****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

Table of Contents

[Document Revision History 3](#_Toc32574607)

[Client 3](#_Toc32574608)

[Instructions 3](#_Toc32574609)

[Developer 4](#_Toc32574610)

[1. Interpreting Client Needs 4](#_Toc32574611)

[2. Areas of Security 4](#_Toc32574612)

[3. Manual Review 5](#_Toc32574613)

[4. Static Testing 5](#_Toc32574614)

[5. Mitigation Plan 5](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **9/12/2021** | **James Porter** | **Initially Completed** |
| **1.1** | **9/19/2021** | **James Porter** | **Reviewed Content** |

## 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

James Porter

## 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?

In this project we are dealing with Artemis Financial which is a company that helps individuals plan using various savings retirement and investment options. They also just modernized and have asked that we look for potential vulnerabilities that have come with this new development. Because of the above information, it is essential that the information dealt with is properly protected from malicious users. This company does produce some international transactions and will be required to follow some government requirements and reporting.

In the history of man, greed has been a very prominent struggle and thus we get malicious users. Anyone trying to make an extra dollar could gain access to this software and steal information and money from users. These users can be either abroad or in country and have access to vast amounts of resources that must be considered. It should be considered that this software will need frequent updates as new threats are discovered and patches for these threats are added. It is also important to consider that open-source libraries also are available to anyone and should be used with caution.

## 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.

There are a lot of areas of security that should be considered when looking at the Vulnerability Assessment Process Flow Diagram. First, we must consider the importance of input validation. It is important that the input from the user is correct and there are no vulnerabilities in reading input from the user. There are a lot of attacks that begin at the login screen. Next, we will discuss both the APIs and the Client/Server interactions. In general, APIs focus on the interconnectivity between computers. This must be secured properly so that someone can’t gain access to another person’s account using connectivity between computers. Similarly, it is important to consider the connection between the server and the client. This connection could allow malicious users access to the server and limitless information regarding investment information. Encryption is essential when consider personal information when it comes to financial records. When data is encrypted, it decreases the possibility that information that is intercepted can be read properly. It is also important to code with quality and encapsulation. When used properly these two security features can benefit a system by ensuring that data is properly dealt with.

This may seem like a lot of security requirements, but it is always important to secure data as important as financial information. If this information was ever leaked it could lead to the devastation or destruction of an individual’s life savings or livelihood.

## 3. Manual Review

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

In the myDateTime.java file there are some security concerns. First and foremost, all the variables in the class do not have any privacy details set. This would allow a malicious user to potentially get access and change the date and time in their own favor. Additionally, none of the functions have privacy details set as to where they can be accessed. Similarly, to the date and time, this could allow malicious user control to some portion of the system that they shouldn’t have.

Another security concern in the system is in the customer.java file. This flaw is like the time and date issue but could cause bigger issues. In this file the account\_balance number does not have a declared privacy. A malicious user could gain access to this system and change their account balance to be an insane balance that they do not actually have.

Lastly, in the DocData.java file there is a potentially dangerous section of code. In this section a catch block is auto generated and could allow the user access to a section of code that isn’t run properly.

## 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
4. CVE-2021-33037 (tomcat-embed-websocket-9.0.30.jar & tomcat-embed-core-9.0.30.jar) – Using the Apache software leads to the possibility users to request and get more information than they should be allowed to have. Oracle has released some patches to the problem and recommends keeping up to date with software that is actively supported.
5. CVE-2021-22118 (spring-core-5.2.3.RELEASE.jar & spring-aop-5.2.3.RELEASE.jar) – In older versions of the spring configuration, a malicious user can artificially increase their privilege. Once they have higher access privilege the user can then access or modify data that shouldn’t be seen by the user. No specific updates have been released, however updating the spring configuration can better protect the system from malicious users.
6. CVE-2017-18604 (snakeyaml-1.25.jar) – This flaw leaves the system vulnerable to an entity expansion attack. This attack allows malicious users the ability to work around and change the shortcuts or strings so functions do more than they should. There are some patch releases that limit the damage that a malicious user can inflict on the system.
7. CVE-2020-9488 (log4j-api-2.12.1.jar) – This issue with Apache allows a malicious user to initiate a man-in-the-middle attack. This attack allows the malicious user to intercept data that they shouldn’t see. Like previous issues, updating to the newer Apache versions that better protect from this issue.
8. CVE-2020-25649 (jackson-databind-2.10.2.jar) – This issue is like the snakey flaw (III above) because the user gains access and can complete an entity expansion attack. The primary threat from this flaw is data integrity. Like all the above issues it is beneficial to keep the software associated with this flaw up to date.
9. CVE-2020-10693 (hibernate-validator-6.0.18.Final.jar) – A bug in the Hibernate Validator allows an invalid expression to be read as valid. This allows the malicious user to bypass the typical input sanitation that the system tries to complete. There is currently no plan for the patch of this flaw as of now because the software involved is only in the technical preview phase. To mitigate this threat, it is important to limit the amount of the system that is reliant on this software.
10. CVE-2018-1000613 (bcprov-jdk15on-1.46.jar) – This flaw allows private keys to be accessed and results in unexpected code to be executed. In situations like these, the malicious user can gain access to private keys that allow access to the system and include references to unexpected code. An update has been issued to patch this flaw.

## 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.

The first step of the mitigation plan should be going through the areas of security section and familiarizing everyone with the important security features of this software. Make sure that everyone who will be developing this system knows how secure this system needs to be and that they are familiar with the initial concerns. Next it is important to begin addressing the privacy issues that are discussed in the manual review section of this document. Additionally, the catch block discussed in the manual review section should be reviewed to ensure that errors are handled properly.

After all the above are dealt with we can start going through our dependency tests to ensure that potential issues are dealt with properly. First, make sure that all to appropriate software used is up to date and includes any patches that the software developers have released. It is also important to limit the number of dependencies that don’t have patches of fixes for bugs and flaws identified such as in flaw VI (above).