# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **Jan 21 2025** | **Chris Bridges** |  |

## Client



## Instructions

Submit this completed vulnerability assessment report. Replace the bracketed text with the relevant information. In this report, identify your security vulnerability findings and recommend the next steps to remedy the issues you have found.

* Respond to the five steps outlined below and include your findings.
* Respond using your own words. You may also include images or supporting materials. If you include them, make certain to insert them in the relevant locations in the document.
* Refer to the Project One Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Chris Bridges

**1. Interpreting Client Needs**

Determine your client’s needs and potential threats and attacks associated with the company’s application and software security requirements. Consider the following questions regarding how companies protect against external threats based on the scenario information:

* What is the value of secure communications to the company?
  1. Artemis Financial requires secure communication for their company to function. Using industry standard protocols such as HTTPS and SSL/TLS certificates could enhance their security and meet the expectations.
* Are there any international transactions that the company produces?
  1. Finance rarely tends to be a domestic affair, so the necessary protocols must be met to allow for international transactions. General Data Protection Regulation must be met in order for Artemis Financial to service clients abroad (GDPR, 2025).
* Are there governmental restrictions on secure communications to consider?
  1. There are numerous governmental standards and restrictions that must be considered when building a financial platform. Secure and encrypted information and data transfers must be considered first and foremost.
* What external threats might be present now and in the immediate future?
  1. Ransomware and Phishing attacks are a concern when dealing with information and finances. These attacks could be focused on clients, the company, or the staff. Effective training and communication with staff and employees could reduce the risk of a phishing attack. Secure email and communication methods within the finance company could reduce the risk of ransomware and phishing attacks.
* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?
  1. Multiple factors regarding modernization and expansion would be considered when developing this software. A web application would be the most effective because it is compatible with multiple operating systems or browsers. Creating modular code that can be modified for new phone models and security changes would improve the application's lifespan. Open source libraries require update patches to maintain their security and should be updated as soon as a new version is released.

**2. Areas of Security**

Refer to the vulnerability assessment process flow diagram. Identify which areas of security apply to Artemis Financial’s software application. Justify your reasoning for why each area is relevant to the software application.

* Input validation is required to reduce the risk of SQL injection and other forms of manipulation.
* Secure API interactions allow for proper authentication, error handling, and encryption.
* Cryptography allows for data to be stored and transmitted in a safer manner.
* Client/Server interactions should be secure using industry standard methods to reduce the risk of man in the middle attacks or other forms of interference.
* Code errors should be logged and reported to the proper employees to correct, reducing diagnostic time and a downed system.
* Code should adhere to industry standard with security, functionality, and modularity as the focus.
* Encapsulation mitigates the risk of unauthorized access to data or accounts.

**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 is no encryption in the customer.java class, which would lead to account balance and account number being easily obtained.
* The CRUD method in the CRUD controller class has no input validation making it vulnerable to SQL injection attacks.
* CRUD controller lacks authentication and could allow anyone to interact with the API
* Read document in the DocData class has no input validation and could also introduce vulnerabilities to SQL injections.
* Printing the stack trace in the read document class could unintentionally expose unnecessary data.
* In myDateTime, the array can be altered by other parts of the application which can cause multiple issues.
* There is no logging for API calls or errors
* In the Customer class, account\_balance is public which could lead to manipulation.

**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 the dependency-check report. Include the following items:

* **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
  + Software communicates with the host that provides a certificate, but doesn’t ensure the certificate is associated with the host.
* **hibernate-validator-6.0.18.Final.jar**
  + Hibernate's Bean Validation (JSR-380) reference implementation.
  + A flaw was found in hibernate-validator's 'isValid' method in the org.hibernate.validator.internal.constraintvalidators.hv.SafeHtmlValidator class, which can be bypassed by omitting the tag ending in a less-than character. Browsers may render an invalid html, allowing HTML injection or Cross-Site-Scripting (XSS) attacks.
  + Sonatype's research suggests that this CVE's details differ from those defined at NVD. See https://ossindex.sonatype.org/vulnerability/CVE-2023-1932 for details
* **jackson-databind-2.10.2.jar**
  + General data-binding functionality for Jackson: works on core streaming API
  + 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.
* **log4j-api-2.12.1.jar**
  + The Apache Log4j API
  + 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. Fixed in Apache Log4j 2.12.3 and 2.13.1
* **logback-classic-1.2.3.jar**
  + logback-classic module
  + A serialization vulnerability in logback receiver component part of logback version 1.4.11 allows an attacker to mount a Denial-Of-Service attack by sending poisoned data.
* **logback-core-1.2.3.jar**
  + logback-core module
  + A serialization vulnerability in logback receiver component part of logback version 1.4.11 allows an attacker to mount a Denial-Of-Service attack by sending poisoned data.
* **snakeyaml-1.25.jar**
  + YAML 1.1 parser and emitter for Java
  + SnakeYaml's Constructor() class does not restrict types which can be instantiated during deserialization. Deserializing yaml content provided by an attacker can lead to remote code execution. We recommend using SnakeYaml's SafeConsturctor when parsing untrusted content to restrict deserialization. We recommend upgrading to version 2.0 and beyond.
* **spring-boot-2.2.4.RELEASE.jar**
  + Spring Boot
  + In Spring Boot versions 3.0.0 - 3.0.5, 2.7.0 - 2.7.10, and older unsupported versions, an application that is deployed to Cloud Foundry could be susceptible to a security bypass. Users of affected versions should apply the following mitigation: 3.0.x users should upgrade to 3.0.6+. 2.7.x users should upgrade to 2.7.11+. Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+.
* **spring-boot-starter-web-2.2.4.RELEASE.jar**
  + Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container.
  + In Spring Boot versions 3.0.0 - 3.0.5, 2.7.0 - 2.7.10, and older unsupported versions, an application that is deployed to Cloud Foundry could be susceptible to a security bypass. Users of affected versions should apply the following mitigation: 3.0.x users should upgrade to 3.0.6+. 2.7.x users should upgrade to 2.7.11+. Users of older, unsupported versions should upgrade to 3.0.6+ or 2.7.11+.
* **spring-core-5.2.3.RELEASE.jar**
  + Spring Core
  + Product: VMware Spring Framework
  + Name: Spring Framework JDK 9+ Remote Code Execution Vulnerability
  + Date Added: 2022-04-04
  + Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-04-25
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2022-22965>
  + A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.
* **spring-expression-5.2.3.RELEASE.jar**
  + Spring Expression Language (SpEL)
  + Product: VMware Spring Framework
  + Name: Spring Framework JDK 9+ Remote Code Execution Vulnerability
  + Date Added: 2022-04-04
  + Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-04-25
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2022-22965>
  + A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.
* **spring-web-5.2.3.RELEASE.jar**
  + Spring Web
  + Product: VMware Spring Framework
  + Name: Spring Framework JDK 9+ Remote Code Execution Vulnerability
  + Date Added: 2022-04-04
  + Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-04-25
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2022-22965>
  + A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.
* **spring-webmvc-5.2.3.RELEASE.jar**
  + Spring Web MVC
  + Product: VMware Spring Framework
  + Name: Spring Framework JDK 9+ Remote Code Execution Vulnerability
  + Date Added: 2022-04-04
  + Description: Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-04-25
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2022-22965>
  + A Spring MVC or Spring WebFlux application running on JDK 9+ may be vulnerable to remote code execution (RCE) via data binding. The specific exploit requires the application to run on Tomcat as a WAR deployment. If the application is deployed as a Spring Boot executable jar, i.e. the default, it is not vulnerable to the exploit. However, the nature of the vulnerability is more general, and there may be other ways to exploit it.
* **tomcat-embed-core-9.0.30.jar**
  + Core Tomcat implementation
  + Product: Apache Tomcat
  + Name: Apache Tomcat Improper Privilege Management Vulnerability
  + Date Added: 2022-03-03
  + Description: Apache Tomcat treats Apache JServ Protocol (AJP) connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-03-17
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2020-1938>
  + When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.
* **tomcat-embed-websocket-9.0.30.jar**
  + Core Tomcat implementation
  + Product: Apache Tomcat
  + Name: Apache Tomcat Improper Privilege Management Vulnerability
  + Date Added: 2022-03-03
  + Description: Apache Tomcat treats Apache JServ Protocol (AJP) connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited.
  + Required Action: Apply updates per vendor instructions.
  + Due Date: 2022-03-17
  + Notes: <https://nvd.nist.gov/vuln/detail/CVE-2020-1938>
  + When using the Apache JServ Protocol (AJP), care must be taken when trusting incoming connections to Apache Tomcat. Tomcat treats AJP connections as having higher trust than, for example, a similar HTTP connection. If such connections are available to an attacker, they can be exploited in ways that may be surprising. In Apache Tomcat 9.0.0.M1 to 9.0.0.30, 8.5.0 to 8.5.50 and 7.0.0 to 7.0.99, Tomcat shipped with an AJP Connector enabled by default that listened on all configured IP addresses. It was expected (and recommended in the security guide) that this Connector would be disabled if not required. This vulnerability report identified a mechanism that allowed: - returning arbitrary files from anywhere in the web application - processing any file in the web application as a JSP Further, if the web application allowed file upload and stored those files within the web application (or the attacker was able to control the content of the web application by some other means) then this, along with the ability to process a file as a JSP, made remote code execution possible. It is important to note that mitigation is only required if an AJP port is accessible to untrusted users. Users wishing to take a defence-in-depth approach and block the vector that permits returning arbitrary files and execution as JSP may upgrade to Apache Tomcat 9.0.31, 8.5.51 or 7.0.100 or later. A number of changes were made to the default AJP Connector configuration in 9.0.31 to harden the default configuration. It is likely that users upgrading to 9.0.31, 8.5.51 or 7.0.100 or later will need to make small changes to their configurations.

**5. Mitigation Plan**

Interpret the results from the manual review and static testing report. Then identify the steps to mitigate the identified security vulnerabilities for Artemis Financial’s software application.

Many things could be improved to increase the security of the Artemis Financial application. The coding could include some changes to enhance input validation and reduce the risk of SQL injection attacks. Encryption could be included to solve some of the plain text information being transmitted. Using a secure communication protocol like HTTPS would allow client/server information to be more secure. The critical issues with the Spring framework could be resolved by updating to the latest version which would solve those vulnerabilities. Upgrading to the latest version of each of the dependencies would resolve most of the security issues within the framework.