### Southern New Hampshire University

### Project One

Elliot Collins

### Dr. Lyon

### CS-305 – Software Security

20 March 2022

****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

[Document Revision History 4](#_Toc32574607)

[Client 4](#_Toc32574608)

[Instructions 4](#_Toc32574609)

[Developer 5](#_Toc32574610)

[1. Interpreting Client Needs 5](#_Toc32574611)

[2. Areas of Security 6](#_Toc32574612)

[3. Manual Review 7](#_Toc32574613)

[4. Static Testing 10](#_Toc32574614)

[5. Mitigation Plan 11](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **19 March 2022** | **Elliot Collins** | **Completed Interpreting Client Needs** |
| **1.1** | **20 March 2022** | **Elliot Collins** | **Completed Areas of Security and Manual Review**  **Began Static Review** |
| **2.0** | **22 March 2022** | **Elliot Collins** | **Completed Static Review**  **Completed Mitigation Plan** |

## Client



## Instructions

Deliver this completed vulnerability assessment report, identifying your findings of security vulnerabilities and articulating recommendations for next steps to remedy the issues you have found.

Respond to the five steps outlined below and include your findings. Replace the bracketed text on all pages with your own words. If you choose to include images or supporting materials, be sure to insert them throughout.

## Developer

Elliot Collins

## 1. Interpreting Client Needs

Determine your client’s needs and potential threats and attacks associated with their application and software security requirements. Consider the following 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 about secure communications to consider?
* What external threats might be present now and in the immediate future?
* What are the “modernization” requirements that must be considered, such as the role of open source libraries and evolving web application technologies?

The value of secure communications at Artemis Financial lies in the great amount of sensitive information that is going to be passed between the company, the website, and the client. The company handles people’s finances so they are going to need their customers to disclose very personal information, that if leaked, could cause devastating effects on people’s finances and lives. This also opens up the possibility for the company to be sued if people’s financial information is not properly secured.

It isn’t specified that Artmeis Financial will be conducting international transactions, however there are government restrictions to be implemented. The Gramm-Leach-Bliley act requires that financial institutions disclose to their customers how they will be sharing their information.

A company that handles people’s money and holds their financial information is going to be one of the biggest targets for online attacks. The company needs to ensure they are constantly keeping up to date with security technologies as attackers are going to try and breach their information every way possible.

It’s definitely important to have a security professional monitoring the company’s security as the world of cybersecurity is ever evolving. The use of open source libraries allows for more skilled individuals to evaluate security threats and provide useful feedback.

## 2. Areas of Security

Referring to the Vulnerability Assessment Process Flow Diagram, identify which areas of security are applicable to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

* Input Validation
  + The RESTful API will need to validate user input to allow them to access the system.
* APIs
  + The system uses RESTful API
* Client/Server
  + Clients will be interacting with the server.
* Code Error
  + The System should print error messages when errors in the code occur.
* Code Quality
  + Whenever handling sensitive information, best and most secure coding practices need to be implemented.
* Encapsulation
  + Customer information should be contained in secure objects.

## 3. Manual Review

Continue working through the Vulnerability Assessment Process Flow Diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

* Input Validation
  + The System uses the DocData class to validate user inputs. The read\_document method on line 21 will read files. There are currently no functions in place to validate account number.

Text

Description automatically generated

* APIs
  + The system uses the RestServiceApplication class to run the RESTful API service.

Text

Description automatically generated

* Client/Server
  + In the CRUDController class, the system uses a CRUD system on line 13 to request information from the database in the DocData class to be presented to the user.

Text

Description automatically generated

Text

Description automatically generated

* Encapsulation
  + The system uses an ID system to identify a user and return their information. The customer class contains the user’s account number and balance. The Greeting class contains the object for the user’s ID, which is then passed to the DocData class to return the user’s information from a document.

A screenshot of a computer

Description automatically generated with medium confidence

Text

Description automatically generated

## 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 dependency check report. Include the following:

1. The names or vulnerability codes of the known vulnerabilities
2. A brief description and recommended solutions provided by the dependency check report
3. Attribution (if any) that documents how this vulnerability has been identified or documented previously

* [cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Abouncycastle&cpe_product=cpe%3A%2F%3Abouncycastle%3Alegion-of-the-bouncy-castle-java-crytography-api&cpe_version=cpe%3A%2F%3Abouncycastle%3Alegion-of-the-bouncy-castle-java-crytography-api%3A1.46)
  + This vulnerability allows for injection attacks to bypass input validation.
  + This issue has since been patched by Bouncycastle
  + <https://nvd.nist.gov/vuln/detail/CVE-2016-1000338>
  + <https://github.com/bcgit/bc-java/commit/9385b0ebd277724b167fe1d1456e3c112112be1f>
* [cpe:2.3:a:qos:logback:1.2.3:\*:\*:\*:\*:\*:\*:\*](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)
  + This vulnerability allowed for attackers to use their access privileges to edit configurations that would allow for execution of malicious code. This issue was solved by Logback.
  + <https://jira.qos.ch/browse/LOGBACK-1591>
* [cpe:2.3:a:apache:log4j:2.12.1:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache&cpe_product=cpe%3A%2F%3Aapache%3Alog4j&cpe_version=cpe%3A%2F%3Aapache%3Alog4j%3A2.12.1)
  + This vulnerability allowed for remote code execution and input of malicious data by using message lookup patterns. This issue was solved by Openwall by removing the functionality of message lookup patterns.
  + <https://www.openwall.com/lists/oss-security/2021/12/14/4>
* [cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Asnakeyaml_project&cpe_product=cpe%3A%2F%3Asnakeyaml_project%3Asnakeyaml&cpe_version=cpe%3A%2F%3Asnakeyaml_project%3Asnakeyaml%3A1.25)
  + This vulnerability allowed for entity expansion during load operations.
  + The issue has since been patched by Apache.
  + [https://lists.apache.org/thread/obr7cqy8lk0dk4gp018pplkqnqsxpkff](%20https:/lists.apache.org/thread/obr7cqy8lk0dk4gp018pplkqnqsxpkff)
* [cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Afasterxml&cpe_product=cpe%3A%2F%3Afasterxml%3Ajackson-databind&cpe_version=cpe%3A%2F%3Afasterxml%3Ajackson-databind%3A2.10.2)
  + This issue caused an error when trying to use a large amount of nested objects in an array list or a JSON file.
  + The developers at FasterXML are currently patching this issue.
  + <https://github.com/FasterXML/jackson-databind/issues/2816>
* [cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Aapache&cpe_product=cpe%3A%2F%3Aapache%3Atomcat&cpe_version=cpe%3A%2F%3Aapache%3Atomcat%3A9.0.30)
  + This issue in Apache Tomcat caused a memory leak when performing Websocket connections and caused an error.
  + The issue has been patched by Apache.
  + <https://lists.apache.org/thread.html/r8097a2d1550aa78e585fc77e602b9046e6d4099d8d132497c5387784@%3Ccommits.myfaces.apache.org%3E>
* [cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Apivotal_software&cpe_product=cpe%3A%2F%3Apivotal_software%3Aspring_framework&cpe_version=cpe%3A%2F%3Apivotal_software%3Aspring_framework%3A5.2.3)
  + This is an issue with the Spring framework that didn’t properly protect against RFD attacks.
  + This issue only exists in older versions so if the Spring framework is up to date there shouldn’t be a vulnerability.
  + <https://lists.apache.org/thread.html/r1c679c43fa4f7846d748a937955c7921436d1b315445978254442163@%3Ccommits.ambari.apache.org%3E>
* [cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*](https://nvd.nist.gov/vuln/search/results?form_type=Advanced&results_type=overview&search_type=all&cpe_vendor=cpe%3A%2F%3Apivotal_software&cpe_product=cpe%3A%2F%3Apivotal_software%3Aspring_framework&cpe_version=cpe%3A%2F%3Apivotal_software%3Aspring_framework%3A5.2.3)
  + This is a repeat of the previous Spring framework issue that has since been patched.

## 5. Mitigation Plan

After interpreting your results from the manual review and static testing, identify the steps to remedy the identified security vulnerabilities for Artemis Financial’s software application.

* Input Validation
  + There is currently no functionality it in place to validate the account number of each customer. By implementing validation parameters, such as length and character type parameters, you can add an initial level of validation.
  + By taking the entered account number and parsing it through a database of valid account numbers, you can ensure that the number is valid and exists in the system.
* API
  + The system uses a REST API. It’s important to keep this system secure, on of the best ways to ensure security is by implementing a permissions system. By ensuring low-level users can’t interact with the API, we can prevent injection attacks and other such malicious behavior. It also helps to make the REST API method password protected.
  + (Sycrido et al., 2021)
* Client/Server
  + The system has a method in place that pulls information from a database. We must ensure that malicious users can not break the system with integral overflow, making too many requests. There should be parameters in place that trigger a denial of service, preventing use when too many requests are made.
* Encapsulation
  + When data is pulled from the database, it is placed into objects so they information can be used elsewhere in the system.
  + Data needs to be protected when placed into objects so that malicious users cannot use it in the wrong areas of the system. The ID object is made private so that it can only be used in the proper areas of the system.
  + (Ram et al., 1966)

**References**

Ram, RamRam 3155 bronze badges, Cheng ChenCheng Chen 40.8k1616 gold badges106106 silver badges166166 bronze badges, & GauravsaGauravsa 5. (1966, December 1). *Encapsulation and data security*. Stack Overflow. Retrieved March 22, 2022, from https://stackoverflow.com/questions/54319187/encapsulation-and-data-security#:~:text=%22Encapsulation%2C%20refers%20to%20an%20object's,overall%20functionality%20of%20the%20system.

Sycrido, Admin, Buzz, C., Alzate, M., Prajapat, B., Varun, Ezzabuzais, Sejal, Rameez, Ibeh, D., K, C., Manoj, Bowen, W., Ja, R., Joe, Amar, Eric, Ignacio, Vikki, … Maes, T. (2021, September 30). *Rest API security essentials*. REST API Tutorial. Retrieved March 22, 2022, from https://restfulapi.net/security-essentials/

*What is an open source library? definition and faqs*. What is an Open Source Library? Definition and FAQs | HEAVY.AI. (n.d.). Retrieved March 22, 2022, from https://www.heavy.ai/technical-glossary/open-source-library

*Secure Coding Guidelines for Java SE*. Secure coding guidelines for java SE. (n.d.). Retrieved March 22, 2022, from https://www.oracle.com/java/technologies/javase/seccodeguide.html