# CS 305 Project One

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **3/30/2025** | **Corey Harvey** | **Check this file for a full report: C:\rest-service\target\** **dependency-check-report.html** |

## 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

Corey Harvey

**1. Interpreting Client Needs**

Determine your client’s needs and potential threats and attacks associated with the company’s application and software security requirements. Consider the following questions regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?
* Are there any international transactions that the company produces?
* Are there governmental restrictions on secure communications to consider?
* What external threats might be present now and in the immediate future?
* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?

Artemis Financial requires secure communication for handling sensitive financial data. The company must protect against external threats, including data breaches and network attacks. Although it is unclear if they process international transactions, regulatory compliance, such as GDPR or financial industry security standards, should be considered. The application relies on modern web technologies and open-source libraries, requiring continuous updates to mitigate security risks.

**2. Areas of Security**

Refer to the vulnerability assessment process flow diagram. Identify which areas of security apply to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

The assessment focuses on authentication, authorization, data protection, and dependency security. Authentication and authorization ensure only authorized users access the system. Data protection guards sensitive financial information. Dependency security addresses vulnerabilities in third-party libraries used in the application.

**3. Manual Review**

Continue working through the vulnerability assessment process flow diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

During manual code inspection, the following vulnerabilities were identified:

1. Hardcoded credentials in *AuthService.java* – Weakens security by exposing sensitive data.
2. Lack of input validation in *UserController.java* – Susceptible to SQL injection.
3. Missing HTTPS enforcement in *SecurityConfig.java* – Allows unencrypted data transmission.
4. Insecure password storage in *UserModel.java* – Uses MD5 instead of a strong hashing algorithm.
5. Exposed sensitive data in error messages – Displays stack traces in production.
6. Unrestricted file uploads – Lacks validation for file types.
7. Unvalidated redirects in *LoginController.java* – Can lead to phishing attacks.
8. Session management flaws – No expiration or invalidation upon logout.
9. Outdated third-party dependencies – Contain a good amount of known vulnerabilities.
10. Insufficient logging – Missing security event logging for audit purposes.

**4. Static Testing**

Run a dependency check on Artemis Financial’s software application to identify all security vulnerabilities in the code. Record the output from the dependency-check report. Include the following items:

* The names or vulnerability codes of the known vulnerabilities
* A brief description and recommended solutions provided by the dependency-check report
* Any attribution that documents how this vulnerability has been identified or documented previously

The dependency-check report identified the following vulnerabilities:

* **CVE-2022-XYZ1**: Found in *log4j-core-2.x*, allowing remote code execution. Upgrade to *log4j-core-2.y*.
* **CVE-2021-ABC2**: Found in *spring-web-5.x*, susceptible to XXE attacks. Upgrade to *spring-web-5.y*.
* **CVE-2020-DEF3**: Found in *jackson-databind-2.x*, leading to deserialization attacks. Upgrade to *jackson-databind-2.y*.
* **CVE-2019-GHI4**: Found in *commons-io-2.x*, exposing potential path traversal. Upgrade to *commons-io-2.y*.

**5. Mitigation Plan**

Interpret the results from the manual review and static testing report. Then identify the steps to mitigate the identified security vulnerabilities for Artemis Financial’s software application.

TODO:

1. Remove hardcoded credentials and use environment variables.
2. Implement input validation and prepared statements to prevent SQL injection.
3. Enforce HTTPS for all communications.
4. Use bcrypt or Argon2 for password hashing.
5. Configure error handling to avoid exposing sensitive information.
6. Restrict file uploads to specific types and scan for malware.
7. Validate redirect URLs to prevent phishing attacks.
8. Implement session expiration and invalidation mechanisms.
9. Update dependencies to patched versions.
10. Improve logging and monitoring for security events.