# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **May 18, 2024** | **James Emery** |  |

## Client



## *Developer*

***James Emery***

**1. Interpreting Client Needs**

As a financial organization that assists clients with both internal and external priority information, Artemis Financial understands the value of safe internal and external communication. Furthermore, given that Artemis Financial is a financial company, I would anticipate that transactions will originate from both domestic and foreign sources. There are currently no state or federal laws limiting secure communications, but security should still be maintained in communication. Artemis Financial must make sure that all their internal and external communications are secure to prevent customer information from being leaked.

In addition to any trade secrets maintained within Artemis Financial, Artemis Financial will handle financial information as well as proprietary client information like biometric data, social security numbers, and account information. As a result, information must be masked both during storage and transmission. In terms of modernization, Artemis Financial must make sure that the libraries in their application are kept up to date to guarantee that the most recent bug and security threat updates are applied.

**2. Areas of Security**

Upon evaluating the security aspects of Artemis Financial, I've discovered some possible weaknesses.

***Input validation:*** Validation is essential for gathering user input. Given that this application permits input string input, validation is required to prevent potential errors or SQL injection.

***APIs:*** A well-developed API will be required because this application will operate both internally and outside, such as on the web browser of the end user. This API should decide which data access techniques are appropriate since it will specify how a user will interact with the software. It will be essential to provide a secure API because this program might interact with third-party software.

***Cryptography:*** Since the transfers will involve international transfers containing confidential customer data, cryptography is required. The security of the data must be such that it complies with all regulations in the country of destination as well as those in North America.

***Code Error:*** This ought to be used in conjunction with input validation and the API. To prevent unauthorized access or privilege access breaches, proper error handling will be essential, particularly when dealing with input validation.

***Code quality***: To prevent inadvertent data exposure when interacting with end users, including input and APIs, code quality is essential. Additionally, it will prevent end users who are not granted access by their user levels from using the methods.

**3. Manual Review**

My initial step involved looking for any Apache validators in the POM.XML file. Next, we entered the greeting controller; it didn't seem that any validation was applied to this input. Once more, there was no output, so I was unable to confirm if this was validated. The application was still able to access data that was granted in an unprotected manner when I next looked for an API, but I was unable to find any that was operational. Instead of using the POST technique, which can allow data to leak into the browser history and be exploited, the application obtains the data via the URL. Although it doesn't display anything, this software still takes input via the URL, which means that it can still be exploited.

The program lacks an API, therefore the only way an end user can interact with it is via accessing the code. A unique interface for user interaction is required for an API to be created as a RESTful API. After completing the API and input validation, I turned my attention to the cryptography check. There was absolutely no data encryption that I could find. To send and receive foreign transactions that abide by global standards, Artemis Financial would need to create some kind of data encryption.

Upon searching the code for error handling, I discovered that the DocData.java class only included try and catch blocks and had no additional error handling assessed. Finally, the software was very user-friendly due to the poor code quality, which was partially functional due to the lack of an API. Furthermore, inadequate input validation and handling of input through the URL rather than the post method may result in data compromise and browser history leakage.

**4. Static Testing**

***log4j-api-2.12.1.jar***

cpe:2.3:a:apache:log4j:2.12.1

Improper validation of certificate with host mismatch in Apache Log4j SMTP appender. This could allow an SMTPS connection to be intercepted by a man-in-the-middle attack which could leak any log messages sent through that appender. Upgrade to 2.13.2 which supports this feature. Previous versions can set the system property mail.smtp.ssl.checkserveridentity to true to globally enable hostname verification for SMTPS connections.

***tomcat-embed-core-9.0.30.jar***

cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_software\_foundation:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

Apache Tomcat 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46 and 8.5.0 to 8.5.66 did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. Specifically, Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response; - Tomcat honored the identify encoding; and - Tomcat did not ensure that, if present, the chunked encoding was the final encoding.

Upgrade to Apache Tomcat 10.0.6 or later.

***tomcat-embed-websocket-9.0.30.jar***

cpe:2.3:a:apache:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_software\_foundation:tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:apache\_tomcat:apache\_tomcat:9.0.30:\*:\*:\*:\*:\*:\*:\*

Apache Tomcat 10.0.0-M1 to 10.0.6, 9.0.0.M1 to 9.0.46 and 8.5.0 to 8.5.66 did not correctly parse the HTTP transfer-encoding request header in some circumstances leading to the possibility to request smuggling when used with a reverse proxy. Specifically: - Tomcat incorrectly ignored the transfer encoding header if the client declared it would only accept an HTTP/1.0 response; - Tomcat honored the identify encoding; and Tomcat did not ensure that, if present, the chunked encoding was the final encoding.

Upgrade to Apache Tomcat 10.0.6 or later.

***bcprov-jdk15on-1.46.jar***

cpe:2.3:a:bouncycastle:bouncy-castle-crypto-package:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:bouncy\_castle\_crypto\_package:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:legion-of-the-bouncy-castle-java-crytography-api:1.46:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:bouncycastle:the\_bouncy\_castle\_crypto\_package\_for\_java:1.46:\*:\*:\*:\*:\*:\*:\*

Legion of the Bouncy Castle Legion of the Bouncy Castle Java Cryptography APIs 1.58 up to but not including 1.60 contains a CWE-470: Use of Externally Controlled Input to Select Classes or Code ('Unsafe Reflection') vulnerability in XMSS/XMSS^MT private key deserialization that can result in Deserializing an XMSS/XMSS^MT private key can result in the execution of unexpected code. This attack appears to be exploitable via A handcrafted private key can include references to unexpected classes which will be picked up from the class path for the executing application. This vulnerability appears to have been fixed in 1.60 and later.

update bouncycastle to:Version update to 1.60

***jackson-databind-2.10.2.jar***

cpe:2.3:a:fasterxml:jackson-databind:2.10.2:\*:\*:\*:\*:\*:\*:\*

cpe:2.3:a:fasterxml:jackson-modules-java8:2.10.2:\*:\*:\*:\*:\*:\*:\*

A flaw was found in FasterXML Jackson Databind, where it did not have entity expansion secured properly. This flaw allows vulnerability to XML external entity (XXE) attacks. The highest threat from this vulnerability is data integrity.

Update to current version.

***spring-aop-5.2.3.RELEASE.jar***

cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

cpe:2.3:a:springsource:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

cpe:2.3:a:vmware:spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

cpe:2.3:a:vmware:springsource\_spring\_framework:5.2.3:release:\*:\*:\*:\*:\*:\*

In Spring Framework versions 5.2.0 - 5.2.8, 5.1.0 - 5.1.17, 5.0.0 - 5.0.18, 4.3.0 - 4.3.28, and older unsupported versions, the protections against RFD attacks from CVE-2015-5211 may be bypassed depending on the browser used with a jsessionid path parameter.

Upgrade to current version.

***hibernate-validator-6.0.18.Final.jar***

cpe:2.3:a:redhat:hibernate\_validator:6.0.18:\*:\*:\*:\*:\*:\*:\*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.

Upgrade to hibernate-validator-6.0.20

***snakeyaml-1.25.jar***

cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25:\*:\*:\*:\*:\*:\*:\*

The Alias feature in SnakeYAML 1.18 allows entity expansion during a load operation, a related issue to CVE-2003-1564.

Migrate to SnakeYAML Engine. It has a configuration option to restrict aliases for collections (the aliases for scalars cannot grow and they are not restricted)

**5. Mitigation Plan**

Examine the manual review and static testing report results to determine which software application security flaws Artemis Financial has. Next, describe how to fix these security holes.   
  
Many of these problems can be fixed by updating to the most recent versions of the software. To address certain vulnerabilities, edit Snakeyaml and restrict aliases for collections.

.