# CS 405 Project Two Script Template

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

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | Hello, my name is Jesse Kniss and I am a developer at Green Pace. Today, we are going to discuss the Green Pace security policy. This policy will discuss expectations for producing secure applications, current implementation guidelines, and recommendations for future maintenance. |
| **2** | As technology has evolved, cyber threats and attacks become more prevalent. Unwanted actors are looking to benefit from vulnerable business systems each year. As attacks increase each year, 53% of cyber-attacks result in damages of $500,000 or more.  Part of the Green Pace’s efforts to combat these attacks is implementing a defense-in-depth strategy. DiD is an approach to cyber security that implements the utilization of multiple layers of security, to protect company assets.  As you look at the chart, you can see the multiple layers and various defense techniques that can be used to create this strategy which we will discuss throughout the presentation. |
| **3** | The threat matrix is a risk assessment tool that provides developers information on the severity of leaving vulnerabilities unchecked. Each rule and recommendation is assigned a priority value that is calculated by taking the severity, likelihood, and remediation cost of each rule and assigned a level based on that analysis.  Green Pace will use this matrix to prioritize repairs of rule violations. Green Pace will also utilize automated detection analysis to check for coding guideline violations for all builds. Tools such as the Secure Coding Validation Suite, SonarQube, and Raxis will be required for all development tasks. |
| **4** | This is a list of the 10 principles and which coding standards apply to each.  Each coding standard is comprised of rules and recommendations, to provide guidance on secure coding practices. Each principle aligns to a minimum one standard but can include additional principles depending on the development environment.  The principles and standards laid out in this policy, shall be followed at all times. |
| **5** | Standards are listed in order of threat level, with high priority threats towards the top. Each standard listed in the Green Pace policy must follow compliance standards to ensure vulnerabilities are not left unchecked. |
| **6** | Encryption is key component of a good defense in depth strategy. At Green Pace, encryption methods shall be decided based on the categories shown.  Encryption at rest involves the protection of data that is not actively moving. This data could be found on hard disk drives, databases, flash memory, or other steady state devices. Storage locations must be encrypted, with access controls and network security included.  Encryption at flight deals with any data that is being transferred from network to network, through email communications, local to cloud storage transfer, and web browsing. Any data in flight will use secure encrypted connections such as SSL, TLS and HTTPS to transfer data along with firewall, remote access security protocols.  Data that is currently being created, viewed, edited, or used will follow encryption in use policies. Data will always be encrypted, regardless of location. Encryption will use SHA-256, access to data will be reviewed by security teams and issues on an as-needed basis.  Encryption of data benefits all encryption polices, by creating a layered, secure approach. If data is compromised, encryption methods can prevent unauthorized users from total access, and potentially reduce damage from a security breach. |
| **7** | Green Pace will employ authentication/authorization requirements for all users. Employees will be required to login with an approved username/password. Employees will also be required to set up a pin for multi-factor authentication (2FA). Employees will be authorized access to content based on job requirements.  Security teams will perform accounting on resources and user activities. Methods such as log files, auditing user access, and monitoring activity on resource usage will ensure all threats are monitored. |
| **8** | Unit testing validates individual components and are required by developers for any code created. Green Pace developers will use only approved testing environments to create unit tests. A proper unit test will check one method only, with as much coverage as possible. The following slides represent examples proper unit tests. |
| **9** | When using googletest, developers will create assertions. Assertions check whether a condition is true and are used to verify the intended behavior of the code. If the assertion crashes, the test is considered failed. This assertion shown is checking that the vector is initialized as empty when a new instance of the collection object is created. |
| **10** | Here we have another assertion statement, this time checking that no errors are thrown when the vector is modified with a resize operation. Googletest can check for a particular error, or any error using an assert statement. |
| **11** | Developers will also use negative tests to check code. A negative test is an assertion that should throw an error, or failure. Negative testing allows developers to check that the program catches errors when they arise, and that the program can terminate gracefully if errors occur. This test case is looking for an out-of-range error to be thrown when an index outside of the vector bounds is called. |
| **12** | The previous slides all use an assert method, that create a fatal failure when testing fail. Assert methods should be used when developers expect the failure should terminate the process. If the failure should notify but continue processing, developers should use an expect method like the one shown. This method expects the vector to equal a size of 25. |
| **13** | Automation will be used for the enforcement of and compliance to the standards defined in this policy. Green Pace will use automated tools throughout the development process. Automated tools allow development teams to focus on meeting business needs, while ensuring quality code, security applications, and regulations are met. |
| **14** | This slide shows a variety of tools for each stage of development. Developers are not required to use each tool show, but this guide should be used as reference when deciding how to build a DevSecOps pipeline for your code requirements. |
| **15** | So what happens if these polices are not followed? Green Pace can become a target for cyber-attacks. If successful, these attacks could lead to loss of valuable data, financial losses and fines, damage to our reputation, loss of consumer trust, and in some cases even loss of life. |
| **16** | To mitigate risks, Green Pace employees will follow all documented policies and procedures. Green Pace also recognizes that more efforts are required to create a robust, secure system.  A recent survey concluded that only 30% of organizations fully implement DevSecOps in development. The primary reason for slow DevSecOps integration in organizations is a lack of internal guidance, awareness of vulnerabilities and overall guidance on how systems should be implemented. For these reasons, employees will be provided training on DevSecOps processes and tools. Green Pace will also enlist help from third-party security firms for routine security reviews. Documents and polies will also be audited regularly for changing security threats. |
| **17** | Security is the responsibility of everyone in DevSecOps. Green Pace will adopt a shift left security mindest, and DevSecOps practices. When implemented correctly, DevSecOps allows for a reduction in expenses, increased delivery rates to customers, and faster speed of recovery in case of a security incident. These polices are in place to protect Green Pace and ensure that security is at the forefront of development. |

# Works Cited

Bhandari, P. (2022, September 16). *DevSecOps Pipeline - A Complete Overview*. Retrieved from Xenonstack: https://www.xenonstack.com/insights/guide-devsecops-pipeline

Cisco. (2022, October 10). *What is a Cyberattack?* Retrieved from Cisco Secuirty: https://www.cisco.com/c/en/us/products/security/common-cyberattacks.html

Doerrfeld, B. (2021 , December 22). *Only 30% of Orgs Fully Implement DevSecOps*. Retrieved from DevOps: https://devops.com/only-30-of-orgs-fully-implement-devsecops/

Francis, W. J. (2022, August 3). *Pulling security to the left: How to think about security before writing code*. Retrieved from Tech Republic: https://www.techrepublic.com/article/pulling-security-left/

Froehlich, A. (2022, June n.d). *8 Benefits of DevSecOps automation*. Retrieved from TechTarget: https://www.techtarget.com/searchsecurity/tip/8-benefits-of-DevSecOps-automation

Google. (2022, October n.d). *GoogletTest User Guide*. Retrieved from Google Test: http://google.github.io/googletest/

Henriquez, M. (2019, October 29). *Don't Shift Left, Start Left: Why Developers Should Be the First Line Of Defense*. Retrieved from Security: https://www.securitymagazine.com/articles/91181-dont-shift-left-start-left--why-developers-should-be-the-first-line-of-defense-in-enterprises

Jeganathan, S. (2019, November 1). *DevSecOps: A Systemic Approach for Secure Software Development*. Retrieved from ISSA Journal: https://eds-a-ebscohost-com.ezproxy.snhu.edu/eds/pdfviewer/pdfviewer?vid=1&sid=16071980-1fe5-4425-81a4-7f402b4a552b%40sdc-v-sessmgr01

Mohanakrishana, R. (2021, December 7). *Top 10 Threat Modeling Tools in 2021*. Retrieved from Spiceworks: https://www.spiceworks.com/it-security/vulnerability-management/articles/top-threat-modeling-tools/

SEI. (2020, May 29). *CERT C++ Coding Standard*. Retrieved from Confulence: https://wiki.sei.cmu.edu/confluence/pages/viewpage.action?pageId=88046682

Software Testing Help. (2022, September 24). *Top 40 Static Code Analysis Tools*. Retrieved from Software Testing Help: https://www.softwaretestinghelp.com/tools/top-40-static-code-analysis-tools/