Dr. Vishwanath Karad MIT WORLD PEACE UNIVERSITY | PUNE

TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

School of Computer Science & Engineering

Department of Computer Science and Applications

2025-2026

Synopsis

On

“Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps”

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Project Based Learning

Course Code: BSC2PR01A

Second Year BSc Computer Science

Year: 2024-2025

Group Id:

Team Leader:

S NO.	NAME	PRN
1	Rohit Bhandari	1132231192
2	Unnatti Agarwal	1132231001
3	Mahima Patel	1132230921
4	Sandesh Amunekar	1132231365
5	Atharva Wani	1132230391
6	Anushka Ajabe	1132230973

Project Title: "Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps"

Name of the Mentor : Dr. Basant Tiwari

Introduction

1. Brief overview of the project

In the modern software industry, rapid development cycles often overlook security, leading to applications that may be functional but vulnerable to cyber threats. This project addresses that gap by designing a Secure Software Development Lifecycle (SDLC) framework that integrates security into every phase of development rather than treating it as a final step.

The framework is based on the DevSecOps philosophy—Development, Security, and Operations—which promotes a “shift-left” approach to security. This means anticipating threats, embedding safeguards, and using automated checks from the earliest stages of development, such as requirement analysis and system design, through coding, testing, deployment, and ongoing maintenance.

The project’s main thrust is to create a practical, easy-to-follow, and cost-effective security model that can be adopted by small development teams and beginners without needing enterprise-level resources. It demonstrates not only the conceptual stages of secure development but also real-world tools that can be seamlessly integrated into day-to-day workflows.

Key Objectives:

Develop a step-by-step Secure SDLC model with defined security tasks at each stage.

Enhance awareness and understanding of common vulnerabilities like SQL injection, XSS, weak authentication, and insecure APIs.

Demonstrate hands-on integration of widely used DevSecOps tools such as:

GitHub & GitHub Actions for collaborative version control and secure CI/CD.

Bandit for Python security scanning.

SonarCloud for bug and vulnerability detection.

Draw\.io/Lucidchart for creating lifecycle diagrams.

Provide a demo pipeline showing automated security checks before deployment.

Promote continuous monitoring and timely patching for post-deployment safety.

Scope:

The focus is on security-first design, not on building a large-scale product. Each SDLC phase—Requirement Analysis, Design, Implementation, Testing, Deployment, and Maintenance—will be studied, documented, and equipped with security guidelines. Simulations and small coding examples will be used to demonstrate these practices in a realistic, but lightweight, environment.

Expected Outcomes:

By the conclusion of the project, the team will deliver:

A comprehensive Secure SDLC framework suitable for small teams and educational purposes.

Demonstrations showing how security can be embedded from day one to save costs and reduce future risks.

A clear, visual model that simplifies complex security principles into actionable steps.

Evidence that secure development can be achieved without high costs or complex infrastructure.

This project ultimately encourages a smarter, more responsible approach to software development—producing applications that are not just feature-rich, but also robust against modern cyber threats.

2.Background of the Project

In today’s technology-driven world, software development is happening faster than ever to meet the growing demands of users and industries. While this speed brings innovation, it also increases the risk of overlooking security. Traditionally, security measures were considered only towards the end of the Software Development Lifecycle (SDLC), often just before deployment. This late-stage approach leaves applications vulnerable, as potential weaknesses are identified too late and may already be exploitable by attackers.

The increasing frequency of cyberattacks, data breaches, and ransomware incidents has made it clear that building security into software from the beginning is no longer optional—it is essential. High-profile security failures have shown that ignoring security during early development stages can lead to severe financial losses, legal consequences, and damage to public trust.

To address this challenge, modern software engineering is shifting towards DevSecOps, which stands for Development, Security, and Operations. DevSecOps promotes a “shift-left” strategy, where security is integrated into every stage of the development process—from requirement gathering and system design to coding, testing, deployment, and maintenance. This proactive approach ensures that vulnerabilities are identified and mitigated early, reducing costs, improving reliability, and enhancing overall application safety.

This project is rooted in the need to make secure software development accessible and practical, even for small teams or educational environments. By designing a Secure SDLC framework and demonstrating it with easily available tools like GitHub Actions, Bandit, and SonarCloud, the project bridges the gap between theory and practice. It aims to show that secure development is not only for large corporations but can be achieved by anyone willing to follow structured, security-focused practices from day one.

3.1 Domain

Cybersecurity — DevSecOps & Secure SDLC

Focus on embedding security controls across the software lifecycle (requirements → design → build → test → deploy → operate).

3.2 Technologies Used

Programming & Repo

Python (example app & scripts)&x20;

Git & GitHub (version control, issues, PRs)&x20;

CI/CD & Automation

GitHub Actions (pipeline to automate build, test, and security checks)&x20;

Application Security (AppSec)

SAST: Bandit for Python secure code scanning&x20;

Code Quality & Vulnerability Detection: SonarCloud&x20;

(Optional add-ons you can include)

SCA: pip-audit / GitHub Dependency Review

DAST: OWASP ZAP for runtime testing

Secrets Scanning: Gitleaks / GitHub Secret Scanning

Design & Documentation

draw\.io / Lucidchart for Secure SDLC and threat-model diagrams&x20;

(Optional Infrastructure & Ops)

Docker for containerized builds

Trivy or Grype for container/image scans

Checkov for IaC (Terraform/YAML) policy checks

GitHub Container Registry / Releases for artifact management

4.**Problem Statement**

Modern software development increasingly adopts agile and DevOps practices to deliver features rapidly. However, this speed often comes at the expense of security, leading to vulnerabilities being detected late in the lifecycle or post-deployment.

Traditional SDLC models treat security as a separate phase, typically addressed only during testing or after release, which increases the risk of costly fixes, compliance failures, and exploitation by malicious actors.

There is a need for an integrated approach—DevSecOps—that embeds security practices and automated checks throughout the SDLC to ensure secure, high-quality software without sacrificing development speed.

Research Gap

While DevSecOps has emerged as a promising approach, several challenges remain:

1. Fragmented Practices – Existing frameworks often focus on either development speed or security, but rarely balance both effectively in real-world agile/CI-CD environments.

2. Limited Standardization – There is no universally accepted, standardized Secure SDLC model tailored specifically for DevSecOps pipelines.

3. Tool Integration Complexity – Multiple security tools (SAST, DAST, SCA) exist, but seamless integration into CI/CD workflows remains inconsistent and resource-intensive.

4. Insufficient Automation – Many security checks are still performed manually, slowing development and leaving gaps in early-stage detection.

5. Lack of Empirical Evaluation – Few studies provide comprehensive case studies or metrics proving the impact of an integrated Secure SDLC framework in DevSecOps on both security and delivery speed.

This research addresses these gaps by designing and evaluating a holistic, automated Secure SDLC framework optimized for DevSecOps environments.

5. Objectives

The primary aim of this project is to create a structured, secure, and practical Software Development Lifecycle (SDLC) framework that seamlessly integrates security at every stage, following the DevSecOps philosophy. The expanded objectives are:

Develop a Secure SDLC Framework

Formulate a step-by-step methodology with clear guidelines, best practices, and checkpoints for incorporating security into all phases of development.

Promote Security Awareness Among Developers

Educate team members and new developers about common vulnerabilities such as SQL injection, cross-site scripting (XSS), insecure dependencies, and improper authentication.

Integrate Security Early (“Shift-Left” Approach)

Ensure security considerations begin at the requirement-gathering stage and continue through design, coding, testing, deployment, and maintenance.

Evaluate and Implement DevSecOps Tools

Research, test, and demonstrate open-source or free-to-use tools (e.g., Bandit, SonarCloud, GitHub Actions) to make security integration practical and cost-effective.

Provide Realistic Demonstrations

Deliver an example mini-project or simulation to showcase how the Secure SDLC framework works in practice.

Ensure Scalability and Accessibility

Design the framework so that small teams, startups, and academic projects can adopt it without heavy infrastructure or costs.

5.2 Scope

The scope of this project is limited to developing and demonstrating a Secure SDLC framework rather than building a complete enterprise-scale application. The framework will be designed with adaptability in mind, making it suitable for both small-scale and future large-scale software projects.

The project scope includes:

Requirement Analysis Stage

Identifying functional and security requirements before coding begins.

Defining compliance standards and possible threat scenarios.

Design Stage

Applying secure design principles such as the Principle of Least Privilege, threat modeling, and defense-in-depth.

Creating architecture diagrams using tools like Draw.io or Lucidchart with security considerations annotated.

Implementation Stage

Following secure coding standards (e.g., OWASP guidelines).

Using static code analysis tools like Bandit and SonarCloud to detect vulnerabilities early.

Testing Stage

Automating functional and security testing in the CI/CD pipeline.

Conducting penetration testing on a small scale to validate the security measures.

Deployment Stage

Implementing CI/CD pipelines with integrated security checks using GitHub Actions.

Ensuring secure configurations in deployment environments.

Maintenance Stage

Regularly applying patches, updates, and security audits.

Monitoring for emerging vulnerabilities and updating security measures accordingly.

Limitations of Scope

No large-scale production deployment.

Focus remains on methodology, tools, and demonstration rather than full application development.

6.Methodology / Implementation Plan

The development of the Secure Software Development Lifecycle (SDLC) framework for DevSecOps will be carried out through a structured and systematic approach. The implementation plan is divided into well-defined phases to ensure clarity, consistency, and effectiveness in achieving the project objectives.

Phase 1 – Requirement Analysis

In this initial stage, both functional and security-related requirements will be identified by studying the problem domain and understanding the current limitations of existing development practices. Relevant security guidelines such as OWASP SAMM and NIST SP 800-218 will be reviewed. This phase will also involve analyzing traditional SDLC models and modern DevOps workflows to determine integration points for security practices.

Phase 2 – Framework Design

A detailed Secure SDLC model will be designed, embedding security activities at every stage of the lifecycle. The design will map DevSecOps principles to each SDLC phase as follows:

Requirements: Security requirement gathering and threat modeling.

Design: Secure architecture design, risk assessment, and compliance considerations.

Implementation: Adoption of secure coding practices and code reviews.

Testing: Integration of automated security testing tools for Static Application Security Testing (SAST), Dynamic Application Security Testing (DAST), and Software Composition Analysis (SCA).

Deployment: Security checks for Infrastructure-as-Code, secrets management, and container security.

Maintenance: Continuous monitoring, logging, and incident response processes.

Workflow diagrams and process models will be prepared to visualize the framework.

Phase 3 – Tool Selection and Integration

Appropriate tools will be selected based on compatibility, automation capabilities, and ease of integration into the CI/CD pipeline:

Version Control & Collaboration: Git and GitHub.

CI/CD Automation: GitHub Actions for pipeline orchestration.

Security Tools:

SAST: Bandit, SonarCloud.

SCA: pip-audit, GitHub Dependency Review.

DAST: OWASP ZAP.

Secrets Scanning: Gitleaks or GitHub Secret Scanning.

These tools will be integrated into the CI/CD pipeline to ensure that every code commit undergoes automated security checks before moving to the next stage.

Phase 4 – Implementation

A sample Python-based application will be developed to serve as the project's case study. The designed Secure SDLC framework will be implemented on this application with the following pipeline stages:

1. Code Commit and Build

2. Automated Security Scanning (SAST, SCA, Secrets)

3. Unit and Integration Testing

4. Dynamic Security Testing (DAST) in a staging environment

5. Secure Deployment

Phase 5 – Testing and Evaluation

The framework will be evaluated based on two key performance areas:

Security Impact: Reduction in vulnerabilities, earlier detection of flaws, and improved compliance.

Development Efficiency: Deployment frequency, build success rate, and reduced time-to-fix for security issues.

Results will be compared against a baseline DevOps pipeline without integrated security practices.

<u>Phase 6 – Documentation and Finalization</u>

All configurations, tools, and processes will be documented for future replication. A final report will present the complete Secure SDLC framework, evaluation metrics, and recommendations for further improvement.

7.Expected Outcome

The implementation of the proposed Secure Software Development Lifecycle (SDLC) framework for DevSecOps is expected to deliver the following outcomes:

1. Integrated Security Across the Lifecycle

Security practices embedded at every stage of the SDLC, ensuring early detection and mitigation of vulnerabilities.

2. Improved Software Quality

Reduction in the number and severity of security flaws through automated static, dynamic, and dependency analysis.

3. Increased Development Efficiency

Faster delivery cycles with minimal disruption to the development workflow due to automated and continuous security checks in the CI/CD pipeline.

4. Seamless Tool Integration

Effective incorporation of security tools (SAST, DAST, SCA, secrets scanning) within the development pipeline without adding significant overhead.

5. Enhanced Compliance and Risk Management

Alignment with recognized security standards and guidelines (e.g., OWASP SAMM, NIST SP 800-218), improving readiness for compliance audits.

6. Replicable Framework

A documented, reusable Secure SDLC model that can be adapted to different projects and organizational needs.

7. Measurable Security Improvements

Quantifiable metrics demonstrating improvements in vulnerability detection rates, remediation times, and overall application security posture.

8. Work Plan / Timeline

The project will be executed over a planned duration, following a phase-wise schedule to ensure systematic progress and timely completion. The tentative timeline is as follows:

Phase	**Activity**	**Duration**	**Timeframe**
Phase 1	**Requirement Analysis**	**1 week**	**Week 1**
Phase 2	**Framework Design**	**1 week**	**Week 2**
Phase 3	**Tool Selection and Integration Planning**	**1 week**	**Week 3**
Phase 4	**Implementation of Sample Application and Pipeline Setup**	**2 weeks**	**Weeks 4–5**
Phase 5	**Testing and Evaluation of Framework**	**2 weeks**	**Weeks 6–7**
Phase 6	**Documentation and Finalization**	**1 week**	**Week 8**

● **Total Duration: 8 weeks**

● **Notes:**

- Overlaps between phases may occur to optimize time utilization.
- Regular progress reviews will be conducted at the end of each phase to ensure alignment with objectives.
- Any technical issues or tool compatibility challenges will be addressed promptly to avoid delays.

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, *Rohit Sandip Bhandari, 1132231192* solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune   Name and Signature of
Student: Rohit Sandip Bhandari

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, *Unnatti Agarwal, 1132231001* solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune Name and Signature of
Student : *Unnatti Agarwal*

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, Atharva Wani, 1132230391 solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune Name and Signature of
Student: Atharva Wani

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, Mahima Patel, 1132230921 solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune   Name and Signature of
Student: Mahima Patel

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, *Sandesh Sandip Amunekar, 1132231365* solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune Name and Signature of
Student: Sandesh Sandip Amunekar

Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

Declaration

This declaration is part of the Synopsis for project work entitled Developing a Secure Software Development Lifecycle (SDLC) Framework for DevSecOps is submitted as part of academic requirement for Semester 5 of BSc CS to the Dr. Vishwanath Karad MIT World Peace University.

I, *Anushka Ajabe, 1132230973* solely declare that

1. I will not use any unfair means to complete the project.
2. I will follow the discipline and the rules of the organization where I am doing the project.
3. I will not do or be part of any act which may impact the college reputation adversely.

The information I have given is true, complete and accurate. I understand that failure to give truthful, incomplete and inaccurate information may result in cancellation of my project work.

Date :13/08/2025
Place: Pune   Name and Signature of
Student: *Anushka Ajabe*