****

# CS 305 Project One

**Artemis Financial Vulnerability Assessment Report**

**Table of Contents**

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

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

[Instructions 3](#_heading=h.3znysh7)

[Developer 4](#_heading=h.2et92p0)

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

[2. Areas of Security 4](#_heading=h.3dy6vkm)

[3. Manual Review 4](#_heading=h.1t3h5sf)

[4. Static Testing 4](#_heading=h.4d34og8)

[5. Mitigation Plan 4](#_heading=h.2s8eyo1)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/24/22** | **Matthew Muller** | **Part 1 and 2 completed** |
| **2.0** | **5/29/22** | **Matthew Muller** | **Rest of Document completed** |

## 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

Matthew Muller

## 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?

Since the client, Artemis Financial, is a financial consulting company, ensuring that the communications between them and their clients are secured is of paramount importance. Since it has been stated that the client helps customers around the world, making certain that the application secures international transactions is crucial. It is important that attention be paid to the different security risks, laws, and regulations of the various countries where the client does business. There are rules and regulations in place that punish financial institutions for neglecting to secure their communications with customers. Organizations such as the FDIC have helped to create laws related to the protection of customers’ financial information and the notification of customers when a data breach occurs. It is essential that the application be in accordance with all of these laws. The external threats to this application’s security come in the form of attackers trying to gain access to customers’ personal financial information. One common form of these attacks is an SQL injection. This is when an attacker inserts or “injects” an SQL query into an input form in an application, which can lead to them being able to access or even modify sensitive data from the application’s database. There are two main modernization requirements that must be considered for this application. The first is ensuring that the newest versions of open source libraries are being used to minimize vulnerabilities. The second is that https rather than http be used as the web application’s communication protocol in order to bolster the security of the client/server interactions.

## 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 -** This area of security is important to the application because customers will need to sign-in in order to access their financial information. It is essential that the application grants access to the proper information based on the sign-in credentials that it receives. Also, implementing protections from SQL injections that exploit input forms is crucial.

**APIs -** This area of security is important to the application because of its use of a RESTful web application programming interface (API). Since APIs act as a mediator between client requests and server responses, they are commonly targeted by attackers. Therefore, it is vital that the applications APIs be properly secured.

**Cryptography -** This area of security is important to the application due to the sensitive nature of the customer information that it uses. Since any leak of customer information will result in a breach of their financial privacy, ensuring that any communications including this information are encrypted is required for this application.

**Client/Server -** This area of security is important to the application because personal financial information will be sent between the client’s device and the server’s database. If these communications are not properly secured, attackers could intercept and exploit the customer’s information.

**Code Quality -** Every application can benefit from secure coding practices/patterns and Artemis Financial’s is no different. Strategies exist to protect web applications from the exploits that this program may face, such as SQL injections and various different API attacks. Implementing these strategies in order to protect customer information is a must for this application.

## 3. Manual Review

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

* Vulnerabilities in CRUD class:
  + public accessor and constructor methods
* Vulnerability in CRUDController class:
  + Vulnerable to code injections
* Vulnerabilities in customer class:
  + public accessor and mutator methods
* Vulnerabilities in DocData class:
  + public accessor and constructor methods
  + input being handled in the URL rather than via the post method could lead to data leaking into browser history
* Vulnerabilities in Greeting class:
  + public accessor and constructor methods
* Vulnerabilities of GreetingController class:
  + No input validation
* Lack of data encryption
* Lack of functioning API

## 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

* bcprov-jdk15on-1.46.jar
  + The Bouncy Castle Crypto package is a Java implementation of cryptographic algorithms. This jar contains JCE provider and lightweight API for the Bouncy Castle Cryptography APIs for JDK 1.5 to JDK 1.7.
  + The version of Bouncy Castle JCE Provider being used should be updated to one after version 1.55
* hibernate-validator-6.0.18.Final.jar
  + A flaw was found in hibernate-validator 6.1.2.Final. A bug in the interpolation of constraint error messages code enables invalid EL expressions to be evaluated as if they were valid. This bug enables attackers to bypass input sanitation (escaping, stripping) controls that developers may have put in place when handling user-controlled data in error messages.
  + This can be fixed by upgrading to hibernate-validator-6.0.20.Final
* jackson-databind-2.10.2.jar
  + General data-binding functionality for Jackson: works on core streaming API
  + Version of jackson-databind being used should be updated to 2.13.0 or newer.
* log4j-api-2.12.1.jar
  + The Apache Log4j API
  + This issue is fixed by limiting JNDI data source names to the java protocol. It is also recommended to upgrade the apache-log4j2 package being used.
* logback-core-1.2.3.jar
  + logback-core module
  + 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.
  + This vulnerability can be mitigated by upgrading to a version of logback newer than 1.2.7
* snakeyaml-1.25.jar
  + YAML 1.1 parser and emitter for Java
  + The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564.
  + This vulnerability can be mitigated by upgrading snakeyaml to 1.26
* spring-aop-5.2.3.RELEASE.jar
  + Spring AOP
  + In Spring Framework versions 5.3.0 - 5.3.16, 5.2.0 - 5.2.19, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition.
  + This issue can be mitigated accordingly: 5.3.x users should upgrade to 5.3.17+. 5.2.x users should upgrade to 5.2.20+
* spring-boot-2.2.4.RELEASE.jar
  + Spring Boot
  + In Spring Framework versions 5.3.0 - 5.3.16, 5.2.0 - 5.2.19, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition.
  + This vulnerability is patched in versions v2.2.11.RELEASE or later.
* spring-core-5.2.3.RELEASE.jar
  + Spring Core
  + In Spring Framework versions 5.3.0 - 5.3.16, 5.2.0 - 5.2.19, and older unsupported versions, it is possible for a user to provide a specially crafted SpEL expression that may cause a denial of service condition.
  + This issue can be mitigated accordingly: 5.3.x users should upgrade to 5.3.17+. 5.2.x users should upgrade to 5.2.20+
* tomcat-embed-core-9.0.30.jar
  + Core Tomcat implementation
  + The documentation for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.
  + Users running clustering over an untrusted network who require full protection should switch to an alternative solution such as running the clustering communication over a VPN.
* tomcat-embed-websocket-9.0.30.jar
  + Core Tomcat implementation
  + The documentation for the EncryptInterceptor incorrectly stated it enabled Tomcat clustering to run over an untrusted network. This was not correct. While the EncryptInterceptor does provide confidentiality and integrity protection, it does not protect against all risks associated with running over any untrusted network, particularly DoS risks.
  + Users running clustering over an untrusted network who require full protection should switch to an alternative solution such as running the clustering communication over a VPN.

## 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 in mitigating the security concerns in Artemis Financial’s web application is to update all dependencies to their newest versions. This will implement a plethora of bug fixes that will help to protect the application from many of the attacks that it is currently vulnerable to. The next step would be to switch to HTTPS as the web application’s communication protocol. This will help to make the client/server interactions more secure. Another vulnerability that needs to be addressed is all of the public accessor, mutator, and constructor methods in the various classes. Also, input validation and error handling methods need to be implemented in a couple of areas. Finally, it is essential that the development team incorporate cryptography strategies in order to protect customer information.