# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/22/25** | **Cody Lockerby** |  |

## 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

Cody Lockerby

**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 directive is to aid their customers with financial plans including savings, retirement, investments, and insurance. For this reason, the value of security should be priceless. The information they use and store to help their customers is incredibly important and includes information about the customer such as names, social security numbers and addresses but more importantly the monetary information. Artemis Financial would need income, current savings and investments, and retirement details to consult with their clientele. As there are no details about whether this company only acts domestically, we need to assume their transactions include global customers. I was unable to find any information on government restrictions in the US as it relates to secure communications, other than if a federal agency believes that malicious or illegal activity is ongoing with a foreign entity they can obtain court orders to survey the communication between the two parties. (Foreign Surveillance Act) The largest threat to Artemis is due to what business they conduct and the data they store being so sensitive it will need to be encrypted as well as ensuring secure communication between clients and Artemis Financial. In the world of technology, information is being uncovered, and tactics are evolving faster than most other departments and because of this software developers need to be evolving with the practices. Hackers are constantly discovering new ways to steal information and use it for nefarious means. Open-source libraries are a great tool to use as a collaborative way to learn to better your own program and ensure the security of your customers. An open-source library is a group project but on a global scale.

**2. Areas of Security**

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

A few areas of the vulnerability assessment process flow diagram stick out to me for Artemis Financial.

* Input validation: Customers will be asked to enter information to access accounts and other sensitive information. This application will need to validate the input given to ensure data is not being given unless it specifically should be.
* APIs: APIs allows communication between the server and Artemis Financial. Ensuring communication is secure between these two systems is essential as it can be an area for attack.
* Cryptography: We are acting on the idea that Artemis Financial will be doing business internationally in which case a face-to-face sit down will not always be possible. This stands to reason that communication between the client and Artemis will need to be encrypted to protect the clients from attacks that could occur during the transit of messages.
* Code Error: Ensuring that errors are handled securely will help bolster defense against attacks.
* Code Quality: An aspect that is overlooked is writing quality code. By certifying the written code we can cover our bases and know that we wrote the best program we could.

**3. Manual Review**

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

Further Examination of the code base highlighted some deficiencies:

* I was unable to find any place in the code or the pom.xml file that indicated an input validator.
* The GreetingController.Java is not written to include any input validation.
* I was unable to locate any error handing in the code. There is an exception handler in the DocData.Java but that is the extent.
* The code quality could be improved upon. There are missing notes on some of the files as well as an issue on the DocData.Java that should be corrected.
* Encryption methods need to be introduced. The data being sent from server to client is sensitive and must be treated as such.

**4. Static Testing**

Run a dependency check on Artemis Financials’ 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

|  |  |  |  |
| --- | --- | --- | --- |
| **Dependencies** | **Description** | **Solution** | **Vulnerability ID** |
| bcprov-jdk15on-1.46.jar | The software communicates with a host that provides a certificate, but the software does not properly ensure that the certificate is associated with that host. | Upgrade to newest Version | cpe:2.3:a:bouncycastle:  bouncy-castle-crypto-package:1.46:\*:\*:\*:\*:\*:\*:\* |
| hibernate-validator-6.0.18.Final.jar | A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages | Upgrade to newest Version | cpe:2.3:a:redhat:hibernate  \_validator:6.0.18:\*:\*:\*:\*:\*:\*:\* |
| jackson-databind-2.10.2.jar | A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity. | Upgrade to newest Version | cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\* |
| log4j-api-2.12.1.jar | Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender | Upgrade to newest Version | cpe:2.3:a:apache:  log4j:2.12.1:\*:\*:\*:\*:\*:\*:\* |
| logback-classic-1.2.3.jar | A serialization vulnerability in logback receiver component part of  logback version 1.4.11 allows an attacker to mount a Denial-Of-Service  attack by sending poisoned data. | Upgrade to newest Version | cpe:2.3:a:qos:  logback:1.2.3:\*:\*:\*:\*:\*:\*:\* |
| logback-core-1.2.3.jar | A serialization vulnerability in logback receiver component part of  logback version 1.4.11 allows an attacker to mount a Denial-Of-Service  attack by sending poisoned data. | Upgrade to newest Version | cpe:2.3:a:qos:  logback:1.2.3:\*:\*:\*:\*:\*:\*:\* |
| snakeyaml-1.25.jar | SnakeYaml's Constructor() class does not restrict types which can be instantiated during deserialization | Upgrade to newest Version | cpe:2.3:a:qos:logback  :1.2.3:\*:\*:\*:\*:\*:\*:\* |
| spring-boot-2.2.4.RELEASE.jar | an application that is deployed to Cloud Foundry could be susceptible to a security bypass. | Upgrade to newest Version | cpe:2.3:a:vmware:  spring\_boot:2.2.4:release  :\*:\*:\*:\*:\*:\* |
| spring-boot-starter-web-2.2.4.RELEASE.jar | an application that is deployed to Cloud Foundry could be susceptible to a security bypass. | Upgrade to newest Version | cpe:2.3:a:vmware:spring\_  boot:2.2.4:release  :\*:\*:\*:\*:\*:\* |
| spring-context-5.2.3.RELEASE.jar | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Upgrade to newest Version | cpe:2.3:a:pivotal\_software:  spring\_framework:  5.2.3:release:\*:\*:\*:\*:\*:\* |
| spring-core-5.2.3.RELEASE.jar | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Upgrade to newest Version | cpe:2.3:a:pivotal\_software:  spring\_framework:  5.2.3:release:\*:\*:\*:\*:\*:\* |
| spring-expression-5.2.3.RELEASE.jar | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Upgrade to newest Version | cpe:2.3:a:pivotal\_software:spring\_  framework:5.2.3:  release:\*:\*:\*:\*:\*:\* |
| spring-web-5.2.3.RELEASE.jar | Pivotal Spring Framework through 5.3.16 suffers from a potential remote code execution (RCE) issue if used for Java deserialization of untrusted data | Upgrade to newest Version | cpe:2.3:a:pivotal\_software:spring\_  framework:5.2.3:  release:\*:\*:\*:\*:\*:\* |
| spring-webmvc-5.2.3.RELEASE.jar | A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. | Upgrade to newest Version | cpe:2.3:a:pivotal\_software:spring\_  framework:5.2.3  :release:\*:\*:\*:\*:\*:\* |
| tomcat-embed-core-9.0.30.jar | When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. | Upgrade to newest Version | cpe:2.3:a:apache:  tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* |
| tomcat-embed-websocket-9.0.30.jar | When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. | Upgrade to newest Version | cpe:2.3:a:apache:  tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* |

**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 Financials’ software application.

Based on review of the code base and the dependency check report there are some issues to be addressed. Security issues need to be corrected in the program itself to include adding an input validator, error handling, and encryption. The data being used for this program is sensitive and must be treated as such. I would also like to see the code be annotated with notes, so any developer is able to step in and understand quickly what each part of the code is meant to do. No one programmer will be working on this alone, therefore, to help ourselves notes would be instrumental in keeping the program clean and concise. Next, the pom.xml file needs to be revied and all libraries and dependencies updated to their most current edition. While these vulnerabilities exist the percentage of attacks rises.