****

# Artemis Financial Vulnerability Assessment Report

# Southern New Hampshire University

Alyssa Mazzucotelli

### Joseph Conlan

### CS-305-T4196 – Software Security 22EW1

3/19/2023

**Table of Contents**

[Document Revision History 3](#_heading=h.gjdgxs)

[Client 3](#_heading=h.30j0zll)

[Instructions 3](#_heading=h.1fob9te)

[Developer 4](#_heading=h.3znysh7)

[1. Interpreting Client Needs 4](#_heading=h.2et92p0)

[2. Areas of Security 4](#_heading=h.tyjcwt)

[3. Manual Review 4](#_heading=h.3dy6vkm)

[4. Static Testing 4](#_heading=h.1t3h5sf)

[5. Mitigation Plan 4](#_heading=h.4d34og8)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **3/19/2023** | **Alyssa Mazzucotelli** | **Initial creation** |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In the report, identify your findings of security vulnerabilities and provide recommendations for 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 choose to include images or supporting materials. If you include them, make certain to insert them in all 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

Alyssa Mazzucotelli

## Interpreting Client Needs

Our client Artemis Financial is a consulting company that develops individualized financial plans for their customers which include savings, retirement, investments, and insurance. Secure communications are extremely important to Artemis Financial as outside attackers could be interested in stealing money as well as personal client information. It does not appear that Artemis Financial currently makes international transactions as it seems they only deposit in one currency type. Artemis Financial is not a government agency so I do not believe any US governmental restrictions apply to them besides possibly the Children’s online Privacy Protection Act (COPPA) in regards to the collection of information on children under the age of 13 if there is no age requirement for account creation. Some threats that might be present now and in the immediate future include the possibility of attackers interested in any way they can access either money, personal information, or a combination of the two so both must be secured. Some modernization requirements to consider would be that as time goes on new vulnerabilities can be found within the dependencies of a project so it is important to update to newer dependencies where those problems are fixed as well as using secure coding practices while creating the code itself.

## Areas of Security

While referring to the Vulnerability Assessment Process Flow diagram I believe the following are important areas of security that are applicable to Artemis Financial’s software application.

* Input Validation: There are user inputs that will need to be secure and validated.
* APIs: The application uses spring framework APIs so these interactions need to be secure.
* Cryptography: There is user information that should be protected while traveling to insure its authenticity, integrity, and confidentiality.
* Code Error: Errors from the previous areas need to be handled securely.
* Code Quality: It is always important to use secure coding practices and patterns to ensure the security of an application.

## Manual Review

* customer.java
  + line 5: account\_balance is a public int
* DocData.java
  + line 27: hard-coded database connection where username and password are passed in plain text.
* General
  + No Input Validation
  + No authentication or encryption

## Static Testing

Name: bcprov-jdk15on-1.46.jar, Bouncy Castle Crypto package

Description: Bouncy Castle versions before 1.61 have cryptographic and information exposure issues as well as improper validation checks so it is recommended to update to a version newer than 1.61.

Attribution:

* CVE-2013-1624
* CVE-2015-6644
* 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-2018-5382
* CVE-2020-0187
* CVE-2020-26939

Name: spring-boot2.2.4.RELEASE.jar, Spring Boot

Description: Spring Framework before 3.2.4 and 4.0.0M1 contains a handful of issues that could allow different exploits like code injection and cross-site request forgery. It is recommended to update to at least 5.3.20+ or 5.3.22+.

Attribution:

* CVE-2013-4152
* CVE-2013-7315
* CVE 2014-0054
* CVE-2016-1000027
* CVE-2018-11039
* CVE-2018-11040
* CVE-2018-1257
* CVE-2020-5421
* CVE-2022-22950
* CVE-2022-22965
* CVE-2022-22968
* CVE-2022-22970
* CVE-2022-27772

Name: logback-core-1.2.3.jar, logback-core module

Description: logback versions before 1.2.7 can allow an attacker to edit configuration files that could allow arbitrary code to be loaded from LDAP servers. Update to newer than 1.2.7.

Attribution: CVE-2021-42550

Name: Log4-api-2.12.1.jar Apache Log4j API

Description: Apache Log4j API has improper validation vulnerabilities which were fixed in 2.12.3 and 2.13.1.

Attributions:

* CVE-2020-9488
* CVE-2021-44228
* CVE-2021-44832
* CVE-2021-45046
* CVE-2021-45105

Name: snakeyaml-1.25.jar, YAML 1.1 parser and emitter for Java

Description: Snake YAML contains several vulnerabilities before 1.31 so it is recommended to update to a newer version.

Attributions:

* CVE-2017-18640
* CVE-2022-25857
* CVE-2022-38749
* CVE-2022-38750
* CVE-2022-38751
* CVE-2022-38752

Name: jackson-databind-2.10.2.jar

Description: A stack overflow and improper restriction of XML External Entity Reference prior to 2.13.0. Update to 2.13.0 or newer

Attributions:

* CVE-2020-25649
* CVE-2020-36518

Name: tomcat-embed-core-9.0.30.jar, Core Tomcat implementation

Description: Apache Tomcat contains several issues including memory leak and information exposure prior to 10.1.0-M14 so it is recommended to update to a newer 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
* CVE-2021-25122
* CVE-2021-25329
* CVE-2021-30640
* CVE-2021-33037
* CVE-2021-41079
* CVE-2022-29885
* CVE-2022-34305

Name: spring-boot-starter-validation-2.2.4.RELEASE.jar

Description: spring-boot versions prior to v2.2.11.RELEASE were vulnerable to temporary directory hijacking. Recommended to update to v2.2.11.RELEASE.

Attribution: CVE-2022-27772

Name: hibernate-validator-6.0.18.Final.jar

Description: Hibernate’s Bean Validation had an improper input validation error in version 6.1.2.Final

Attributions: CVE-2020-10693

Name: spring-core-5.2.3.RELEASE.jar

Description: Pivotal Spring Framework through 5.3.16 contained several vulnerabilities. Update to 5.3.20+ or higher.

Attributions:

* CVE-2016-1000027
* CVE-2020-5421
* CVE-2021-22060
* CVE-2021-22096
* CVE-2021-22118
* CVE-2022-22950
* CVE-2022-22965
* CVE-2022-22968
* CVE-2022-22970
* CVE-2022-22971

## Mitigation Plan

After both the manual review and the static testing I believe there are a few ways that Artemis Financial could mitigate some of the identified security vulnerabilities. Many security problems could be avoided by updating the dependencies for the project to newer released versions as several have patched the previously found issues from older versions. The other recommendation I have is to implement some form of input validation to avoid outside files or code from being injected in.