Competency

In this project, you will demonstrate your mastery of the following competency:

- Analyze how advanced security concepts are applied to develop secure code

Scenario

You work as a developer for a software engineering company, Global Rain, that specializes in custom software design and development. The software is for entrepreneurs, businesses, and government agencies around the world. Part of the company's mission is that "Security is everyone's responsibility." Global Rain has promoted you to their new agile scrum team.

At Global Rain, you work with a client, Artemis Financial, a consulting company that develops individualized financial plans for their customers. The financial plans include savings, retirement, investments, and insurance.

Artemis Financial wants to modernize their operations. As a crucial part of the success of their custom software, they also want to use the most current and effective software security. Artemis Financial has a RESTful web application programming interface (API). They are seeking Global Rain's expertise about how to protect the organization from external threats.

As part of the team, you must examine Artemis Financial's web-based software application to identify any security vulnerabilities. You'll document what you learn in a vulnerability assessment report that will be used for mitigating the security vulnerabilities that you find.

Directions

You must conduct a vulnerability assessment. In it, you'll examine Artemis Financial's web-based software application. Use what you have learned so far and the resources provided in the Supporting Materials section to help you. Review and analyze the security vulnerabilities specific to Artemis Financial's web-based software application. Use the Project One Template, linked in What to Submit, to document the following for your vulnerability assessment report:

1. **Interpreting Client Needs**: Review the scenario to determine your client's needs and potential threats and attacks associated with their application and software security requirements. Document your findings in your vulnerability assessment report. Consider the scenario information and the following questions regarding how companies protect against external threats:
    a. What is the value of secure communications to the company?
    b. Does the company make any international transactions?
    c. Are there governmental restrictions about secure communications to consider?
    d. What external threats might be present now and in the immediate future?
    e. What are the modernization requirements that you must consider? For example:
        i. The role of open-source libraries
        ii. Evolving web application technologies
2. **Areas of Security**: Use what you've learned in step 1 and refer to the Vulnerability Assessment Process Flow Diagram

provided. Think about the functionality of the software application to identify which areas of security apply to Artemis Financial's web application. Document your findings in your vulnerability assessment report and justify why each area is relevant to the software application.

Please note: Not all seven areas of security in the Vulnerability Assessment Process Flow Diagram apply to the company's software application.

3. **Manual Review**: Refer to the seven security areas outlined in the Vulnerability Assessment Process Flow Diagram. Use what you've learned in steps 1 and 2 to guide your manual review. Identify all vulnerabilities in the Project One Code Base, linked in Supporting Materials, by manually inspecting the code. Document your findings in your vulnerability assessment report. Be sure to include a description that identifies where the vulnerabilities are found (specific class file, if applicable).
4. **Static Testing:** Integrate the dependency-check plug-in into Maven by following the instructions outlined in the Integrating the Maven Dependency-Check Plug-in tutorial provided in Supporting Materials. Run a dependency check on Artemis Financial's software application to identify all security vulnerabilities in the code. Specifically, identify all vulnerabilities in the code base by analyzing results from running the code through a static test. Include these items from the dependency-check report in your vulnerability assessment report:
    a. The names or vulnerability codes of the known vulnerabilities
    b. A brief description and recommended solutions that are found in the dependency-check report
    c. Attribution (if any) that documents how this vulnerability has been identified or how it was documented in the past
5. **Mitigation Plan:** Interpret the results from the manual review and static testing report. Identify steps to mitigate the identified security vulnerabilities by creating an action list that documents how to fix each vulnerability in your vulnerability assessment report.

    Please note: You do not need to fix these vulnerabilities in this project.

What to Submit

To complete this project, you must submit the following:

Vulnerability Assessment Report
Use the Project One Template to complete your vulnerability assessment report.

Supporting Materials

The following resources support your work on the project:

Diagram: Vulnerability Assessment Process Flow Diagram
Reference this process flow diagram during the project to determine which of the seven areas of security to assess for Artemis Financial's software application.

A text-only version is available: Vulnerability Assessment Process Flow Diagram Text-Only Version.

Code Base: Project One Code Base
Open a new Java project in Eclipse and upload this zipped file folder. This folder contains the code for the web application from Artemis Financial. It also contains security vulnerabilities for you to identify using the guidelines provided.

Website: Secure Coding Guidelines for Java SE
In this project, you must examine Java code for security vulnerabilities. This website gives you an up-to-date list of common Java security issues, examples of Java code to examine, and possible Java code solutions.

Website: OWASP Secure Coding Practices Quick Reference Guide
In this project, you must examine Java code for security vulnerabilities. The OWASP website gives you an up-to-date list of common web application security issues. The site also provides examples of using the dependency-check tool to test Java code for security issues with Java project library dependencies.

Textbook: *Iron-Clad Java: Building Secure Web Applications*, Chapters 1, 3, 7, and 10
Revisit these chapters. Chapter 1 gives you a basic security foundation in the context of web applications. Chapter 3 explains the benefits and common practices for access control. Chapter 7 explains the common attack of SQL injection. Chapter 10 gives you a foundation for security testing and security in the development process.

Tutorial: Integrating the Maven Dependency-Check Plug-in Tutorial

Follow the instructions in this tutorial to learn how to integrate the dependency-check plug-in into Maven. You'll need to edit the pom.xml file to add the dependency-check plug-in to Artemis Financial's software application. When you compile your code, Eclipse will run the Maven plug-in.

Tool: OWASP Dependency-Check Maven

OWASP Dependency-Check Maven is an open-source solution. It's used to scan Java applications to identify the use of known vulnerabilities. You'll use this to run a dependency check to statically test the code and output an HTML report.

Project One Rubric

Criteria	Exemplary (100%)	Proficient (85%)	Needs Improvement (55%)	Not Evident (0%)	Value
Interpreting Client Needs	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Determines client's needs and potential threats and attacks associated with their application and software security requirements	Shows progress toward proficiency, but with errors or omissions	Does not attempt criterion	15
Areas of Security	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Identifies which areas of security apply to Artemis Financial's web application, and justifies why each area is relevant to the software application	Shows progress toward proficiency, but with errors or omissions	Does not attempt criterion	15
Manual Review	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Identifies all vulnerabilities in the code base by manually inspecting the code	Shows progress toward proficiency, but with errors or omissions	Does not attempt criterion	20
Static Testing	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Identifies all vulnerabilities in the code base by analyzing results from running the code through a static test	Shows progress toward proficiency, but with errors or omissions	Does not attempt criterion	20
Mitigation Plan	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Identifies steps to mitigate the identified security vulnerabilities by creating an action list that documents how to fix each vulnerability	Shows progress toward proficiency, but with errors or omissions	Does not attempt criterion	25
Articulation of Response	Exceeds proficiency in an exceptionally clear, insightful, sophisticated, or creative manner	Clearly conveys meaning with correct grammar, sentence structure, and spelling, demonstrating an	Shows progress toward proficiency, but with errors in grammar, sentence structure, and spelling, negatively	Submission has critical errors in grammar, sentence structure, and spelling, preventing understanding of	5

		understanding of audience and purpose	impacting readability	ideas	
				Total:	100%