Secure Software Development Lifecycle (SDLC) Policy and Procedures

1. Purpose

The purpose of this document is to outline policies, procedures, and controls for integrating security into every phase of the Software Development Lifecycle (SDLC) at DocuPanda. This ensures that all applications are developed with security as a priority, reducing vulnerabilities and risks.

---

2. Scope

This policy applies to all software development activities at DocuPanda, including new projects, enhancements, and maintenance of existing applications. It covers all personnel involved in the SDLC, including developers, QA engineers, product managers, and system administrators.

---

3. Policy Overview

3.1 Security Integration

Security must be an integral part of each phase of the SDLC:

1. **Planning**
   - Security requirements must be identified alongside functional requirements.
   - Risk assessments and threat modeling must be conducted.
2. **Design**
   - Adhere to secure design principles (e.g., least privilege, defense in depth).
   - Use design review processes to identify potential security issues.
3. **Development**
   - Follow secure coding standards (e.g., OWASP Top Ten).
   - Use tools for static code analysis to detect vulnerabilities.
4. **Testing**

    - Conduct security testing, including vulnerability assessments and penetration testing.
    - Use dynamic analysis tools to identify runtime vulnerabilities.

5. **Deployment**
    - Ensure secure configuration of production environments.
    - Implement runtime security controls and monitoring.

6. **Maintenance**
    - Monitor for vulnerabilities and apply patches promptly.
    - Perform periodic security reviews and audits.

---

4. Secure SDLC Procedures

4.1 Planning Phase

- **Risk Assessment and Threat Modeling**
  Identify potential security risks and define mitigation strategies. Use tools like STRIDE or DREAD for threat modeling.

- **Security Requirements Definition**
  Define security requirements alongside functional requirements. These include data protection, authentication, and access control mechanisms.

4.2 Design Phase

- **Secure Architecture Review**
  Conduct architecture reviews to ensure the system design meets security standards.

- **Data Flow Analysis**
  Ensure secure handling of sensitive data, including data encryption and secure data transmission.

4.3 Development Phase

- **Secure Coding Practices**
  - Follow industry best practices (e.g., OWASP, CERT).
  - Avoid known vulnerabilities, such as SQL injection, XSS, etc.
- **Static Application Security Testing (SAST)**
  Use SAST tools (e.g., SonarQube) to analyze source code for vulnerabilities.
- **Dependency Management**
  Regularly scan and update third-party libraries and dependencies to avoid using vulnerable versions.

4.4 Testing Phase

- **Dynamic Application Security Testing (DAST)**
  Perform DAST on running applications to identify vulnerabilities in runtime.
- **Penetration Testing**
  Conduct regular penetration tests to simulate real-world attacks and assess system robustness.
- **Automated Security Testing**
  Integrate automated security tests into CI/CD pipelines to identify and fix vulnerabilities early.

4.5 Deployment Phase

- **Configuration Management**
  Verify that production environments are securely configured (e.g., secure credentials, least privilege access).
- **Pre-Deployment Security Checks**
  Ensure that all security tests are passed before deployment to production.

4.6 Maintenance Phase

- **Vulnerability Management**
  Regularly monitor for new vulnerabilities using vulnerability scanners and threat intelligence feeds.
- **Patch Management**
  Apply security patches in a timely manner to all environments.

- **Incident Response**
  Have an incident response plan in place to quickly address and mitigate security incidents.

---

5. Roles and Responsibilities

- **Developers**
  Responsible for adhering to secure coding practices and fixing vulnerabilities identified during the development phase.
- **QA Engineers**
  Perform security testing and report vulnerabilities to developers for remediation.
- **System Administrators**
  Ensure secure configuration and maintenance of production environments.
- **Security Team**
  Provide guidance on security best practices, conduct audits, and lead incident response efforts.

---

6. Training and Awareness

- All employees involved in the SDLC must undergo regular training on secure coding practices, vulnerability management, and emerging threats.
- Training will include real-world examples of security breaches and their impact.

---

7. Monitoring and Auditing

- **Code Review**
  All code must undergo peer reviews focusing on security concerns.
- **Security Audits**
  Regular security audits will be conducted on all applications and environments.
- **Monitoring**
  Implement security monitoring tools (e.g., intrusion detection systems) to detect and respond to threats in real time.

8. Compliance and Enforcement

- **Compliance**
  Adhere to relevant security standards and regulations, such as GDPR, CCPA, and ISO 27001.
- **Enforcement**
  Non-compliance with secure SDLC policies will result in disciplinary action, up to and including termination for employees.

9. Policy Review

This policy will be reviewed annually or as necessary to adapt to changing security requirements and technological advancements.

Approved By: Uri Merhav (CIO)
Effective Date: 10/01/2024 **Next Review Date:** 01/01/2025