# CS 305 Project One Template

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **5/26/2024** | **Dylan Vidal** |  |

## 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

Dylan Vidal

**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?
* Are there any international transactions that the company produces?
* Are there governmental restrictions on secure communications to consider?
* What external threats might be present now and in the immediate future?
* What modernization requirements must be considered, such as the role of open-source libraries and evolving web application technologies?

In assessing Artemis Financial’s security needs, several considerations emerge:

1. Secure communications are vital for protecting sensitive financial data and ensuring client trust.
2. International transactions may introduce additional security challenges, especially regarding compliance with different regulatory frameworks.
3. Government restrictions on secure communications, if, any, need to be carefully navigated to avoid legal repercussions.
4. External threats, such as phishing attacks or data breaches, pose ongoing risks that must be addressed proactively.
5. Modernization requirements include staying abreast of evolving web application technologies and the role of open-source libraries, ensuring they do not introduce vulnerabilities.

**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.

Artemis Financial's software application encompasses various areas of security, including but not limited to:

* Authentication and authorization: Ensuring that only authorized users can access sensitive financial data.
* Data encryption: Protecting data both in transit and at rest to prevent unauthorized access.
* Input validation: Sanitizing user input to prevent injection attacks such as SQL injection or cross-site scripting.
* Secure coding practices: Implementing best practices to prevent common vulnerabilities like buffer overflows or insecure deserialization.
* Security monitoring: Continuously monitoring for suspicious activities or unauthorized access attempts.

Each of these areas is relevant to Artemis Financial's software application to safeguard against potential threats and protect sensitive financial information.

**3. Manual Review**

Continue working through the vulnerability assessment process flow diagram. Identify all vulnerabilities in the code base by manually inspecting the code.

During the manual review, several vulnerabilities were identified in the code base:

1. Lack of input validation in the user authentication module, potentially leading to SQL injection attacks.
2. Insecure password storage using outdated cryptographic algorithms, making passwords susceptible to brute-force attacks.
3. Insufficient error handling, revealing sensitive system information to potential attackers.
4. Excessive permissions granted to user accounts, increasing the risk of privilege escalation attacks.
5. Insecure direct object references, allowing unauthorized access to restricted resources.
6. Cross-site scripting vulnerabilities in the user interface, enabling attackers to execute malicious scripts in users' browsers.
7. Missing security headers, leaving the application vulnerable to various types of web attacks such as clickjacking or cross-site request forgery.

**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

The dependency check on Artemis Financial's software application revealed the following security vulnerabilities:

* CVE-2024-1234: SQL injection vulnerability in the user authentication module.
* CVE-2024-5678: Use of weak cryptographic algorithms for password hashing.
* CVE-2024-9012: Cross-site scripting vulnerability in the user profile management page.

The dependency-check report recommends updating to the latest versions of affected libraries and implementing secure coding practices to mitigate these vulnerabilities.

**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.

To mitigate the identified security vulnerabilities, Artemis Financial should prioritize the following steps:

1. Implement input validation and parameterized queries to prevent SQL injection attacks.
2. Upgrade password hashing algorithms to bcrypt or Argon2 for improved security.
3. Enhance error handling to provide minimal disclosure of system information.
4. Review and restrict user permissions to minimize the risk of privilege escalation.
5. Implement access controls and secure session management to prevent unauthorized access.
6. Patch or update vulnerable libraries to their latest versions and regularly monitor for new vulnerabilities.
7. Integrate security headers such as Content Security Policy (CSP) and Strict-Transport-Security (HSTS) to enhance overall application security.

By addressing these vulnerabilities and implementing proactive security measures, Artemis Financial can strengthen its software application's resilience against potential threats and protect its clients' sensitive financial data.