Software Code Standard Improvement Automation

Prepared By:
Kaustubh Hemang Pandya 002344302
Nikunjkumar Khandar - 002347606
Priyank Nilesh Dabhi - 002348509

Introduction: -

In the world of software development, code standards are crucial for maintaining consistency and readability in codebases. However, manual enforcement of these standards poses significant challenges, including time-consuming processes and increased risk of errors. As codebases grow, ensuring adherence to standards becomes even more challenging, impacting productivity and increasing debugging time. To address these challenges, our proposed project focuses on developing an automated solution for code standards enforcement. By leveraging automation, we aim to streamline the enforcement process, enhance collaboration among developers, and ultimately improve software quality.

Project Objectives: -

The project objective is to develop an automated enforcement mechanism to ensure consistency and readability in software development by adhering to code standards. This automation addresses challenges associated with manual enforcement, such as time-consuming processes and error-prone methods. By enhancing collaboration among developers and simplifying code maintenance, the project aims to improve productivity and reduce debugging time, ultimately optimizing software development processes.

Description and Methodology: -

Requirement Analysis: Conduct a comprehensive analysis of existing code standards and practices within the organization or industry.

Iterative Development Process: Adopt an Agile software development methodology, such as Scrum or Kanban, to facilitate iterative development and frequent collaboration with team.

System Design: Design the architecture and functionality of the automated enforcement system based on the gathered requirements and stakeholder input.

Architecture and Development and Testing: Implement the automated enforcement system using suitable programming languages and technologies, following best practices for software development and coding standards.

Assumption: The selected static code analysis tool (e.g., ESLint, Pylint, or SonarQube) can analyze code written in various programming languages and frameworks commonly used in the project.

Hypothesis: By incorporating automated code analysis and enforcement into the CI/CD pipeline, the development team will benefit from continuous feedback on code quality, leading to early detection and resolution of issues, improved collaboration, and faster delivery of high-quality software releases

Resources: -

Static Code Analysis Tools,Version Control System (VCS), Continuous Integration (CI) Server, and Standard development machines with sufficient processing power and memory are sufficient for development and testing purposes

Learning Experience

1.Technical Skills Enhancement **2.** Understanding Software Development Practices **3.** Problem-Solving **4.** Innovation and Professional Growth

Deliverables:

1. Static code analyzer **2.** Code formatting **3.** Code review **4.** Continuous integration **5.** Report

References:

https://www.codegrip.tech/productivity/using-code-automation-strategy-to-improve-software-development/
https://www.perforce.com/blog/sca/enforce-coding-standards-automated-static-analysis
https://www.codemotion.com/magazine/backend/automation-coding-best-practices/