# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **3/22/2025** | **Julianne Takaya** |  |

## 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

Julianne Takaya

**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 is a financial consulting company that develops financial plans for its customers, including savings, retirement, investment, and insurance. As they deal with highly sensitive financial and personally identifiable information, all communication between the company and their clients must be secure. They will also need to adhere to any government regulations regarding financial transactions and communications in whatever country their client resides, which will affect data retention and security requirements. RESTful APIs such as Artemis Financial plans to use are vulnerable to data interception if requests and responses are not structured securely. Particularly, the service must use HTTPS for all communication and send confidential information in response headers or in the request or response body. Additionally, a secure authentication scheme should be used.

**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.

* Input validation – Artemis Financial will accept user input from clients. This input must be validated
* APIs – The web service includes a RESTful API, which will need to communicate securely
* Cryptography – PII and Financial data should be encrypted before sending or storing, to protect user privacy.
* Code Error – All effort should be made to eliminate coding errors that could cause insecure code.
* Code Quality – Good quality code is secure code.

**3. Manual Review**

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

* Input in GreetingController did not appear to use validation.
* Did not find working API, program still able to access unsecured data.
* Program accesses data via URL through POST method, which can lead into browser history.
* Program accepts input via URL, which can be exploited.
* No data encryption has been implemented.
* No error handling

**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

|  |  |  |
| --- | --- | --- |
| **Dependency** | **Vulnerability** | **Attribution** |
| Bcprov-jdk15on-1.46.jar | CWE-310: cryptographic issues  CWE-200: information exposure  CWE-347 Improper Verification of Cryptographic Signature  CWE-361 7PK - Time and State  CWE-320 Key Management Errors  CWE-354 Improper Validation of Integrity Check Value  CWE-125 Out-of-bounds Read | CVE-2013-1624  **CVE-2015-6644**  **CVE-2015-7940**  [**CVE-2016-1000338**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000338)  [**CVE-2016-1000339**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000339)  [**CVE-2016-1000341**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000341)  [**CVE-2016-1000342**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000342)  [**CVE-2016-1000343**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000343)  [**CVE-2016-1000344**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000344)  [**CVE-2016-1000344**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000344)  [**CVE-2016-1000346**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000346)  [**CVE-2016-1000352**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-1000352)  [**CVE-2018-5382**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2018-5382)  **CVE-2020-26939**  **CVE-2024-29857** |
| Spring-boot-2.2.4.RELEASE.jar | CWE-668 Exposure of Resource to Wrong Sphere  CWE-400 Uncontrolled Resource Consumption ('Resource Exhaustion') | [**CVE-2022-27772**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-27772)  [**CVE-2023-20883**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-20883) |
| Logback-classic-1.2.3.jar | CWE-502 Deserialization of Untrusted Data | [**CVE-2021-42550**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550) |
| Logack-core-1.2.3.jar | CWE-502 Deserialization of Untrusted Data  Malicious logback configuration files can allow the attacker to execute  arbitrary code using the JaninoEventEvaluator extension.  The attacks involves the modification of DOCTYPE declaration in  XML configuration files. | [**CVE-2021-42550**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-42550)  **CVE-2024-12798**  **CVE-2024-12801** |
| Log4j-api-2.12.1.jar | CWE-295 Improper Certificate Validation  CWE-20 Improper Input Validation  CWE-674 Uncontrolled Recursion | [**CVE-2020-9488**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9488)  [**CVE-2021-44228**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-44228)  [**CVE-2021-44832**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-44832)  [**CVE-2021-45105**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-45105) |
| Snakeyaml-1.25.jar | CWE-776 Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion')  CWE-502 Deserialization of Untrusted Data  CWE-776 Improper Restriction of Recursive Entity References in DTDs ('XML Entity Expansion')  CWE-400 Uncontrolled Resource Consumption ('Resource Exhaustion')  CWE-787 Out-of-bounds Write | [**CVE-2017-18640**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2017-18640)  [**CVE-2022-1471**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-1471)  [**CVE-2022-25857**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-25857)  [**CVE-2022-3064**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-3064)  [**CVE-2022-38749**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-38749) |
| Jackson-databind-2.10-2.jar | CWE-611 Improper Restriction of XML External Entity Reference ('XXE')  CWE-787 Out-of-bounds Write  CWE-770 Allocation of Resources Without Limits or Throttling  CWE-502 Deserialization of Untrusted Data | [**CVE-2020-25649**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-25649)  [**CVE-2020-36518**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-36518)  [**CVE-2021-46877**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-46877)  [**CVE-2022-42003**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2022-42003)  [**CVE-2023-35116**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2023-35116) |
| Tomcat-embed-core-9.0.30.jar | CWE-444 Inconsistent Interpretation of HTTP Requests  CWE-401 Improper Release of Memory Before Removing Last Reference ('Memory Leak'), CWE-476 NULL Pointer Dereference  CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop')  CWE-200 Information Exposure  CWE-444 Inconsistent Interpretation of HTTP Requests ('HTTP Request Smuggling')  CWE-706 Use of Incorrectly-Resolved Name or Reference  CWE-502 Deserialization of Untrusted Data  CWE-200 Information Exposure  CWE-116 Improper Encoding or Escaping of Output | [**CVE-2019-17569**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2019-17569)  [**CVE-2020-11996**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-11996)  [**CVE-2020-13934**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13934)  [**CVE-2020-13935**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-13935)  [**CVE-2020-17527**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-17527)  [**CVE-2020-1935**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-1935)  [**CVE-2020-9484**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2020-9484)  [**CVE-2021-24122**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-24122)  [**CVE-2021-25122**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-25122)  [**CVE-2021-30640**](http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2021-30640) |
| Hibernate-validator-6.0.18.Final.jar | CWE-79 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') | **CVE-2023-1932** |
| Spring-web-5.2.3.RELEASE.jar | CWE-502 Deserialization of Untrusted Data  CWE-601 URL Redirection to Untrusted Site ('Open Redirect')  CWE-400 Uncontrolled Resource Consumption ('Resource Exhaustion') | **CVE-2016-1000027**  **CVE-2020-5421**  **CVE-2024-22262**  **CVE-2024-38809** |
| Spring-beans-5.2.3.RELEASE.jar | CWE-94 Improper Control of Generation of Code ('Code Injection') | **CVE-2022-22965** |
| Spring-webmvc-5.2.3.RELEASE.jar | CWE-22 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal') | **CVE-2021-22060**  **CVE-2024-38816** |
| Spring-context-5.2.3.RELEASE.jar | CWE-178 Improper Handling of Case Sensitivity | **CVE-2022-22968** |
| Spring-expression-5.2.3.RELEASE.jar | CWE-400 Uncontrolled Resource Consumption ('Resource Exhaustion') | **CVE-2023-20861** |

**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.

Artemis Financial must implement changes to their software in order to produce a secure program. They must add input validation to any user-entered data before it is parsed. They must add encryption to the data before it is sent. All communications must go through HTTPS. There are many vulnerabilities due to using outdated dependencies. Recommend updating dependencies to most current versions, and continue to monitor dependencies for updates over time.