Deliverables and Marking Criteria

1. Release Version: Publish a new version of the application incorporating all fixes. (1%) (Make sure you give us the link in the report to your fixed version of application.)
2. Provide detailed documentation and commentary for each security issue identified in the SonarQube analysis (Vulnerabilities and Security Hotspots only(6 at least). (2%)

Sample

Issue 1: SQL Injection Vulnerability

- **Description:** Potential SQL Injection vulnerability in *UserController.java* line 45.
- **Severity:** Critical
- **Effort:** 20 minutes
- **Status:** Fixed
- **Type:** Vulnerability
- **Severity:** Critical
- **Status:** Fixed
- **Assigned to:** Team member name
- **CWE:** CWE-327: Formatting SQL queries is security-sensitive
- **Clean code attribute:** Responsibility | Not trustworthy
- **Software qualities impacted:** Security
- **False Positive:** YES/NO

Analysis:

- The code was found to concatenate user input directly into SQL queries, which can lead to SQL injection attacks.

Action Taken:

- Refactored the code to use prepared statements with parameterized queries to prevent SQL injection.
- Reviewed all similar code sections to ensure no other instances of this issue exist.

Commentary:

- Using prepared statements ensures that user input is properly sanitized and avoids the risk of SQL injection.
- This change improves the security posture of the application by eliminating a critical vulnerability.

3. A zip file comprising: (3% will be evaluated based on the pre- and post-fixes reports and sessions analysis)
    - The ZAP session(s) containing all the scans performed.
    - Zap Scanning HTML Reports of the Pre- and Post-fixes.
4. A report containing this and above: (6%, 1% each)

- Screenshots from SonarQube showcasing changes in issues over time. (You may use http://localhost:9000/extension/cayc/stats one as well.
- A detailed summary of the penetration testing process, including descriptions and screenshots of scans and tests performed.
- An analysis of the most severe vulnerabilities identified by ZAP.
- A comparison of vulnerabilities detected by both tools, explaining discrepancies and the role of each tool in a thorough security evaluation.
- Threat/Vulnerabilities matrix. (See Table of Contents > Chapter 7. Software Security Testing > Slides)
- A brief discussion (one to two paragraphs) on additional tools and techniques that could further enhance the security testing of the application.