Secure Software Design Project Proposal

CY321

Secure Software Design

Muhammad Younas 2022456

Taha Juzar 2022585

Ahmer Ayaz 2022070

Bashir Ahmad 2022646

Faculty: **Cyber Security**

"On my honor, as student oath Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, I have neither given nor received unauthorized assistance on this academic work."

Submitted to:

Dr Zubair Ahamd

Proposal for Code Sentinel: An AI-Driven Secure Code Review Tool

Project Overview:

The advanced security solution code Sentinel relies on AI to realize its mission of improving codebase security through vulnerability detection and secure coding standards regulation. The system applies artificial intelligence for extensive security evaluations and identifies dangers to stop possible threats before attackers can exploit them. Secure applications can be built efficiently through builders' access to detailed reports and practical recommendation features within this tool.

Objectives:

- An AI system should be built to detect vulnerabilities in codebases through focused OWASP Top 10 security risk analysis.
- The platform should include a best practices evaluation system which helps developers write safer code.
- The system generates security reports that present details about identified weaknesses and their severity and suggested solutions.
- The tool will provide both a command-line interface and a web-based application which guarantees convenient usage.
- Continued improvement of the software relies on an open-source community platform to collect ongoing enhancement

1. **Vulnerability Detection:**
    - Identifies critical security issues such as SQL injection, XSS, buffer overflows, and remote code execution.
    - Special focus on OWASP Top 10 vulnerabilities.
2. **Secure Coding Practices Evaluation:**
    - Analyzes code for best practices in authentication, input validation, access control, and error handling.
3. **Detailed Security Reports:**
    - Provides reports detailing vulnerabilities, severity levels, affected code snippets, and remediation steps.
4. **Mitigation Guidance:**

    - Offers step-by-step instructions and secure code refactoring examples to resolve identified issues.

5. **Dual Mode Usability:**
    - **Command Line Interface (CLI):** Allows users to scan code repositories via terminal commands.
    - **Web Application:** Provides an interactive dashboard for enhanced user experience.

Technical Stack:

- **Programming Language:** Python
- **Frameworks:** Stream lit (for web UI), Scikit-learn (AI-based analysis)
- **Security Standards:** OWASP Guidelines, Secure Coding Best Practices
- **Deployment:** Docker, Virtual Environments

Implementation Plan:

1. **Phase 1: Research & Planning**
    - Define security criteria and AI-based detection models.
    - Establish CLI and web application structure.
2. **Phase 2: Development**
    - Implement vulnerability detection mechanisms.
    - Develop secure coding analysis and report generation features.
3. **Phase 3: Testing & Refinement**
    - Conduct security assessments on real-world repositories.
    - Optimize AI detection algorithms and improve UI/UX.
4. **Phase 4: Deployment & Documentation**
    - Deploy as an open-source project with clear documentation.
    - Encourage community contributions and enhancements.

Expected Outcomes:

- A robust, AI-driven security tool that helps developers identify and mitigate code vulnerabilities.
- Enhanced awareness and adoption of secure coding practices.
- A growing community of developers contributing to the project for continuous improvements.

Target Audience:

- Software Developers
- Security Analysts
- Open-Source Community
- Enterprises seeking automated security code reviews

Conclusion:

code Sentinel aims to bridge the gap between software development and security by providing an automated, AI-enhanced tool for secure code reviews. By integrating cutting-edge vulnerability detection with actionable recommendations, Code Sentinel will serve as an invaluable asset for developers and organizations striving to build secure applications efficiently.