****

# 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** | **03/19/2022** | **Andrei Kourouchin** | **Created initial vulnerability assessment report** |

## 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

Andrei Kourouchin

## 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?

As Artemis Financial is a financial consulting company, secure communications are number one priority. Regardless of whether it is to protect transactions, their client’s data or reporting to one or more regulatory or government agencies. Government restrictions on communications to consider could be found in the "FFIEC Guidance on Electronic Financial Services and Consumer Compliance” which details information on how electronic financial services should be provided securely. There are multiple external threats that are present now, and will be in the immediate future, such as government agencies spying on foreign companies, lone actors looking to make money, as well as simply people who want to deface or hijack the website for their own nefarious purposes. Depending on the business model of Artemis Financial and their record with treating their clients, there is also the possibility of hacktivism coming into play against them. As there are open-source libraries that will be used, as well as are used, making sure that keeping all libraries up to date to ensure the security trifecta is secure is one thing to consider. The open-source nature of the software means that attackers will have a grey box view of the system where they can look for vulnerabilities in the source code.

## 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.

* As this application will be taking user input, we need to ensure **input validation** is correctly implemented
* **APIs** will be used to transfer data between client and server, and as such, we need to ensure secure API interactions are used
* As users can make mistakes, we need to ensure **error handling** works correctly to ensure that the application does not do anything unexpected.
* **Code Quality** - We need to ensure we use secure coding practices to ensure that no unexpected user information is leaked, or able to be edited.

## 3. Manual Review

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

* The database path in DocData.java is simply localhost/DBNAME which should be obscured to ensure enumeration techniques do not work.
* Hard coded passwords exist in the DocData.java file, hard coded passwords are also root/root which is default for Linux machines.
* There is no input validation in the deposit function in the customer.java class
* Request parameters are not validated
* Business names are sent as request parameters in the CRUDController class.

## 4. Static Testing

Run a dependency check on Artemis Financials’ 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

jackson-databind

* **CRITICAL**
* <https://nvd.nist.gov/vuln/detail/CVE-2020-36518>
* jackson-databind before 2.13.0 allows a Java StackOverflow exception and denial of service via a large depth of nested objects.
* Upgrade will fix the issue

Apache Tomcat 9.0.30

* **CRITICAL** Upgrade to fix the issue
* Multiple vulnerabilities that impact availability
* <https://nvd.nist.gov/vuln/detail/CVE-2021-42340>

spring-core-5.2.3

* <https://nvd.nist.gov/vuln/detail/CVE-2020-5421>
* The protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used through the use of a jsessionid path parameter.
* Upgrade to fix the issue

logback-core-1.2.3

* <https://nvd.nist.gov/vuln/detail/CVE-2021-42550>
* In logback version 1.2.7 and prior versions, an attacker with the required privileges to edit configurations files could craft a malicious configuration allowing to execute arbitrary code loaded from LDAP servers.
* Upgrade to most recent version to solve current vulnerability.

log4j-api-2.12.1

* <https://nvd.nist.gov/vuln/detail/CVE-2021-44832>
* Apache Log4j2 versions 2.0-beta7 through 2.17.0 (excluding security fix releases 2.3.2 and 2.12.4) are vulnerable to a remote code execution (RCE) attack when a configuration uses a JDBC Appender with a JNDI LDAP data source URI when an attacker has control of the target LDAP

server. This issue is fixed by limiting JNDI data source names to the java protocol in Log4j2 versions 2.17.1, 2.12.4, and 2.3.2.

* Upgrading to the most recent version will solve this vulnerability

SnakeYAML 1.25

* The Alias feature in SnakeYAML 1.25 allows entity expansion during a load operation
* <https://nvd.nist.gov/vuln/detail/CVE-2017-18640>
* Solution is to patch the library

Bouncy Castle Crypto package

* ECIES implementation allowed the use of ECB mode. This mode is regarded as unsafe and support for it has been removed from the provider.
* Solution is to patch the library
* <https://security.netapp.com/advisory/ntap-20181127-0004/>

hibernate-validator-6.0.18

* <https://nvd.nist.gov/vuln/detail/CVE-2020-10693>
* A flaw was found in Hibernate Validator version 6.1.2.Final. A bug in the message interpolation processor enables invalid EL expressions to be evaluated as if they were valid. This flaw allows attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages.
* Upgrading to the most recent version will fix the issue.

## 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.

We need to do a few items to get this software up to a state that we can use it:

* Switch to HTTPS for all communications
* Update all vulnerable libraries as detailed above
* Remove hard coded database credentials
* Implement secure authentication
* Move request parameters to body rather than using URI