# CS 405 Project Two Script Template

video link: https://youtu.be/KCFDOiEUMeg

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

| **Slide Number** | **Narrative** |
| --- | --- |
| **1** | In Green Pace’s best interest, I have developed a security policy presentation to go over what exactly the 10 principles are, what coding standards we should look out for while we develop, and why this is all important. Furthermore, we will go over encryption policies, triple-a policies, examples of unit testing, tools, and the risk and benefits to waiting or acting fast to any vulnerabilities. Finally, I will go over my recommendations once we go over all the slides. |
| **2** | Defense in depth is one of the strongest approaches to security implementation..  However, it is also one of the most expensive. By adding layers of security at multiple layers seen in the diagram, we are able to dissuade any attempts to take advantage of possible vulnerabilities. |
| **3** | In this slide, we can see 4 sections of the threat matrix: likely, priority, unlikely, and low priority. For likely, we have string correctness. For priority, we have SQL injection. For low priority, we have don't repeat yourself. Finally, for unlikely we have data values. |
| **4** | Our ten principles are as follows:   * *Validate Input Data - “Data Types”* * *Heed Compiler Warnings* - “String correctness” * *Architect and Design for Security Policies* - “Good formatting” * *Keep It Simple* - “String correctness” * *Default Deny* - “SQL injection” * *Adhere to the Principle of Least Privilege* - “SQL injection” * *Sanitize Data Sent to Other Systems* - “Memory Protection” * *Practice Defense in Depth* - “Assertions”, “Exceptions” * *Use Effective Quality Assurance Techniques* - “Good formatting” * *Adopt a Secure Coding Standard* - All of the above |
| **5** | With 10 being the most important to implement/maintain to 1 being the least critical to implement or maintain.   * Data types - 9 * Data values - 3 * String correctness - 4 * SQL injection - 10 * Memory Protection - 7 * Assertions - 6 * Exceptions - 5 * Immutable Objects - 2 * Don’t Repeat Yourself - 1 * Good formatting - 8   In terms of vulnerabilities, I choose those that enable the most types of attacks at lower levels of importance. Repeating ourselves in our code can lead to vulnerabilities in our code out of redundancies that may go ignored due to us repeating ourselves in our code. However, it is not quite as important as data type errors or setting protection in case of sql injection attempts. |
| **6** | First, encryption at rest involves encrypting data in physical storage devices such as hard drives or databases, or similar devices such as computers and phones. Encryption of this data can be done through a multitude of encryption tools or encryption on the device itself.  Secondly, in flight data is data that is moving from one device or database to another. To encrypt this sort of data, the best approaches are through implementing encryption features and tools into the network or servers themselves, email encryption, or authorization and authentication features.  Lastly, encryption for data in use is what is used for programs that are constantly running such as any software or tools. Encryption tools and user authentication are two of the best approaches for maintaining data encryption for data in us |
| **7** | Authentication is the process of ensuring that the user is someone who should have access to the data they are trying to access. This is one of many ways of applying defense and is one of the less expensive approaches requiring only a database and server if online for user authentication. This is done at the time that the user logins and can be done when they change databases as well.  Authorization is the process of granting the correct privileges to a user. To access sensitive data, authorizing after authenticating is one of the best approaches to adding layers of security. While they may have access to one database, authorization can prevent a user from viewing another database if they are not supposed to see it. This allows us to set levels of access to the user database and make for a more secure workplace. Whenever a new user is added, authorization levels by default should be low in comparison to someone who has already been granted higher levels of authorization.  Accounting is the process of monitoring any user in a server or database regardless of authentication or authorization. This is an additional layer on top of the two previously mentioned to be able to see where any issues come from and stop malicious attempts before they happen. |
| **8** | These next few examples show unit tests. Two positive and then two negative. |
| **9** | This slide shows the automation summary that goes over the development cycle of a DevSecOps developer and how security is implemented into the cycle itself |
| **10** | DevSecOps is DevOps with security integrated into the design and maintenance philosophy. As such, this means that while developing any code, a DevSecOps developer must keep in mind what vulnerabilities are easily exploitable so as to develop the code in a way that is prepared to have defense against such attacks.  Automated tools such as log collection, event alerting, and intrusion detection are great ways of further preventing any sort of exploit. |
| **11** | While there are many risk to both acting fast and waiting, each have their own benefits. While some issues may only become worse over time, waiting allows the developer to think of stronger solutions that will lead to less problems in the long run. However, waiting too long can lead to either allowing sensitive data to be accessible through exploits which cannot be undone. |
| **12** | The biggest gap in the security policy is that it lacks an outline on how to respond to any potential ongoing attack. While automated systems and tools are great for stopping any vulnerabilities, they do not always catch everything. As such, I believe that an outline on how to handle ongoing exploits is the best step to making the security policy as strong as possible. |
| **13** | Overall, I believe every single one of the coding standards should be kept in mind while developing any code here at Green Pace. While they are not all as important in terms of priority or all as likely as each other, to be a great secure developer keeping all of the security policies and their examples as guidelines on possibilities of vulnerabilities will lead to the most security. |
| **14** | In this last slide, we have my references. Thank you for your time. |