* Briefly summarize your client, Artemis Financial, and its software requirements. Who was the client? What issue did the company want you to address?

Artemis Financial is a financial consulting organization that develops individualized financial plans for its customers to include products such as savings, retirement, investments, and insurance. In an effort to modernize the security of its public web application, the organization requested Global Rain’s expertise on how to protect its client’s data and financial information through employing a system that resists potential attacks, protects financial data, and complies with industry best practices. As a developer, my role was to analyze their codebase for any potential vulnerabilities, risks, and provide professional insights into strengthening the security posture of their software.

* What did you do well when you found your client’s software security vulnerabilities? Why is it important to code securely? What value does software security add to a company’s overall well-being?

Throughout my vulnerability assessment, I performed both manual code reviews and running a dependency check utilizing the open-source Maven dependency check. Through this, I documented all vulnerabilities, to include risk level and the associated impact; allowing Artemis Financial to prioritize identified critical issues and shore-up their security posture. With respect to secure code, it is vital for organizations to employ security throughout the development process; particularly through the implementation of DevSecOps—where security is no longer an afterthought, but a continuous process embedded in development. Moreover, such implementation of secure code ultimately reduces the risk of attacks like data breaches, which can harm customer trust and lead to financial and reputation damage. As such, software security’s role for a company’s overall well-being is valued through protecting company assets, ensuring compliance with regulatory requirements, and improving overall system reliability.

* Which part of the vulnerability assessment was challenging or helpful to you?

Fortunately, all aspects of the vulnerability assessment were extremely helpful; especially the dependency check with Maven, as it was particularly helpful as it automated the process of finding known vulnerabilities in third-party libraries. Practicing the use of dependency checks, manual code review, reading documentation, and learning more about web application security were extremely useful. Conversely, while I enjoyed the manual code review, I found that some aspects of manually inspecting the code to identify potential issues such as input validation/sanitation or exception handling was quite challenging for me as it requires a deep understanding of the software architecture and secure coding practices.

* How did you increase layers of security? In the future, what would you use to assess vulnerabilities and decide which mitigation techniques to use?

Understanding the foundational security aspects within Java web application deployment and reading the helped me to increase security layers by identifying specific vulnerabilities, such as outdated dependencies and insufficient input validation, and suggesting specific mitigations such as updating dependencies and implementing stricter validation. As a measure for future considerations, I would employ the use of tools like the OWASP Maven Dependency-Check, integrated static analysis tools (e.g., PMD plugin in Eclipse, ), and ultimately just follow secure coding guidelines as recommended by the OWASP Top 10. Additionally, I would select mitigation techniques that specifically prioritize identified vulnerabilities based on the severity, exploitability, and impact on the application.

* How did you make certain the code and software application were functional and secure? After refactoring the code, how did you check to see whether you introduced new vulnerabilities?

In an effort to ensure the Java web application remained in a functional and secure state, I consistently tested the refactored code after each change to confirm it behaved as expected and didn’t introduce errors. As a good practice in programming, it’s always good to recompile the code after changes were implemented to ensure no additional bugs were introduced throughout development. Additionally, as a precaution, I also consistently re-ran the Maven dependency check within Eclipse and crosschecked the information with previously saved dependency-check reports. I believe that this iterative approach within the software development process helped to maintain both functionality and security of Artemis Financials’ codebase.

* What resources, tools, or coding practices did you use that might be helpful in future assignments or tasks?

After learning how to utilize the Maven dependency check plugin within Eclipse, I think it has become an invaluable tool for identifying vulnerabilities or CVEs within third-party libraries. Furthermore, the practice of secure coding practices like input validation/sanitation, exception handling, and adhering to OWASP guidelines were essential. Lastly, the documenting of vulnerabilities, both within the dependency check and with manual code review, was key for prioritizing the identified threats based on their risk level and will serve as a cornerstone on future assignments or related tasks.

* Employers sometimes ask for examples of work that you have successfully completed to show your skills, knowledge, and experience. What might you show future employers from this assignment?

This completed assignment will demonstrate to future employers my ability to work within the DevSecOps field by performing security-related programming tasks, such as conducting a comprehensive vulnerability assessment, identifying security issues, and providing actionable solutions to customers. Additionally, I can showcase my proficiency with tools like Maven, my understanding of secure coding practices, and my ability to document findings effectively.

While not directly demonstrated in this project, I also have experience using tools like Burp Suite, which are valuable for identifying vulnerabilities in web applications and ensuring that RESTful applications meet secure coding standards. Ultimately, this project reflects some of the technical software security skills I have developed at SNHU. While not exhaustive, it effectively demonstrates my ability to analyze, prioritize, and communicate vulnerabilities clearly and confidently.