# CS 405 Project Two Script Template

Complete this template by replacing the bracketed text with the relevant information.

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | Welcome, everyone, to the Green Pace Security Policy Presentation. I'm excited to share with you our comprehensive security policy, designed to enhance the security of our systems and protect our valuable assets.  I'm Laynie Tierney and I'll be guiding you through this presentation today. Our security policy is a vital component of our commitment to safeguarding our organization against evolving cyber threats.  So, let's dive in and explore the principles, standards, and best practices that form the foundation of our security strategy. Together, we'll ensure that Green Pace remains at the forefront of security excellence.  Let's get started! |
| **2** | I'm pleased to present Green Pace's new Security Policy. This policy was developed in response to the increasing complexity of cyber threats and the critical need to safeguard our systems and data. By implementing this policy, we aim to bolster our defense-in-depth strategy, which involves layering security measures to provide comprehensive protection against potential attacks. As illustrated in the diagram, this approach ensures that even if one layer of defense is breached, there are additional layers in place to mitigate the impact. |
| **3** | Now, let's take a closer look at our Threats Matrix. This matrix provides a comprehensive overview of the security risks we face. By categorizing these risks based on likelihood and priority, we can prioritize our efforts to address the most critical vulnerabilities first. Additionally, we'll explore how automation can be leveraged to detect and mitigate these coding vulnerabilities efficiently, enhancing our overall security posture. |
| **4** | Next, let's examine the foundational principles of our security policy. We've identified 10 key principles that guide our approach to security. Each principle is closely aligned with specific coding standards, ensuring consistency and adherence to best practices. By linking principles to standards, we establish a clear framework for addressing security concerns throughout the development lifecycle. |
| **5** | Now, let's discuss our prioritized coding standards. We've established 10 key standards that govern our development practices, prioritized based on their impact on security. Our prioritization system considers factors such as the surface area of attack and the assumption of vulnerability. By focusing on these critical standards, we can effectively mitigate potential security risks and ensure the integrity of our codebase. |
| **6** | Encryption is a cornerstone of our security strategy, particularly when it comes to protecting sensitive data. In our Encryption Strategy, we outline policies for encryption in flight, at rest, and in use. By encrypting data during transmission, storage, and processing, we mitigate the risk of unauthorized access and data breaches. This comprehensive approach to encryption ensures that our data remains secure, regardless of its state or location. |
| **7** | Triple-A Framework, a foundational concept in cybersecurity that encompasses authentication, authorization, and accounting. Let's delve into each component:  Authentication ensures that users are who they claim to be by verifying their identity through credentials such as usernames and passwords, biometrics, or multifactor authentication.  Authorization determines what actions users are allowed to perform within a system once they've been authenticated. It involves defining access levels, permissions, and privileges based on roles or user attributes.  Accounting involves tracking and logging user activities and system events for auditing, monitoring, and reporting purposes. It provides visibility into who accessed what, when, and from where, helping to detect and investigate security incidents.  By implementing robust policies and controls aligned with the Triple-A Framework, we can enhance the security and integrity of our systems, safeguard sensitive data, and ensure compliance with regulatory requirements. Let's proceed to explore how these policies are integrated into our security strategy |
| **8** | On this slide, we'll dive into unit testing to identify and prevent coding vulnerabilities. We'll focus on a specific vulnerability, such as input validation, and create a series of tests to check for both positive and negative scenarios. Let's proceed to the individual test slides to see how this works in practice. |
| **9** | This first test, titled 'Valid Input Test,' evaluates how the system handles valid input data. We provide input that meets the expected format and verify that the system processes it successfully. |
| **10** | The next test, titled 'SQL Injection Test,' assesses the system's resilience against SQL injection attacks. We attempt to inject SQL code into input fields and verify that the system detects and rejects the malicious input. |
| **11** | The 'Boundary Test' examines how the system handles input values at or near the boundaries of allowed ranges. We test extreme values to ensure the system behaves as expected without exhibiting unexpected behavior. |
| **12** | The 'Null Input Test' evaluates how the system handles null or empty input. We provide input with no values and verify that the system handles it gracefully without crashing or producing errors. |
| **13** | The 'Cross-Site Scripting (XSS) Test' assesses the system's ability to detect and sanitize malicious scripts injected into input fields, thereby preventing XSS attacks. Let's see how the system handles this security challenge. |
| **14 (image)** | Our pipeline incorporates security at every stage, from code development to deployment. During development, code analysis tools help identify vulnerabilities early. Automated security checks ensure only secure code is merged.  In the build phase, static application security testing (SAST) tools scan for vulnerabilities, and in testing, dynamic application security testing (DAST) tools simulate attacks.  During deployment, configuration management tools provision secure infrastructure. This holistic approach ensures security is not an afterthought but a priority throughout our development process.  By embracing automation, we detect and mitigate security risks early, delivering robust and secure software to our customers. |
| **15** | Showcasing how security is integrated throughout our software development lifecycle. We utilize external security tools strategically placed at various stages to automate checks and ensure continuous protection against threats. For example, static code analysis tools help identify vulnerabilities during code development, while vulnerability scanners analyze dependencies for known issues. By incorporating these tools, we strengthen our security posture and mitigate risks effectively. |
| **16** | We explore the risks and benefits associated with our security strategy. Delaying the implementation of security measures exposes our systems to vulnerabilities and cyber threats. However, by acting now and implementing proactive security measures, such as regular security assessments and audits, we can mitigate existing vulnerabilities and strengthen our defenses. Proactive security measures not only protect our organization from financial and reputational damage but also enhance customer trust and confidence. It's imperative that we prioritize security to safeguard our systems and data against potential threats. |
| **17** | Our gap analysis has revealed areas for improvement in our security policy. These include the need for a comprehensive incident response plan, an enhanced employee security training program, and better integration of security into our software development lifecycle. By addressing these gaps, we can strengthen our overall security posture and better protect our systems and data from potential threats. |
| **18** | To prevent future problems and mitigate security risks, we recommend adopting specific standards. These include the implementation of a robust incident response plan, enhancement of our employee security training program, and better integration of security into our software development lifecycle. By implementing these measures, we can enhance our security posture and better protect our organization against evolving threats. |
| **19** | References Listed |