# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **01/26/2025** | **Grace Fletcher** |  |

## 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

Grace Fletcher

**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, a consulting company that develops individualized financial plans for its customers, wants to modernize its operations. Their financial plans include savings, retirement, investments, and insurance. Not only do they want to modernize their operations, they also want to use the latest and most effective software security. Artemis Financial currently uses a RESTful web application programming interface (API).

Secure communications is a crucial part of Artemis Financials’ success, as their databases contain highly classified data on clients such as social security numbers, tax information, contact information, etc. Since there was no information indicating that Artemis Financial is based solely in the United States, it can be a safe assumption that they also deal with international trade/transactions.

Due to the containment of highly classified information, a big security threat is going to be threats directly on the clients, such as phishing attacks. A good action plan will include having regular maintenance on the application/website.

**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.

* Code Quality
  + Good quality code is essential in controlling access to the website. Artemis Financial will have to make sure that methods within their code are only accessible to the appropriate type of user. For example, customers should have access to just their information.
* Validation of User Input
  + Input validation should be required in order for users to access their accounts. This would protect all users’ privacy.
* API
  + The creation and implementation of an API is essential for the application as it is running both internally and externally. This would act as a way to approve or deny data.
* Error Handling
  + The implementation of code error handling will help the development team understand what areas of the application are working and what areas need fixing. This will help the development team keep the application up to date with maintenance.
* Cryptography
  + The implementation of cryptography would make it so that user data would not be compromised in different places geographically. This needs to be taken into consideration as different languages and currencies will be involved in Artemis Financials’ processes.

**3. Manual Review**

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

After reviewing the code while using the Vulnerability Assessment Diagram, I immediately noticed some errors with the POM.XML file. The project variable tags were using “http” instead of “https”, which was preventing the file from running. Looking further into the code, I also noticed that there is a lack of input validation, which is something I would take note of and implement in the future. When taking a look at the API, there seems to be no data validation implemented, which can be seen as a serious threat if not addressed. Further going through the code, I also noticed a lack of cryptography, which is something I would add in in the future. There was also a complete lack of error handling, which can make working on/updating the application as needed difficult. The quality of code was good and up to industry standards, I just would add in more in-line notes throughout the code.

**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
* Apache Tomcat 9.0.30 had multiple vulnerabilities
  + Recommendation: update to the latest version
  + Attributions:
    - CVE-2019-17569
    - CVE-2020-11996
    - CVE-2020-13934
    - CVE-2020-13935
    - CVE-2020-13943
    - CVE-2020-17527
    - CVE-2020-1935
    - CVE-2020-1938
    - CVE-2020-8022
    - CVE-2020-9484
    - CVE-2021-24122
* Bouncy Castle 1.46 had multiple vulnerabilities
  + Recommendation: update to the latest version
  + Attributions:
    - CVE-2013-1624
    - CVE-2015-6644
    - CVE-2015-7940
    - CVE-2016-1000338
    - CVE-2016-1000339
    - CVE-2016-1000341
    - CVE-2016-1000342
    - CVE-2016-1000343
    - CVE-2016-1000344
    - CVE-2016-1000345
    - CVE-2016-1000346
    - CVE-2016-1000352
    - CVE-2017-13098
    - CVE-2018-1000613
    - CVE-2018-5382
* Spring Core 5.2.3 had one vulnerability
  + Recommendation: update to the latest version
  + Attribution:
    - CVE-2020-5421
* SnakeYAML 1.25 had one vulnerability
  + Recommendation: update to the latest version
  + Attribution:
    - CVE-2017-18640

**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.

Based on the results from the manual review and static testing report, I would first recommend that Artemis Financial updates all plug-ins/dependencies to the lates version of that plug-in/dependency (see above). I also recommend that they shift all protocol from HTTP to HTTPS, as this is a more secure route of data encryption. They should also implement some sort of input validation system. Input validation is crucial to their success as a banking system since they deal with highly confidential user information on a daily basis. Input validation would act as one of the first barriers of defense against their information being stolen. Going along this same method of thinking, Artemis Financial should also implement cryptography into their code design, especially since they will be dealing with different languages/types of currencies. Finally, Artemis Financial should add in error handling to their design. This will make it easier for them to update/maintain their application as needed.