# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **11/16/2024** | **Ashlee Wood** |  |

## Client



## Developer

Ashlee Wood

**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?

Secure communications are essential for Artemis Financial. This consulting company provides specialized financial plans for its customers, including savings, retirement, investments, and insurance. It is important to ensure that such sensitive information cannot be attacked.

* Are there any international transactions that the company produces?

Global Rain is software for entrepreneurs, businesses, and government agencies worldwide, so Artemis Financial could produce international transactions through set designs.

* Are there governmental restrictions on secure communications to consider?

Artemis Financial is a consulting company that offers services related to savings, retirement, investments, and insurance information communications. Governmental restrictions could be present in certain situations, and if the transactions involve international customers, there could be restrictions to consider.

* What external threats might be present now and in the immediate future?

External threats that could be present in Artemis Financial are the attack on clients' sensitive information and unauthorized access to modify clients' information.

* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?

Two-factor authentication could be implemented to help stop any fake login attempts and all communication should be done through HTTPS because sensitive information will be going back and forth between the company and its customers.

**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.

1. Input Validation –
2. API – secure interactions relevant to Artemis Financial and its RESTful API service.
3. Code review – important to catch errors in code
4. Cryptography – keeping the confidential information private.
5. Code quality – keeping code clean-cut and efficient.

**3. Manual Review**

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

* I do not see any input validation in CRUDController. Request parameters leave it open to attacks.
* There is no limited input validation length.

This can lead to invalid injection into the code base.

* There is no authentication present for users.
* No error handling that can display useful messages to the user on what issues arose.
* No cryptography present.
* Update the POM.xml
* No authorization to only give access control to certain uses. POLP

**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:

|  |  |  |  |
| --- | --- | --- | --- |
| Dependency: | Vulnerability IDs: | | Solutions |
| bcprov-jdk15on-1.46.jar | cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.46:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:bouncy\_castle\_crypto\_package:1.46:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:bouncy\_castle\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:bouncycastle:the\_bouncy\_castle\_crypto\_package\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\* | | It has a few vulnerabilities, and the solution involves JDK 1.5 to 1.7. |
|  |  |  | |
| hibernate-validator-6.0.18.Final.jar | cpe:2.3:a:redhat:hibernate\_validator:6.0.18:\*:\*:\*:\*:\*:\*:\* | This allows attackers to bypass input sanitation controls. The solution is to upgrade the hibernate-validator to 6.0.20. | |
| jackson-databind-2.10.2.jar | cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:fasterxml:jackson-modules-java8:2.10.2:\*:\*:\*:\*:\*:\*:\* | The solution to these vulnerabilities is to upgrade to the newest current | |
| log4j-api-2.12.1.jar | cpe:2.3:a:apache:log4j:2.12.1:\*:\*:\*:\*:\*:\*:\* | Fixed in Apache Log4j 2.12.3 and 2.13.1 | |
| logback-core-1.2.3.jar | cpe:2.3:a:qos:logback:1.2.3:\*:\*:\*:\*:\*:\*:\* | The solution is to upgrade to the latest version. 1.2.7 beyond | |
| snakeyaml-1.25.jar | cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\* | We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. We recommend upgrading to version 2.0 and beyond. | |
| spring-boot-2.2.4.RELEASE.jar | cpe:2.3:a:vmware:spring\_boot:2.2.4:release:\*:\*:\*:\*:\*:\* | 3.0.x users should upgrade to 3.0.6+. 2.7.x users should upgrade to 2.7.11+. Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+ | |
| spring-boot-starter-web-2.2.4.RELEASE.jar | cpe:2.3:a:vmware:spring\_boot:2.2.4:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:web\_project:web:2.2.4:release:\*:\*:\*:\*:\*:\* | 3.0.x users should upgrade to 3.0.6+. 2.7.x users should upgrade to 2.7.11+. Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+. | |
| spring-core-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\* | In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation. Upgrade the versions. | |
| spring-expression-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\* | In Spring Framework, versions 5.2.x prior to 5.2.15 and versions 5.3.x prior to 5.3.7, a WebFlux application is vulnerable to a privilege escalation. Upgrade versions from prior. | |
| spring-web-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:web\_project:web:5.2.3:release:\*:\*:\*:\*:\*:\* | Upgrade versions | |
| spring-webmvc-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*  cpe:2.3:a:web\_project:web:5.2.3:release:\*:\*:\*:\*:\*:\* | n Spring Framework versions 5.3.0 - 5.3.16 and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition. | |
| tomcat-embed-core-9.0.30.jar | cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* | SP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations. | |
| tomcat-embed-websocket-9.0.30.jar | cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*  cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\* | upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.  NVD-CWE-Other | |

**5. Mitigation Plan**

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

The static testing report has a few steps. Reviewing the report and vulnerabilities will help you determine what to implement next.

Input validation is fundamental to protecting the company from injection attacks. Upgrading the software's versions can resolve most of the vulnerabilities in this testing report. The next step would be implementing the HTTPS protocol to secure online access and communication, which will be useful in preventing attacks. Then, I would implement authentication and remove any hard-coded data. Code review is the next step to ensure it all looks functional and precise. A second pair of eyes is always useful in this. Cryptography Is a good step to keep information private. Lastly, take the time to run the test again with improvements and management.