Secure Development Policy Template

Document Control Information

- **Document Title:** Secure Development Policy
- **Document Version:** 1.0
- **Last Updated:** [DATE]
- **Document Owner:** [ROLE/NAME]
- **Approved By:** [ROLE/NAME]
- **Next Review Date:** [DATE]

1. Introduction

1.1 Purpose

This Secure Development Policy establishes [ORGANIZATION NAME]'s requirements for incorporating security throughout the software development lifecycle in accordance with ISO 27001:2022 requirements. It provides a framework for ensuring that applications and systems are designed, developed, and maintained with security as a fundamental consideration.

1.2 Scope

This policy applies to: - All software development activities conducted by or on behalf of [ORGANIZATION NAME] - All types of software, including internal applications, commercial products, web applications, mobile applications, and APIs - All development methodologies, including waterfall, agile, and DevOps - All employees, contractors, consultants, and third parties involved in software development - All environments, including development, testing, staging, and production

1.3 Policy Statement

[ORGANIZATION NAME] is committed to: - Integrating security throughout the software development lifecycle - Following secure coding practices and standards - Conducting regular security testing and code reviews - Addressing security vulnerabilities in a timely manner - Providing secure development training to development teams - Continuously improving the security of our development processes

2. Secure Development Lifecycle

2.1 Security Requirements

- Security requirements shall be defined at the beginning of each project
- Security requirements shall be based on:
  - Risk assessment results
  - Compliance requirements
  - Industry standards and best practices
  - Threat modeling
- Security requirements shall be documented and tracked
- Security requirements shall be reviewed and approved by the security team
- Changes to security requirements shall follow change management procedures

2.2 Secure Design

- Security shall be considered during the design phase
- Threat modeling shall be performed for new applications and major changes
- Security design reviews shall be conducted
- Secure design patterns shall be used
- Defense-in-depth principles shall be applied
- Least privilege principles shall be incorporated
- Input validation and output encoding shall be designed into applications
- Authentication and authorization mechanisms shall be properly designed
- Session management shall be securely designed
- Error handling shall be designed to avoid information disclosure

2.3 Secure Coding

- Secure coding standards shall be established and followed
- Developers shall be trained in secure coding practices
- Code shall be written to prevent common vulnerabilities, including:
  - Injection flaws (SQL, NoSQL, OS command, etc.)
  - Cross-site scripting (XSS)
  - Broken authentication and session management
  - Cross-site request forgery (CSRF)
  - Security misconfigurations
  - Sensitive data exposure
  - Insufficient access control
  - XML external entities (XXE)
  - Insecure deserialization
  - Using components with known vulnerabilities
  - Insufficient logging and monitoring

- Security-related comments shall be removed from production code
- Debugging code shall be removed from production releases

2.4 Security Testing

- Security testing shall be integrated into the development process
- Automated security testing tools shall be used where appropriate
- The following security testing shall be performed:
  - Static application security testing (SAST)
  - Dynamic application security testing (DAST)
  - Interactive application security testing (IAST) where applicable
  - Software composition analysis (SCA)
  - Penetration testing for critical applications
- Security testing shall be performed before deployment to production
- Security testing results shall be documented and tracked
- Critical and high-risk vulnerabilities shall be addressed before deployment

2.5 Secure Deployment

- Deployment processes shall be documented and secure
- Deployment scripts shall be version controlled
- Production deployments shall be approved by authorized personnel
- Deployment credentials shall be secured and rotated regularly
- Deployment logs shall be maintained
- Rollback procedures shall be established and tested
- Production data shall not be used in non-production environments without anonymization

2.6 Security Verification and Validation

- Security requirements shall be verified before release
- Security acceptance criteria shall be defined and validated
- Final security review shall be conducted before major releases
- Security sign-off shall be required for production deployment
- Post-deployment security verification shall be performed

3. Development Environment Security

3.1 Environment Separation

- Development, testing, staging, and production environments shall be separated
- Access to environments shall be based on role and need

- Data flow between environments shall be controlled
- Production data shall not be used in non-production environments without anonymization
- Environment configurations shall be documented

3.2 Source Code Management

- Source code shall be stored in secure repositories
- Access to source code repositories shall be restricted
- Source code changes shall be tracked and attributed
- Code reviews shall be performed before merging
- Branching and merging strategies shall be defined
- Sensitive information shall not be stored in source code repositories

3.3 Development Tools and Technologies

- Development tools and technologies shall be approved before use
- Development tools shall be kept updated with security patches
- Development workstations shall be secured
- Integrated development environments (IDEs) shall be configured securely
- Development tools shall be scanned for vulnerabilities

4. Third-Party Code and Components

4.1 Third-Party Libraries and Components

- Third-party libraries and components shall be approved before use
- A software bill of materials (SBOM) shall be maintained
- Third-party components shall be obtained from trusted sources
- Third-party components shall be scanned for vulnerabilities
- Vulnerable components shall be updated or replaced
- Unnecessary features and components shall be disabled or removed

4.2 Open Source Software

- Open source software usage shall comply with license requirements
- Open source components shall be evaluated for security
- Open source vulnerabilities shall be monitored and addressed
- Open source usage shall be documented
- Legal review shall be conducted for open source licenses

4.3 Outsourced Development

- Security requirements shall be included in contracts

- Third-party developers shall follow this policy
- Code developed by third parties shall undergo security review
- Access to systems and data shall be controlled
- Third-party development activities shall be monitored

5. Secure Coding Standards

5.1 General Coding Standards

- Code shall be written to be maintainable and understandable
- Code complexity shall be minimized
- Code shall be properly commented and documented
- Consistent coding style shall be used
- Code shall be modular with clear separation of concerns

5.2 Input Validation

- All input shall be validated
- Input validation shall be performed on the server side
- Input validation shall include:
    - Data type validation
    - Range validation
    - Format validation
    - Length validation
    - White list validation where possible
- Input from untrusted sources shall be treated with special care

5.3 Output Encoding

- Output shall be encoded appropriate to the context
- HTML, JavaScript, CSS, and URL encoding shall be used where appropriate
- Character encoding shall be explicitly specified
- Content type headers shall be correctly set
- User-supplied data shall be encoded before inclusion in output

5.4 Authentication and Authorization

- Authentication mechanisms shall be secure
- Multi-factor authentication shall be supported for sensitive functions
- Passwords shall be stored using strong, salted hashing algorithms
- Session IDs shall be generated using secure random number generators
- Sessions shall timeout after periods of inactivity
- Authorization checks shall be performed at each request

- Principle of least privilege shall be applied

5.5 Data Protection

- Sensitive data shall be identified and protected
- Encryption shall be used for sensitive data storage
- Transport layer security shall be used for data transmission
- Sensitive data shall not be stored in logs or error messages
- Temporary files containing sensitive data shall be protected
- Data retention policies shall be implemented

5.6 Error Handling and Logging

- Errors shall be handled gracefully
- Error messages to users shall not reveal sensitive information
- Detailed error information shall be logged securely
- Exception handling shall not expose security vulnerabilities
- Security-relevant events shall be logged
- Logs shall be protected from unauthorized access and modification

6. Security Testing and Vulnerability Management

6.1 Security Testing Approaches

- Security testing shall be performed throughout the development lifecycle
- Automated security testing shall be integrated into the CI/CD pipeline
- Manual security testing shall complement automated testing
- Security testing shall cover all security requirements
- Security testing shall simulate real-world attack scenarios

6.2 Vulnerability Management

- A process shall be established for tracking and managing vulnerabilities
- Vulnerabilities shall be prioritized based on risk
- Timeframes for addressing vulnerabilities shall be defined:
  - Critical: [X] days
  - High: [Y] days
  - Medium: [Z] days
  - Low: Next release cycle
- Vulnerability remediation shall be verified
- Vulnerability trends shall be analyzed to improve security

6.3 Security Monitoring

- Applications shall include appropriate logging for security events
- Security logs shall be centrally collected and analyzed
- Suspicious activities shall be alerted and investigated
- Security monitoring shall be continuous
- Security incidents shall be reported and managed according to the Incident Management Policy

7. Training and Awareness

7.1 Developer Training

- Developers shall receive secure coding training
- Training shall be provided at onboarding and regularly thereafter
- Training shall be updated to address emerging threats
- Training effectiveness shall be measured
- Specialized training shall be provided for specific technologies

7.2 Security Champions

- Security champions shall be designated within development teams
- Security champions shall receive advanced security training
- Security champions shall promote security awareness
- Security champions shall assist with security reviews
- Security champions shall collaborate with the security team

8. Compliance and Exceptions

8.1 Compliance Verification

- Compliance with this policy shall be regularly assessed
- Security reviews shall verify adherence to secure development practices
- Automated tools shall be used to verify compliance where possible
- Non-compliance shall be reported and addressed

8.2 Exceptions

Exceptions to this policy shall be: - Documented with justification - Risk-assessed and approved by the Information Security Manager - Time-limited and regularly reviewed - Accompanied by compensating controls where appropriate

9. Roles and Responsibilities

9.1 Development Team

- Follow secure coding standards
- Participate in security training
- Perform peer code reviews with security focus
- Address identified security vulnerabilities
- Report security concerns

9.2 Security Team

- Develop and maintain secure coding standards
- Provide security guidance and consultation
- Review security architecture and design
- Conduct security testing and code reviews
- Verify security controls implementation

9.3 Project Managers

- Ensure security requirements are included in project planning
- Allocate resources for security activities
- Track security-related tasks and issues
- Ensure security sign-off before release
- Report on security status to stakeholders

9.4 Quality Assurance Team

- Include security testing in test plans
- Verify security requirements implementation
- Report security defects
- Validate security fixes
- Maintain security test cases

10. Related Documents

- Information Security Policy
- Change Management Policy
- Vulnerability Management Policy
- Incident Management Policy
- Access Control Policy
- Cryptography Policy
- [LIST OTHER RELEVANT POLICIES AND PROCEDURES]

11. Approval

This Secure Development Policy is approved by:

Name: ________________________ Position: ______________________ Date:
_________________________ Signature: _____________________