# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **May 23, 2025** | **Jean Karst** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In this report, identify your security vulnerability findings and recommend the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also include images or supporting materials. If you include them, make certain to insert them in the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Jean Karst

**1. Interpreting Client Needs**

Artemis Financial is transitioning to a modernized infrastructure using a RESTful web API for its consulting services in savings, retirement, investments, and insurance. As financial data is highly sensitive, the company requires robust security to protect user confidentiality, maintain data integrity, and ensure availability.

Key Security Needs:

* Secure Communications: Encryption is essential to protect customer data during transmission
* International Transactions: If Artemis handles foreign investments or clients, compliance with data privacy regulations such as GDPR or international banking standards may be necessary.
* Government Restrictions: Compliance with U.S. laws and regulations should be considered.
* External Threats: Artemis may be exposed to threats such as SQL injection, XSS, CSRF, API abuse, and insecure authentication/authorization mechanisms.
* Modernization Requirements:
  + Use of open-source libraries must be vetted for vulnerabilities
  + Adoption of modern frameworks should include secure default configurations.
  + RESTful APIs should follow OWASP API Security Top 10.

**2. Areas of Security**

Based on the vulnerability assessment process flow and the nature of the Artemis Financial application, the following areas apply:

* Input Validation: Crucial to prevent injection attacks; user data must be validated both client-side and server-side.
* Secure Input and Representations: Required to ensure input data does not get misrepresented or transformed into unsafe formats.
* Architecture Review: Necessary to assess secure design patterns, microservices use, and API gateway protections.
* Code Review: Must check for hardcoded credentials, unhandled exceptions, logging issues, and business logic flaws.
* Secure Coding Practices: Ensure use of proper design patterns, avoid code duplication, and maintain separation of concerns.
* Cryptography: Must use strong encryption algorithms for data in transit and at rest; assess for proper key management.
* Secure API Interactions: Important due to RESTful architecture; rate limiting, authentication, and CORS policies need review.

**3. Manual Review**

A manual review of the Project One Code Base revealed several serious security issues. First, the application does not use input validation, which means user data is not being checked or sanitized before being processed—this opens the door to attacks like SQL injection or cross-site scripting. Additionally, there is no exception or error handling in place, which can lead to crashes or expose sensitive system information to users. Another major concern is the lack of encryption—data is not being secured during storage or transfer. Finally, the project includes outdated dependencies, such as Bouncy Castle and Spring Boot, both of which have known security flaws in the versions used. These vulnerabilities together suggest the application needs major improvements in security to protect user data and maintain system integrity.

**4. Static Testing**

**A white background with black text

AI-generated content may be incorrect.**

|  |  |  |  |
| --- | --- | --- | --- |
| Vulnerability Name or Code | Description | Recommended Solution | Attribution |
| CVE-2024-34447 | Improper Validation of Certificate with Host Mismatch. | Ensure hostname verification is enabled and performed correctly in SSL connections. | Documented by OSSINDEX and classified under CWE-297 by MITRE. Referenced in Bouncy Castle release notes. |
| CVE-2016-1000338 | Improper Verification of Cryptographic Signature. | Upgrade to Bouncy Castle version 1.56 or later to fix DSA encoding verification. | Identified by OSSINDEX and listed under CWE-347. Detailed in NVD and Bouncy Castle release notes. |
| CVE-2020-25649 | Improper Restriction of XML External Entity Reference. | Disable XXE processing or upgrade to a safe version. | Disclosed via GitHub issue tracker . Found in FasterXML’s GitHub advisories. |
| CVE-2023-1932 | XSS vulnerability via SafeHtmlValidator in Hibernate. | Avoid using unsafe HTML or sanitize input correctly. | Reported on GitHub Advisory Database. Associated with CWE-79. Reviewed in Sonatype’s OSSINDEX vulnerability platform. |
| CVE-2020-9488 | Improper validation of certificate in Apache Log4j. | Upgrade to 2.12.3 or 2.13.1. | Reported by Apache Security Team. Detailed in official Apache mailing list and advisories. |
| CVE-2023-6378 | Deserialization vulnerability in logback. | Avoid unsafe deserialization or upgrade to newer version | Reported in GitHub Advisory Database and NVD. Documented in logback release notes and advisories. |

**5. Mitigation Plan**

|  |  |
| --- | --- |
| Issue | Recommended Mitigation |
| Outdated Cryptographic Library | Upgrade to newer version; ensure SSL/TLS with hostname validation; implement secure key management. |
| Insecure Data Binding | Upgrade to newer version, |
| Logging Framework Risks | Use to latest versions |
| No Input Validation | Validate and sanitize all input on both client and server sides using whitelisting. |
| No Exception Handling | Implement try-catch blocks with sanitized error messages; log errors securely. |
| No Encryption | Use TLS 1.2+ for API transport, encrypting database fields with AES-256. |
| Outdated Spring Framework | Upgrade Spring Boot and related dependencies; enable secure config defaults. |
| Missing RESTful API Protections | Implement OAuth2/JWT, set CORS policies, add rate limiting and API gateway security. |

**Sources:**

**OWASP Foundation.** (2023). OWASP API Security Top 10 - 2023. https://owasp.org/www-project-api-security/