# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **02/02/25** | **Colt Beavers** |  |

## 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

Colt Beavers

**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? Large amounts of money could potentially be at risk if we do not put enough focus on our security. With that being said, it is of vital importance that we focus on secure communications to mitigate potential leaks of our client’s information to the best of our abilities.
* Are there any international transactions that the company produces? Yes, we are going to have international services available to our clients.
* Are there governmental restrictions on secure communications to consider? There will be several regulations to follow, such as PCI DSS, CCPA and more.
* What external threats might be present now and in the immediate future? Many factors could be potentially threatening now and moving forward, such as poor authentication, SQL injections, and coding errors that could lead to vulnerabilities.
* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies? We need to assess and mitigate risks that come with using a cloud-based platform if we decide to go that route. We also need to be cognizant of the fact that open-source libraries require frequent updates for maintenance.

**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.

Areas of security that stick out to me as being particular areas requiring attention are input validation, cryptography, and APIs. Input validation will be essential as without it, we knowingly put ourselves at risk for very basic attacks that many companies fall victim to. Additionally, we will be working with extremely sensitive data of both our own and our clients. Proper cryptography will help ensure that what must be private, remains that way. Finally, RESTful makes use of several APIs, so ensuring their security will go a long way towards keeping our entire application secure.

**3. Manual Review**

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

We need to implement input validation in the DocData.java file

We need to implement input validation in the CRUD.java file

We need to implement input validation in the CRUDController.java file and implement error handling.

We need to implement input validation in the customer.java file, and ensure that private fields are marked correctly, such as the account balance field.

**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: log4j-api-2.12.1.jar, spring-boot-2.2.4.RELEASE.jar, snakeyaml-1.25.jar, tomcat-embed-core-9.0.30.jar, spring-web-5.2.3.RELEASE.jar, spring-beans-5.2.3.RELEASE.jar, spring-webmvc-5.2.3.RELEASE.jar, bcprov-jdk15on-1.46.jar, logback-classic-1.2.3.jar, logback-core-1.2.3.jar, jackson-databind-2.10.2.jar, spring-expression-5.2.3.RELEASE.jar, hibernate-validator-6.0.18.Final.jar, and spring-context-5.2.3.RELEASE.jar are all listed as vulnerabilities.
* A brief description and recommended solutions provided by the dependency-check report:  
  Focusing on the vulnerabilities that were listed as critical severity first we have log4j-api-2.12.1.jar. This is an improper host mismatch fixed in a later version of Apache. Next, spring-boot-2.2.4.RELEASE.jar is vulnerable to directory hijacking, and can be fixed by upgrading to a newer version. snakeyaml-1.25.jar has a feature that allows entity expansion during a load operation and can be fixed by upgrading to a version past 1.26. Finally, tomcat-embed-core-9.0.30.jar possesses multiple vulnerabilities due to regression. Many versions seemed to be affected, and may be easier to replace with a different package entirely.

**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.

First, we need to begin by addressing all the vulnerabilities shown in the dependency check that have a critical severity level, using the methods outlined above. Then, input validation will need to be implemented in almost every part of the code for the software. Finally, I would work towards proper cryptography and ensuring that fields that need to be marked as private are done so correctly. With these changes implemented, I believe we will take vital steps towards the security of this project and will be ready for another round of review afterwards.