[Entity] Information Technology Standard	No:
IT Standard: **Secure Coding**	Updated:
	Issued By: Owner:

1.0 Purpose and Benefits

Government organizations are under constant cyber-attacks that attempt to exploit vulnerabilities within computer systems and thereby threaten the confidentiality, integrity, and availability of information. A large number of vulnerabilities that are successfully exploited are due to software coding weaknesses and coding implementation flaws.

The objective of this coding standard is to ensure that code written is resilient to high-risk threats and to avoid the occurrence of the most common coding errors which create serious vulnerabilities in software. While it is impossible to write code that is completely impervious to all possible attacks, implementing these coding standards throughout information systems will significantly reduce the risk of disclosure, alteration or destruction of information due to software vulnerabilities.

2.0 Authority

[Entity Authority]

3.0 Scope

[Entity Scope]

4.0 Information Statement

As per the Information Security Policy, all software written for or deployed on systems must incorporate secure coding practices, to avoid the occurrence of common coding vulnerabilities and to be resilient to high-risk threats, before being deployed in production.

The items enumerated in this standard are not an exhaustive list of high-risk attacks and common coding errors but rather a list of the most damaging and pervasive.

Therefore, code written must contain mitigating controls not only for the items specifically articulated in the standard below, but also for any medium and high-risk threats that are identified during a system's life cycle.

High risk threats include, but are not limited to:

1. Code Injection
2. Cross-site scripting (XSS)
3. Cross-site request forgery (CSRF)
4. Information leakage and improper error handling
5. Missing Authentication for Critical Function
6. Missing Encryption of Sensitive Data
7. URL Redirection to Untrusted Site ('Open Redirect')

At a minimum, code must eliminate or mitigate the threats identified in the current version of the Open Web Application Security Project (OWASP) Top 10 Most Critical Application Security Risks ('OWASP Top 10') and the Common Weakness Enumeration (CWE)/SANS Top 25 Most Dangerous Software Errors ('CWE/SANS Top 25') publications (see Appendix A).

Both OWASP and CWE/SANS periodically reissue their respective lists based on changes in vulnerability and exploitation patterns. Developers are required to independently remain aware of updates to these lists and incorporate any new recommendations.

Use of common security control libraries and common API's, that have undergone security testing, is required to ensure a consistent approach that minimizes defects and prevents exploitation. When available, publicly available or vendor-supplied libraries or APIs should be used unless there's a business case developed and exception granted by the Information Security Officer (ISO)/designated security representative to develop a custom library.

To prevent defects or detect and remove them early, thereby realizing significant cost and schedule benefits to the entity, code must be checked for errors throughout development and during maintenance.

Entities must verify that the software assurance model used by the vendor is in line with this standard through vendor assurances, security testing and/or contract requirements.

5.0 Compliance

This standard shall take effect upon publication. Compliance is expected with all enterprise policies and standards. Policies and standards may be amended at any time.

If compliance with this standard is not feasible or technically possible, or if deviation from this policy is necessary to support a business function, entities shall request an exception through the Chief Information Security Officer's exception process.

6.0 Definitions of Key Terms

Term	Definition

7.0 Contact Information

Submit all inquiries and requests for future enhancements to the policy owner at:
[Entity Address]

8.0 Revision History

This standard shall be subject to periodic review to ensure relevance.

Date	**Description of Change**	**Reviewer**

9.0 Related Documents

Open Web Application Security Project (OWASP) Top 10 Most Critical Application Security Risks ('OWASP Top 10')

Open Web Application Security Project (OWASP) Developer Cheat Sheets

Open Web Application Security Project (OWASP) Enterprise Security API

Common Weakness Enumeration (CWE)/SANS Top 25 Most Dangerous Software Errors 'CWE/SANS Top 25')

Common Weakness Enumeration (CWE) List

Carnegie Mellon Software Engineering Institute CERT Secure Coding Standards

Appendix A: Coding Resources

Open Web Application Security Project (OWASP)

The OWASP Top 10 is authored by OWASP, an open-source application security community project which aims to raise security awareness of web application security risks. Although OWASP is focused on web application security, the standards and controls presented by this organization are generally also applicable to non-web based information systems.

In addition to the "Top 10" list, OWASP also produces the Enterprise Security API (ESAPI) library and developer cheat sheets. The ESAPI library is an open source, web application security control library designed to mitigate risks to web applications. The ESAPI library provides a framework to implement code to address the risks listed within the OWASP Top Ten project. The cheat sheets provide a concise collection of high value information on specific web application security topics.

Additional information regarding OWASP, the ESAPI library and the Top Ten project is available at https://www.owasp.org/.

Common Weakness Enumeration/SANS

The CWE/SANS Top 25 Most Dangerous Software Errors publication is the result of collaboration between the SANS Institute, MITRE, and many top software security experts in the US and Europe. The publication is a list of the most widespread and critical errors that can lead to serious vulnerabilities in software. They are often easy to find, and easy to exploit. They are dangerous because they will frequently allow attackers to completely take over the software, steal data, or prevent the software from working at all.

The MITRE website provides detailed guidance to software programmers for mitigating and avoiding each of the common weaknesses enumerated within the Top 25 list with the Common Weakness Enumeration (CWE) List.