# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **11/19/2024** | **Abdulrahman Al-Nachar** |  |

## Client



## Developer

Abdulrahman Al-Nachar

**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:

Artemis Financial is a company in the finance sector, its goal is to help customers with their finances by providing financial services that manage the client’s financial portfolio including their retirement plan, investments plan, insurance, and savings, Artemis Financial Services must secure their data and provide a service that is secure and safe to use since Artemis Financial is a company in the finance sector they will have international clients as well as having local US based clients Since the company is based in the US they would have to follow US regulations like the Gramm-Leach-Billey act of 2001, The program provided can several external threats that would undermine its security and safety and functionality, like injection attacks, access by unwanted actors or unauthorized controls, and potential data leaks, it’s crucial to modernize vulnerable software used like dependencies in the program to help bolster its security.

**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.

1- Input validation: the program needs to validate inputs to protect the program from injections that can reveal sensitive data, system information, and more.

2- secure communications within the API to protect the interactions between client and server

3- Cryptography: Crypto is a must for this program since it deals with sensitive data to prevent unauthorized access to data.

**3. Manual Review**

1- Lack of HTTPS, this program isn’t secured by any security.

2- in the CRUDController file, the inputs getting in aren’t being validated, which can expose the program to injection attacks

3- in the CRUDController file, errors aren’t handled by the program. Which can expose the program to information leaks and possible exploitation.

4- in the DocData file, the inputs getting in aren’t being validated , which can expose the program to injection attacks

5- The program lacks data encryption, like in DocData & Customer file.

6- Change the names of some of the methods in the program to follow secure coding guidelines and make the program easier for developers to maintain.

7- The program lacks proper authentication & authorization mechanisms.

8- Old dependencies are being used in the program that are exposed to cyber-attacks.

**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:

* The names or vulnerability codes of the known vulnerabilities
* A brief description and recommended solutions provided by the dependency-check report
* Any attribution that documents how this vulnerability has been identified or documented previously

Dependency Vulnerability ID PackageSeverity

|  |  |  |  |
| --- | --- | --- | --- |
| bcprov-jdk15on-1.46.jar | cpe:2.3:a:bouncycastle:bouncy\_castle\_for\_java:1.46 | pkg:maven/org.bouncycastle/bcprov-jdk15on@1.46 | HIGH |
| hibernate-validator-6.0.18.Final.jar | cpe:2.3:a:redhat:hibernate\_validator:6.0.18 | /org.hibernate.validator/hibernate-validator@6.0.18.Final | MEDIUM |
| jackson-databind-2.10.2.jar | cpe:2.3:a:fasterxml:jackson-databind:2.10.2 | /com.fasterxml.jackson.core/jackson-databind@2.10.2 | HIGH |
| log4j-api-2.12.1.jar | cpe:2.3:a:apache:log4j:2.12.1 | /org.apache.logging.log4j/log4j-api@2.12.1 | LOW |
| logback-core-1.2.3.jar | cpe:2.3:a:qos:logback:1.2.3 | /ch.qos.logback/logback-core@1.2.3 | HIGH |
| snakeyaml-1.25.jar | cpe:2.3:a:snakeyaml\_project:snakeyaml:1.25 | /org.yaml/snakeyaml@1.25 | CRITICAL |
| spring-boot-2.2.4.RELEASE.jar | cpe:2.3:a:vmware:spring\_boot:2.2.4 | /org.springframework.boot/spring-boot@2.2.4.RELEASE | CRITICAL |
| spring-boot-starter-web-2.2.4.RELEASE.jar | cpe:2.3:a:vmware:spring\_boot:2.2.4 | /org.springframework.boot/spring-boot-starter-web@2.2.4.RELEASE | CRITICAL |
| spring-core-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3 | /org.springframework/spring-core@5.2.3.RELEASE | CRITICAL\* |
| spring-expression-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3 | /org.springframework/spring-expression@5.2.3.RELEASE | CRITICAL\* |
| spring-web-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3 | /org.springframework/spring-web@5.2.3.RELEASE | CRITICAL\* |
| spring-webmvc-5.2.3.RELEASE.jar | cpe:2.3:a:pivotal\_software:spring\_framework:5.2.3 | /org.springframework/spring-webmvc@5.2.3.RELEASE | CRITICAL\* |
| tomcat-embed-core-9.0.30.jar | cpe:2.3:a:apache:tomcat:9.0.30 | /org.apache.tomcat.embed/tomcat-embed-core@9.0.30 | CRITICAL\* |
| tomcat-embed-websocket-9.0.30.jar | cpe:2.3:a:apache:tomcat:9.0.30 | /org.apache.tomcat.embed/tomcat-embed-websocket@9.0.30 | CRITICAL\* |
|  |  |  |  |

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

**Problem:** In Bouncy Castle JCE Provider version 1.55 and earlier the DSA does not fully validate ASN.1 encoding of signature on verification. It is possible to inject extra elements in the sequence making up the signature and still have it validate, which in some cases may allow the introduction of 'invisible' data into a signed structure.

**Fix:** update Bouncy Castle to the newest version.

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

**Problem:** 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.

**Fix:** update Hibernate Validator to the newest version.

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

**Problem:** 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.

**Fix:** Upgrade com.fasterxml.jackson.core:jackson-databind to version 2.6.7.4 or higher.

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

**Problem:** 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

**Fix:** update log4j-api-2.12.1. to 2.12.3 .

**logback-core-1.2.3.jar:**

**Problem:** 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.

**Fix:** update logback-core-1.2.3. to 1.2.13.

**snakeyaml-1.25.jar:**

**Problem:** 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.

**Fix:** this issue has been fixed in the SnakeYAML 2.0 version

**spring-boot-2.2.4.RELEASE.jar:**

**Problem:** 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+.

**Fix:** update spring-boot-2.2.4. to the newest supported 3.0.6 version.

**spring-boot-starter-web-2.2.4.RELEASE.jar:**

**Problem:** In Spring Boot versions 3.0.0 - 3.0.6, 2.7.0 - 2.7.11, 2.6.0 - 2.6.14, 2.5.0 - 2.5.14 and older unsupported versions, there is potential for a denial-of-service (DoS) attack if Spring MVC is used together with a reverse proxy cache.

**Fix: :** update spring-boot-2.2.4. to the newest supported 3.0.6 version.

**spring-core-5.2.3.RELEASE.jar:**

**Problem:** 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.

**Fix:** upgrade spring framework to the newest version.

**spring-expression-5.2.3.RELEASE.jar:**

**Problem:** In spring framework versions prior to 5.2.24 release+ ,5.3.27+ and 6.0.8+ , it is possible for a user to provide a specially crafted SpEL expression that may cause a denial-of-service (DoS) condition.

**Fix:** upgrade spring framework to the newest version.

**spring-web-5.2.3.RELEASE.jar:**

**Problem:** 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 through the use of a jsessionid path parameter.

**Fix:** upgrade spring framework to the newest version.

**spring-webmvc-5.2.3.RELEASE.jar:**

**Problem:** In spring framework versions prior to 5.3.20+ , 5.2.22+ and old unsupported versions, application with a STOMP over WebSocket endpoint is vulnerable to a denial of service attack by an authenticated user.

**Fix:** upgrade spring framework to the newest version.

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

**Problem:** 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.

**Fix:** Apache released this version that fixes this 9.0.31.

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

**Problem:** Apache Tomcat 8.5.0 to 8.5.63, 9.0.0-M1 to 9.0.43 and 10.0.0-M1 to 10.0.2 did not properly validate incoming TLS packets. When Tomcat was configured to use NIO+OpenSSL or NIO2+OpenSSL for TLS, a specially crafted packet could be used to trigger an infinite loop resulting in a denial of service.

**Fix:** update apache tomcat to the latest supported version.

**5. Mitigation Plan**

1- Secure the program with HTTPS.

2- validate inputs in the CRUDController file to protect the program from injection attacks.

3- Handle errors & manage error outputs in the program (CRUDController, GreetingController, customer, etc) , to protect the program from information leaks and exploitation

4- Validate inputs in the DocData file.

5- Encrypt sensitive data found in the program like methods found in DocData and Customer file.

6- Adhere to Java naming conventions to make it easier for developers to maintain the program, like the method and variables found in customer file.

7- Add authentication & authorization mechanisms in the program to protect against unwanted controls & unwanted actors, like the endpoint in CRUDController and GreetingController , etc.

8- Upgrade the dependencies used in the program to their newest versions.