****

# 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 4](#_Toc32574613)

[4. Static Testing 4](#_Toc32574614)

[5. Mitigation Plan 4](#_Toc32574615)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **7-20-2022** | **Taylor Murphy** |  |

## 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

Taylor Murphy

## 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 in the client’s case since they are a financial company is paramount. The security of not only the company’s financials but the financials of their clients need to be protected. Securing the communications between the company and its clients is very important with how quickly hackers and third parties can use this information and what little information they need in today’s world to carry out their plans. It is not stated in the scenario whether they are involved with international transactions but if they were communication protocols with the international interties need to be another important aspect to the company’s security along with being able to have communication with these foreign entities as well. There’s seems to be a federal shift towards financial companies having better cyber security systems but there doesn’t seem to be any federal restrictions on the level of security. There are some restrictions on how the companies go about securing certain types of communications with clients and the information transferred to be considered but nothing that should restrict what the program can do. External threats now and in the immediate future are vulnerabilities in the code that can be exploited such as someone being able to intercept communications between the host and client with out being noticed and the information still make it to the client but has also been received by a third party. There also an increasing number of DDOS attacks in recent times where a sever can be bombarded with multiple requests that it becomes vulnerable as it processes these junk request for someone to come in a back door. The modernization requirements to consider are keeping within the restrictions of the REST system so that communications with other REST APIs can be secured which by using some open source libraries that makes this feat easier and much more standardized can also ensure that we could have a changing and evolving system which means the best for security since changing makes it harder for third parties to learn the system.

## 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.

Going through the vulnerability assessment diagram starting with architecture everything seems to be good here. Input validation needs some work in my opinion with the simple greeting and greetingcontroller files with a simple get Id and get content I believe this can be easily exploited by third parties maybe the inclusion of another additional check that requires something outside the program could be a good for security. The API interactions seem secure from what I can see but in my opinion there needs to be more encryption of user financial information such as the account-number and account-balance variables in the customer.java file. In the DocData.java file the connection between the server and clients seems to be secure as well as code errors seem to be in check from what I’ve seen the code quality though does seem kind of basic for the importance of the information that is being handled and processed by the system needs to be more robust or include more intricate interactions between files and classes to encrypt the user information and the code base from third parties.

## 3. Manual Review

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

There are some vulnerabilities within the greeting.java and greeting.java files where there is a simple get id and get content items that could exploited by third parties. Encryption of the some of the items in the customer.java and DocData.java files could help to better secure the user information from third parties such as the account-number and account-balance items int the customer.java file that pull this important information. There are some vulnerabilities I could identify in the overall code base in how simplistic some of it is when the level of importance the information that is being processed is.

## 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

Graphical user interface, text, application, email

Description automatically generated

Above is the dependency check I ran on the code base that shows some of vulnerabilities and dependencies that the code base suffers from. The bcprov-jdk15on-1.46.jar is of an older version of the JCE provider that does not validate ASN.1 encoding of verification this can lead to the injection of extra data into the sequence and still allows validation which can lead to third party access into the program. A flaw was discovered in the Hibernate Validator where there is a bug in the message processor that enables attackers to bypass input sanitation in the user-controlled data in error messages. The Jackson Databind has a flaw where it allows a vulnerability to XXE attacks which can damage data integrity. There are some improper validation certifications with hos mismatch in the Apache Log4j SMTP appender which can lead to interception of messages between the host SMTP server and the client this has been fixed in the 2.12.3 version. The logback-core module in the 1.2.7 version allows an attacker with required privileges to configure files and execute arbitrary code from servers. The YAML 1.1 parser has an alias feature in this version that allows entity expansion during the load operation but has been fixed with anything after 1.26. The most critical vulnerabilities are the tomcat-embed core and WebSocket files which can lead to the possibility of HTTP request smuggling which was seen as unlikely and the spring web, boot and core RELEASE files after some review seem to be some false positives as the vulnerabilities only affect versions no longer supported by the maintainer.

## 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 Financials’ software application.

Some steps I would include to better secure the code base would as stated earlier the inclusion of more robust and intricate code along with some encryption in user information items. Securing the input validation in the greeting files to include more complex entry and maybe the addition of another validation code that requires something outside the database much like two-factor authentication for the best security when the user is signing in. When looking back at the static test and dependency check there seems to be a lot of vulnerabilities that can be remedied by updating the version of the included files to newer ones where there where identified problems in the current versions that allowed third party access or intervention. There seemed to be some false positives in the spring web, boot and core RELEASE files that when looking into further do not require any intervention since the vulnerabilities only affects versions no longer supported by the maintainer.