# CS 405 Project Two Script Template

Video Link: <https://youtu.be/T3mCeG3YwbQ>

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

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | Green Pace Security Policy Presentation |
| **2** | Defense in Depth refers to making sure that security is a priority and has multiple layers to ensure the best possible protection for the company.  The following chart shows a defense in depth illustration with the multiple layers that compose it. |
| **3** | The following are the major code standards rankings accorinding to how likely it will happen. I will go over the standards themselves in a later slide  But I want to point out the priority square. All security should be a priority in the digital age. |
| **4** | The following are the 10 Principles and noted are standards that are associated with each   1. Validate Input Data: All data should be validated to prevent software vulneabilites. 2. Heed Compiler Warnings – The Highest Compiler Warning levels shall be used to compile code. 3. Architect and Design for Security Policies - Design and build software that enforces and heeds security policies. 4. Keep it Simple – Code design shall be kept as small and simple as possible 5. Default Deny – Access decisions shall be based on permissions 6. Adhere to the Principle of least Privilege – Each process shall execute with the least amount of privilege needed to perform that function. 7. Sanitize Data Sent to Other Systems – Data Shall be sanitized Prior to passing it along to another system. 8. Practice Defense in Depth – Systems shall be protected in Layers. 9. Use Effective Quality Assurance Techniques – Vulnerabilites can be identified and eliminated using good quality assurance techniques. 10. Adopt a Secure Coding Strandard – A secure coding standard shall be developed for each individual programming language and platform. |
| **5** | I Have listed the following in order of mostly likely to occur in development   1. Expressions: Not properly dealing with expressions can cause all sorts of havoc during development 2. Integers: Not watching for intergers going out of bounds can cause untold issues as the program reachs and grabs items out of memory not in the bounds of the variable 3. Containers – making sure the containers on objects are used properly can help ensure key data structures don’t break overtime or have unknown bugs to them 4. Characters and Strings – Make sure to follow best practices when it comes to reading in and storing strings. Not following this can lead to no null terminated characters which can cause major issues. 5. Object Oriented Programming – make sure to follow best practice in regards to building and using classes 6. Concurrency The standard and rules deal with using multithreading the correct way. Such as proper way to close the threads. 7. Memory Management – this deals with allocating and deallocating memory the correct way such as not mixing up c style allocation and c++ style allocation in the same program. 8. Declarations and Initialization Rules – Make sure to follow best practices when declaring variables and functions to ensure readable and functional code 9. Input Output – Rules handing opening and closing output and input streams 10. Exceptions and Error handling – ideals on how to hand exceptions and error handling in the programs as well as making effective unit test. |
| **6** | * In Rest – Protects the data where it is stored. This could be on a computer, phone, database, or cloud. Uses tools such as VeraCrypt or AxCrypt. Keeps data from being physically, logically, or breached any other way * At Flight – Protects data as it’s moved from one location to another. This would be things like your emails. This would also include protective networks on both offices and remote workers ( using vpns and such). * In Use – Protects data as it’s being accessed or created. This is the in between state of the other two policies. There are a variety of system tools and operating system tools to implement this. |
| **7** | Authentication – Authentication verifies who a user is. This can take the form of passwords, SSO systems, biometrics, etc   * Authorization – Authorization is a system of permissions to access data. It’s important that we only allow as little permission as possible to accounts so that data is protected. * Accounting - This would be the logging of interactions with our systems and our data. |
| **8** | The following are Unit test that test different properties of a collections object. This will show a variety of examples on how unit test work and why they are effective. |
| **9** | * The test here verifies that a value can be added to an empty collection.   This could be added on by adding different entries, or values  This is done using ASSERT commands |
| **10** | * Building off the last test this does exactly that by adding 5 entries. * This could be further tested by adding multiples other than 5. |
| **11** | * This tests that the capacity is greater than or equal to the size. Unit test can be used in a variety of ways. * A way to take this further would be to test for paritial values. |
| **12** | * This test verifies that the size property of the collection is resizable. * Another test that could be performed is the ability to shrink the collection. |
| **13** | * The following shows a negative test. It’s testing to make sure that exceptions are working and picking up errors.   This is done via EXPECT statements |
| **14** | * + The DevSecOps pipeline is integrating security measures into each step of the DevOps toolchain. This will allow security to become an important and key part in an ever growing technological world. This will also make sure that the company is as protected as possible from outside threats.   + There are a variety of tools that can be used to help with providing security coverage. Google Testing framework for unit testing c++ applications for instance. CPPChecker for static code analysis. These are just a few of the software tools for development, but there are many more protocols and hardware based security tools that can help ensure that company data is kept private. |
| **15** | I recommend a few places that we can start with security.   * + 1. Check all complier errors in code and run the code through a static code analysis program. This will allow us to see what the risk in the code are and what we can do to fix this.   + 2. Make sure that we are keeping up with critical security patches. These are very important as soon as a day one vulnerability is announced hackers will know how to attack your systems.   + 3. Enact account authorization and authentication. That way we can make sure one persons account being hacked doesn’t compromise the entire system.   + 4. Security training for all employees. Let them know the dangers and how to be aware of common attacks ( such as email spoofing   + By starting with the 4 above we can start making an positive impact in your company. Overall we need to fully adopt a security first policy, but that we take time to fully implement. In the meantime insuring that we are making constant progress is important. |
| **16** | * Defense in Depth – This is a must, having multiple layers of coverage will help prevent attacks and mitigate slight attacks * DevSecOps – Moving to a DevSecOps model will help to ensure that security is a priority in all rollouts and development. * Authentication and Authorization – Making sure you validate who uses the systems and what can access are very important for protecting data * Tools – Make good use of third party tools to help supplement your own internal security ( ie don’t build your own encryption). * Priority – Make security a priority so that the business can continue and grow. |