# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **[11-13-2024]** | **[Ben Douglas]** |  |

## 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

[Ben Douglas]

**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?

[It’s important to have secure comms for Artemis financial, because they consult and develop individual financial plans for the customers like savings, retirement, investments, and insurance, and a lot of the times you are dealing with customers’ identities like social security numbers, date of births, driver’s license, and/or other identifications. Since they financially plan for customers, then Artemis financial would help customers around the globe to invest and/or retire thus resulting in global transactions. They are government restrictions that needs considering with secure comms inside of Artemis financial, because you don’t need the customers’ info to be breached, and you would need HIPAA to regulate the customers’ health data.

The external threats that could be in the present now and promptly in the future are cybercriminals searching for financial data, malicious actors attacking the functionality of the software application with DDoS, phishing the customers’ credentials, data breaching that is vulnerable in the RESTful API, insider threats that come from angry employees, and malware that gains access and acquires financial info. The modernization requirements are updating and patching outdated libraries, the libraries must obey the license agreements, using libraries from trusted sources, use automatic scanning of open-source dependencies, use OAuth 2.0 to authenticate, using HTTPS to encrypt comms, use API rate-limit to prevent abuses, use a Web Application Firewall, implement a secure software development lifecycle, and create an zero-trust security model.]

**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.

[An area of security that applies to the Artemis financial’s software application is Input Validation, because it prevents malicious data like SQL injection from being processed by the application. Another area is APIs, because it serves as an interface between the client and the server that handles sensitive data so implementing OAuth 2.0 would help secure the data. Another area is Cryptography, because encryption would make financial data that is sensitive more confidential and it would make it have integrity so I would use HTTPS over API comms. Another area is Code Error, because it will handle errors if an unauthorized user tries to access the application through input validation. The last area is Code Quality, because it helps the data from being exposed, it keeps unauthorized users from having access to the application, and it makes sure that the application functions properly.]

**3. Manual Review**

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

[The vulnerability of Input Validation is the GreetingController class uses the name parameter from the URL (@RequestParam(value = "name", defaultValue = "World") but it has no validation nor sanitization for this input. The vulnerability of the APIs is the /greeting has no authentication and anyone can access it through the public. The vulnerability of the Cryptography is in the DocData has no secure transmission of sensitive info. The vulnerability of the Code Error is in the DocData class that the exception is caught, but not handled that could lead to attackers getting sensitive info. The vulnerability of the Code Quality is in the DocData class connects to the database that has no secure connection pool nor proper exception handling, and hardcoding "root" could lead to a vulnerability. The vulnerability of Encapsulation is in the DocData class that exposes the id property directly with a getter method.

The vulnerability of DocData class in the read\_document method doesn’t use prepared statements nor parameterized queries.

**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 names or vulnerability codes of the know vulnerabilities are  [cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Abouncycastle&cpe_product=cpe%3A%2F%3Abouncycastle%3Alegion-of-the-bouncy-castle-java-crytography-api&cpe_version=cpe%3A%2F%3Abouncycastle%3Alegion-of-the-bouncy-castle-java-crytography-api%3A1.46)

, [cpe:2.3:a:vmware:spring\_boot:2.2.4:release:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Avmware&cpe_product=cpe%3A%2F%3Avmware%3Aspring_boot&cpe_version=cpe%3A%2F%3Avmware%3Aspring_boot%3A2.2.4),  [cpe:2.3:a:qos:logback:1.2.3:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aqos&cpe_product=cpe%3A%2F%3Aqos%3Alogback&cpe_version=cpe%3A%2F%3Aqos%3Alogback%3A1.2.3), and others but I only had so much room.

The description of them are **Description:**

The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7, **Description:**

Spring Boot, and **Description:**

logback-core module. The solutions for them are upgrade to the newest library of bouncy library 1.46, use security patches, check for compatibility and read the release notes of the newest library of bouncy library, check the national vulnerability database for patches or updates, review the security guidelines, use Java Cryptography Extension, update to the newest Spring Boot, examine the CVE details for solutions, use Dependabot for a new CVE and/or security patches, update the Logback 1.2.3 to 1.2.6 and 1.4.x, review the changelog and security advisories for release notes and security advisories on the Logback website and GitHub, and monitor for new vulnerabilities by checking the CVE database.

The attributions are The TLS implementation in the Bouncy Castle Java library before 1.48 and C# library before 1.8 does not properly consider timing side-channel attacks on a noncompliant MAC check operation during the processing of malformed CBC padding, which allows remote attackers to conduct distinguishing attacks and plaintext-recovery attacks via statistical analysis of timing data for crafted packets, a related issue to CVE-2013-0169, spring-boot versions prior to version v2.2.11.RELEASE was vulnerable to temporary directory hijacking. This vulnerability impacted the org.springframework.boot.web.server.AbstractConfigurableWebServerFactory.createTempDir method. NOTE: This vulnerability only affects products and/or versions that are no longer supported by the maintainer, and In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers.]

**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.

[Most of the vulnerabilities are from outdated versions. The steps to mitigate are implementing network segmentation and secure boundaries internal and external-facing services, use ORM frameworks for proper sanitization and parameterized queries, implement proper authentication or OAuth for control of access of sensitive APIs, use SSL/TLS for encryption of database connections and sensitive comms, implement proper exception handling for avoiding to expose sensitive info in error messages, refactor the code for better maintainability, implement role-based access control for database queries, apply mitigation workarounds that is suggested by the Bouncy Castle team or the security advisor, patch management, network restrictions, logging and monitoring, limit logback configuration, throttle log generation, and make sure logback isn’t exposed to untrusted sources.]