# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **09/19/2024** | **Cory Graham** |  |

## 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

Cory Graham

**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 company that will have a broad range of clients from around the world. This being a financial firm, the need to protect and secure all transactions foreign and domestic, as well as securing all accounts from hacks or data breaches is paramount. No current government restrictions are in place to restrict secure communications, but keeping up to date with current legislation and potential issues will help secure communications in the future. Artemis Financial should strive to prevent hacking and data attacks by implementing several factors of authentication, limit level of authorization for employees to what they need, and constantly update software to protect their system.

**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: Artimis Financial will want to prevent SQL injection attacks as well as validating who is accessing their system. Putting in a validator will offer protection from these attacks.
* APIs: APIs expose endpoints that can be used by external parties to interact with the application. To prevent this, we could implement a RESTful API
* Cryptography: Artimis Financial should be wary of middlemen in the communications between employees or clients and the servers. Securing and finding existing middlemen will ensure data security.
* Client/Server: Errors in security features can allow for unauthorized access. Checking for errors and handling them will secure the financial system.
* Code Error: Errors in security features can allow for unauthorized access. Checking for errors and handling them will secure the system.
* Code Quality: High quality code will ensure maximal discovery of errors and bugs and minimal vulnerabilities.

**3. Manual Review**

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

* Greeting.java contains no input validator.
* API has not been implemented. Only access to the system is through the code itself.
* No evidence of cryptography found.
* Client/Server has no protections.
* Code Error had an issue on DocData.java. This issue is on line 26, it says the variable con was not used.

**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** | **Description** | **Evidence Count** |
| --- | --- | --- |
| [**bcprov-jdk15on-1.46.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l1_991c96a4e31e6c19e2b9136c8955bd423f2dc4c7) | **The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7.** | **38** |
| [jackson-databind-2.10.2.jar](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l5_0528de95f198afafbcfb0c09d2e43b6e0ea663ec) | General data-binding functionality for Jackson: works on core streaming API | **39** |
| [**log4j-api-2.12.1.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l10_a55e6d987f50a515c9260b0451b4fa217dc539cb) | The Apache Log4j API | **42** |
| [**logback-core-1.2.3.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l12_864344400c3d4d92dfeb0a305dc87d953677c03c) | [logback-core module](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aqos&cpe_product=cpe%3A%2F%3Aqos%3Alogback&cpe_version=cpe%3A%2F%3Aqos%3Alogback%3A1.2.3" \t "_blank) | **31** |
| [**spring-boot-2.2.4.RELEASE.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l15_225a4fd31156c254e3bb92adb42ee8c6de812714) | Spring Boot | **39** |
| [**spring-core-5.2.3.RELEASE.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l17_3734223040040e8c3fecd5faa3ae8a1ed6da146b) | Spring Core | **36** |
| [**spring-expression-5.2.3.RELEASE.jar**](file:///C:\Users\Cory%20Graham\Downloads\rest-service\target\dependency-check-report.html#l18_d0c6bb10758805b2153c589686b8045554bfac2d) | Spring Expression Language (SpEL) | **36** |

**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.

Looking at the current state of the system, there is a lot that can be handled updated as the development continues. First, make sure all add on systems are updated as to secure that access. Second, implement a validation system. Make sure only authorized people have access. Once those are in place other areas can then be improved on and other areas of concern can be found and addressed.