Security Regulatory Compliance Report

# Security Compliance Report

## Executive Summary This report details the findings of a security compliance assessment conducted on the \*\*June-Voice-AI\*\* repository. The assessment identified several vulnerabilities of medium severity, while no critical or high-risk issues were discovered. The primary areas of concern relate to the lack of specified timeouts in outbound HTTP requests, which exposes the application to potential Denial of Service (DoS) attacks. These findings impact compliance with key security standards, including the OWASP Top 10, PCI DSS, and SOC 2, particularly concerning secure application development and service availability.

Overall, the application has a moderate security posture. Immediate remediation of the identified medium-priority findings is recommended to mitigate risks and improve the application's compliance status. This report provides a detailed breakdown of all findings, their compliance implications, and a clear roadmap for remediation.

## Compliance Overview - \*\*Assessment Date\*\*: 2025-10-12 - \*\*Repository\*\*: June-Voice-AI - \*\*Sector\*\*: SaaS Platform

## Compliance Standards Assessment

### OWASP Top 10 Compliance The assessment revealed partial compliance with the OWASP Top 10 2021 framework. The identified medium-severity vulnerabilities, related to making network requests without a timeout, map to the following OWASP categories:

- \*\*A04:2021 - Insecure Design\*\*: The application's design does not adequately protect against resource exhaustion caused by unresponsive external services. The lack of a timeout mechanism in network calls is a design flaw that can lead to Denial of Service (DoS), making the system unavailable.  
- \*\*A10:2021 - Server-Side Request Forgery (SSRF)\*\*: While not a direct SSRF vulnerability, the uncontrolled nature of the outbound requests could be a contributing factor in more complex attack chains. Ensuring requests have strict controls, like timeouts, is a fundamental defense-in-depth practice against SSRF-related impacts.

\*\*Status: Partially Compliant.\*\* Remediation is required to address insecure design patterns.

### PCI DSS Requirements The findings have implications for compliance with the Payment Card Industry Data Security Standard (PCI DSS), even if the application does not directly process cardholder data. The principles of secure development are universal.

- \*\*Requirement 6: Develop and Maintain Secure Systems and Applications\*\*: This requirement mandates the protection of systems against known vulnerabilities. The identified findings violate secure coding best practices outlined in \*\*Req 6.5.10\*\*, which focuses on protecting against injection flaws and other common vulnerabilities (as informed by standards like the OWASP Top 10). A DoS vulnerability can impact the availability of the system, which is a core security tenet.

\*\*Status: Partially Compliant.\*\* The application does not fully adhere to secure coding guidelines necessary to protect system availability.

### SOC 2 Controls The vulnerabilities impact the Trust Services Criteria for a SOC 2 audit, specifically related to Availability.

- \*\*Availability (A1.2)\*\*: This criterion requires the entity to have controls in place to protect against events that could impair system availability. The identified vulnerabilities could allow an attacker or a failing external service to render the application unresponsive, directly violating the principle of maintaining system availability and recoverability. The lack of timeouts is a failure to implement controls that manage system resources and performance.

\*\*Status: Partially Compliant.\*\* The current implementation presents a risk to service availability, a key control objective under SOC 2.

## Critical Findings No critical findings were identified during this assessment.

## High Priority Findings No high-priority findings were identified during this assessment.

## Medium Priority Findings Three medium-priority findings were identified. All relate to the use of the `requests` library without a specified timeout, creating a risk of Denial of Service (DoS).

| Finding ID | Vulnerability Description | File Path | Line Number |  
| :--- | :--- | :--- | :--- |  
| 1 | \*\*Request without Timeout\*\*: The application makes an HTTP POST request to the Kokoro TTS service without a timeout. A slow or non-responsive server at `localhost:8880` could cause this process to hang indefinitely. | `june.py` | 88 |  
| 2 | \*\*Request without Timeout\*\*: The application makes an HTTP POST request to the recording API endpoint without a timeout. If the recording service is unavailable or slow, this call could block indefinitely, freezing the application. | `june.py` | 112 |  
| 3 | \*\*Request without Timeout\*\*: The application makes an HTTP POST request to the transcription API endpoint without a timeout. This creates a DoS risk if the transcription service fails to respond in a timely manner. | `june.py` | 120 |

## Remediation Roadmap The following roadmap outlines the steps to address the identified vulnerabilities. These actions should be prioritized to enhance the security and reliability of the application.

| Priority | Vulnerability | Recommendation | Affected Components | Estimated Effort |  
| :--- | :--- | :--- | :--- | :--- |  
| \*\*Medium\*\* | Lack of Timeout in HTTP Requests | Add an explicit `timeout` parameter to all `requests.post()` calls. A reasonable value (e.g., `timeout=15` seconds) will prevent the application from hanging on unresponsive network calls. | `june.py` (Lines 88, 112, 120) | \*\*Low\*\* |

\*\*Example Implementation:\*\*  
```python  
# Before  
response = requests.post(T\_API\_URL)

## After response = requests.post(T\_API\_URL, timeout=15) # 15-second timeout ```

## Compliance Status Summary \*\*Overall Status: Partially Compliant\*\*

The June-Voice-AI application is partially compliant with major security standards. While it is free of critical and high-severity vulnerabilities, the presence of medium-severity issues related to insecure service integration poses a tangible risk to service availability. These vulnerabilities must be remediated to achieve a higher level of compliance with OWASP, PCI DSS, and SOC 2 requirements for secure and resilient systems.

## Next Steps and Recommendations 1. \*\*Immediate Remediation\*\*: Implement the fixes detailed in the Remediation Roadmap, specifically by adding timeout parameters to all external `requests` calls in `june.py`. This is the highest priority action item. 2. \*\*Secure Coding Review\*\*: Conduct a broader review of the codebase to identify other instances of insecure practices, particularly around network communication, error handling, and input validation. 3. \*\*Automated Security Scanning\*\*: Integrate a Static Application Security Testing (SAST) tool like Bandit into the development pipeline. This will help identify similar vulnerabilities automatically before they are introduced into production code. 4. \*\*Verification Scan\*\*: After applying the recommended fixes, perform a follow-up security scan to confirm that the vulnerabilities have been successfully resolved and no new issues have been introduced.