techniques that decrease energy consumption. Also, cutting material waste with lean manufacturing methods and streamlining production procedures lowers environmental impact while also making the purchase and disposal of materials cheaper.

Despite considerable progress, there is always need for additional study and development in DevSecOps implementation. These include further investigation into metrics for measuring DevSecOps effectiveness, additional case studies, security practice improvement, and the use of emerging technology to meet increasing security concerns.

DevSecOps research highlights the significance of incorporating security across the software development lifecycle.

VIII. REFERENCES

[1] Myrbakken, H. and Colomo-Palacios, R., 2017. DevSecOps: a multivocal literature review. In *Software Process Improvement and Capability Determination: 17th International Conference, SPICE 2017, Palma de Mallorca, Spain, October 4–5, 2017, Proceedings* (pp. 17-29). Springer International Publishing.

[2] Díaz, J., Pérez, J.E., Lopez-Peña, M.A., Mena, G.A. and Yagüe, A., 2019. Self-service cybersecurity monitoring as enabler for devsecops. *Ieee Access*, *7*, pp.100283-100295.

[3] Akbar, M.A., Smolander, K., Mahmood, S. and Alsanad, A., 2022. Toward successful DevSecOps in software development organizations: A decision-making framework. *Information and Software Technology*, *147*, p.106894.

[4] Hsu, T.H.C., 2018. *Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps*. Packt Publishing Ltd.

[5] Kumar, R. and Goyal, R., 2020. Modeling continuous security: A conceptual model for automated DevSecOps using open-source software over cloud (ADOC). *Computers & Security*, *97*, p.101967.

[6] 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*, *8*(23592365), pp.10-5281.

[7] Zunnurhain, K. and Duclervil, S.R., 2019, December. A new project management tool based on devsecops. In *2019 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 239-243). IEEE.

[8] Morales, J., Turner, R., Miller, S., Capell, P., Place, P. and Shepard, D.J., 2020. Guide to implementing devsecops for a system of systems in highly regulated environments.

[9] Gupta, A., 2022. An Integrated Framework for DevSecOps Adoption. *arXiv preprint arXiv:2207.04093*.

[10] IBM Report: Cost of a Data Breach Hits Record High During Pandemic. Available: https://newsroom.ibm.com/2021-07-28-IBM-Report-Cost-of-a-Data-Breach-Hits-Record-High-During-Pandemic

[11] Puppet State of DevOps Report. Available: https://www.puppet.com/resources/state-of-devops-report

[12] Prates, L., Faustino, J., Silva, M. and Pereira, R., 2019. Devsecops metrics. In *Information Systems: Research, Development, Applications, Education: 12th SIGSAND/PLAIS EuroSymposium 2019, Gdansk, Poland, September 19, 2019, Proceedings 12* (pp. 77-90). Springer International Publishing.

[13] Singh, T., 2019, July. Constrained Markov decision processes for intelligent traffic. In 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT) (pp. 1-7). IEEE.

[14] Vinod, D.N. and Singh, T., 2018, May. Autonomous agricultural farming robot in closed field. In 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 1-7). IEEE.

[15] Singh, T. and Darshan, V.S., 2015, October. A modern data architecture with apache Hadoop. In 2015 International Conference on Green Computing and Internet of Things (ICGCIoT) (pp. 574-579). IEEE.

[16] 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, 2(1), 79-89.

[17] Gonçalves, D. H. A. (2022). DevSecOps for web applications: a case study (Doctoral dissertation, Instituto Politecnico do Porto (Portugal)).

[18] Sojan, A., Rajan, R., & Kuvaja, P. (2021, November). Monitoring solution for cloud-native DevSecOps. In 2021 IEEE 6th International Conference on Smart Cloud (SmartCloud) (pp. 125-131). IEEE.

[19] Alawneh, M., & Abbadi, I. M. (2022, December). Expanding DevSecOps Practices and Clarifying the Concepts within Kubernetes Ecosystem. In 2022 Ninth International Conference on Software Defined Systems (SDS) (pp. 1-7). IEEE.

[20] Putra, A. M., & Kabetta, H. (2022, October). Implementation of DevSecOps by Integrating Static and Dynamic Security Testing in CI/CD Pipelines. In 2022 IEEE International Conference of Computer Science and Information Technology (ICOSNIKOM) (pp. 1-6). IEE