Secure Software Development Lifecycle (SSDLC) Policy

This policy mandates the integration of security activities and controls into every phase of the **Software Development Lifecycle (SDLC)** to minimize vulnerabilities, ensure compliance, and protect the organization's applications and data.

1. Policy Overview and Scope

This policy applies to all software developed, purchased, or significantly customized by the organization, including web applications, mobile applications, APIs, and internal tools.

Key Principles:

- **Security by Design:** Security is a core requirement, not an add-on.
- **Shift Left:** Security activities must be moved to the earliest possible phase of the SDLC.
- **Continuous Improvement:** Security practices and tools are reviewed and updated regularly.

2. SSDLC Phases and Mandatory Security Gates

The following security activities are **mandatory** within each phase of the SDLC. No project may move to the next phase without meeting the security gate requirements for the current phase.

2.1. Planning & Requirements (Define)

Security Activity	Description	Security Gate Requirement
Asset Classification	Classify the application and the data it will process (e.g., Confidential, Internal, Public).	Formal **Data Classification Assessment** completed and approved by the CISO or delegate.

Security Activity	Description	Security Gate Requirement
Security Requirements	Define specific, measurable security requirements based on the classification and regulatory mandates (e.g., "All authentication must use MFA," "All PII must be encrypted at rest").	Security requirements document signed off by the **Product Owner and Security Architect**.
Risk Assessment	Perform an initial high-level risk assessment to identify key threats and compliance obligations.	Documentation of high-level risks and proposed mitigation strategies.

2.2. Design & Architecture (Design)

Security Activity	Description	Security Gate Requirement
Threat Modeling	A structured process to identify, categorize, and mitigate threats to the application and its components (e.g., *STRIDE* methodology).	Formal **Threat Model** documentation, including identified threats and design-level countermeasures.
Security Architecture Review	Review application architecture, technology stack, and third-party components for security flaws and adherence to security standards.	**Security Architecture Review sign-off** by the Security Architect.
Secure Design Patterns	Utilize pre-approved, security-hardened design patterns (e.g., input validation modules, secure logging services).	Proof that documented **Secure Design Patterns** have been integrated into the architecture.

2.3. Implementation & Coding (Develop)

Security Activity	Description	Security Gate Requirement
Secure Coding Standards	All developers must adhere to established secure coding guidelines (e.g., OWASP Top 10 prevention guidelines).	Developer training on secure coding completed within the last year.
Static Application Security Testing (SAST)	Run automated tools against the source code to identify security vulnerabilities *without* executing the application.	**SAST scan executed on every build**, with all High and Critical findings remediated or formally accepted as a risk.
Secrets Management	All credentials, keys, and tokens must be stored and accessed via an **approved secrets management platform**. Hardcoding secrets is **strictly prohibited**.	Automated check verifies that no secrets are present in code repositories.

2.4. Testing & Quality Assurance (Test)

Security Activity	Description	Security Gate Requirement
Dynamic Application Security Testing (DAST)	Run automated tools against the running application to simulate external attacks.	**DAST scan executed** with all High and Critical findings remediated.
Manual Penetration Testing (Pen Test)	Independent security experts perform manual testing for business logic flaws and complex vulnerabilities.	**Penetration Test Report** completed, with all High and Critical findings resolved or deferred with CISO approval. **(Mandatory for all production releases)**.
Third-Party Component Analysis	Use Software Composition Analysis (SCA) tools to identify vulnerabilities in open-source and third-party libraries.	**SCA scan** shows no high or critical known vulnerabilities in dependencies.

2.5. Deployment & Operations (Deploy)

Security Activity	Description	Security Gate Requirement
Secure Configuration Review	Review all production environment configuration (firewalls, cloud settings, network segregation) before deployment.	**Production readiness checklist** with security section approved by Security Operations.
Logging and Monitoring	Ensure security events, access attempts, and key application transactions are **logged and monitored** by the Security Operations Center (SOC).	Defined logging and alerting rules deployed to the production environment.
Secure Decommissioning	For retiring applications, ensure data is securely **archived or destroyed** according to the data retention policy.	Documentation of the secure data archival/destruction process.

3. Tooling and Automation

The Security Team will provide and manage the following mandatory tooling to ensure policy adherence:

- **SAST/DAST Tools:** Integrated into the continuous integration/continuous deployment (CI/CD) pipeline.
- **Secrets Manager:** Centralized vault for storing and retrieving sensitive credentials.
- **SCA Tool:** Integrated into the build process to check dependencies for known vulnerabilities.
- **Centralized Logging:** System to aggregate and analyze all application and infrastructure security logs.