Unit 2: UML Modelling to Support Secure System Planning

Seminar 2

Question 1: Table

Task:

Create a 2-column multi-line table. In the left-hand column, include the software development stages of the Scrum agile life cycle approach to project management. In the right-hand column, describe the processes which you recommend are applied at each stage to ensure that secure software is produced at the end of the development. To support the preparation of your response, you can refer to the following literature:

Sharma, A. & Bawa, R. K. (2020) Identification and Integration of Security Activities for Secure Agile Development. International Journal of Information Technology.

Answer:

As an Agile framework, Scrum focuses on software delivery through iterative cycles and incremental progress, emphasizing adaptability and responsiveness to change. However, this approach may introduce security vulnerabilities due to its fast-paced nature and minimal documentation. To address these concerns, Sharma and Bawa (2020) suggest incorporating security engineering practices into each Scrum phase, aligning with established secure software development methods to reduce risks without sacrificing agility. The following outlines a systematic approach to embedding security measures throughout the Scrum process, ensuring the creation of secure software products.

Scrum Development Stage	Recommended Security Processes
Product Backlog Creation	Define Security Requirements: Identify critical security needs early and include them in the backlog (Sharma & Bawa, 2020). Threat Modeling: Recognize potential threats for each backlog item, integrating them as part of risk assessment (Sharma & Bawa, 2020).
Sprint Planning	Security Education and Role Matrix Setup: Train team members on secure coding practices, assign roles based on security impact, and prioritize backlog items with security concerns (Sharma & Bawa, 2020).
Sprint Execution	Code Analysis and Secure Design Principles: Apply secure design practices and use static code analysis tools to identify vulnerabilities in code during development (Sharma & Bawa, 2020). Automated Security Testing: Integrate security testing tools to detect vulnerabilities dynamically as new code is added, supporting both continuous integration and deployment goals (Sharma & Bawa, 2020).
Sprint Review	Security Requirement Verification: Confirm that security requirements are met in each feature presented for review (Sharma & Bawa, 2020). Penetration Testing: Perform penetration testing on completed features to ensure resilience to potential security breaches (Sharma & Bawa, 2020).
Sprint Retrospective	Review Security Incidents and Update Practices: Analyze any security issues or incidents that occurred, adjust security policies, and incorporate learning into future sprints (Sharma & Bawa, 2020). Continuous Improvement of Security Standards: Refine and adapt security processes based on team feedback and encountered threats (Sharma & Bawa, 2020).
Release	Final Security Review and Code Signing: Conduct a thorough security review, sign the code for integrity, and document security features (Sharma & Bawa, 2020). Operational Planning and Incident Response Preparation: Prepare operational documentation, establish incident response protocols, and improve repository practices for secure deployment (Sharma & Bawa, 2020).

The methods described align with Sharma and Bawa's suggestions, successfully integrating Scrum's flexible approach with a strong security framework.

References

Sharma, A. & Bawa, R. K. (2020) Identification and Integration of Security Activities for Secure Agile Development. International Journal of Information Technology.