Secure Software Development (SDLC) Policy

Version number	3.1
Last Approved	Apr 29, 2025
Classification	PUBLIC

Overview

The Secure Software Development Policy provides a documented description of how software is built and maintained, emphasizing privacy and security. It describes the various phases of the development process and activities performed during each phase, to the extent applicable.

Applicability

The applicability of this statement falls under purview of the Security Documentation Overview.

Purpose

This Software Development Security Policy shall outline a standard expectation for implementation of a Software Development Lifecycle (SDLC) that produces software that is secure, accessible, and compliant with development standards, policies, and practices.

Scope

This policy defines the development and implementation requirements for Company products and applies to all Personnel who are involved with the development, hosting, and maintenance of Company products.

Secure Software Development Policy

Prerequisites and Development Standards

Security and Privacy *by Design* are crucial elements of the software development process and are risk reduction strategies for software engineers. The Company applies these concepts by

‘baking’ them into the software development life cycle (SDLC), incorporating security and privacy into each phase.

7 Foundational Principles of Privacy by Design

> *‘Proactive not reactive; preventative not remedial’*
>
> *‘Privacy as the default setting’*
>
> *‘Privacy embedded into design’*
>
> *‘Full functionality — positive sum, not zero sum’*
>
> *‘End-to-end security — full lifecycle protection’*
>
> *‘Visibility and transparency — keep it open’*
>
> *‘Respect for user privacy — keep it user-centric’*

Developers partake in annual Secure Development training created and administered by the Security team to reinforce secure coding concepts and standards.

SDLC Phases

Software development projects and releases must address the following areas in a manner consistent with industry standards and development practices. All SDLC phases must be addressed and incorporated in a consistent manner. Developers may make necessary adaptations based on the size and complexity of projects and releases. The Company’s SDLC uses a Scrum-style Agile methodology, consisting of six phases:

1. Preliminary Analysis Phase
2. Systems and Requirement Analysis Phase
3. Design and Development Phase
4. Documentation and Testing Phase
5. Operations and Deployment Phase
6. Maintenance Phase

Phase 1: Preliminary Analysis

Based upon a stakeholders initiation request, the objective of this phase is to conduct a preliminary analysis, propose alternative solutions, describe costs and benefits and submit a preliminary plan with recommendations.

Phase 2: Systems and Requirement Analysis

The second phase defines project/release goals into defined functions and operation of the intended application. It analyzes end-user information needs and addresses requirements for security, privacy, fraud prevention, platform use expectations, and AI considerations, including human oversight, transparency, and bias mitigation. Operational security concerns, such as virus scanning, zero day and malware protection, must also be factored in as appropriate based on the form and functions of the application.

In this phase, features and operations are described in detail, including screen layouts, business rules, process diagrams, pseudo code and other documentation. Depending upon the size of the project/release, prototyping can be useful in this stage. Larger complex projects/releases require more definition and more controls while smaller projects/releases may move directly to faster methodologies.

Phase 3: Design and Development

The design and development phase is focuses on writing code and converting design documentation into the actual software within the software development process. This stage of SDLC is generally the longest as it's the backbone to the whole process. The Company utilizes tools for project management, tracking bugs, stories, and other tasks. Development considerations include:

- **Privacy**: Application implementation and data collection need to be compliant with The Company's Privacy Statement, Client Data Privacy Compliance Policy and international privacy standards, Data Privacy Framework such as the EU/UK GDPR, the European Union (EU)-United States (US) Data Privacy Framework (DPF) Adequacy decision for the EU-US Data Privacy Framework | European Commission and the California Consumer Privacy Act.
- **Security**: Applications must be deployed within a secure hosting environment and should adhere to industry standards and protocols, such as https://owasp.org/www-project-application-security-verification-standard/ and https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards .
- **Fraud Prevention**: Appropriate controls must be defined and implemented to detect and prevent fraudulent activity/use of the system.

Phase 4: Documentation and Testing

Product documentation should adequately cover product functionality and design in order to enable the appropriate use and understanding of the products' purpose and functionality.

The testing phase brings all the pieces together into a testing environment, including checks for errors, bugs and interoperability, accessibility, mobility, performance, standards compliance, and

security.

Production data may not be used in Non-Prod environments, unless permission has been granted by a customer.

System hardening and testing elements include:

- Regression testing
- Upgrade testing
- Performance testing
- Secure code review
- Software license management
- Security testing
  - External Penetration Testing
  - Software Composition Analysis (SCA)
  - Static Application Security Testing (SAST)
  - Dynamic Application Security Testing (DAST)
- Anti-virus/malware protection
- Testing/validation of fraud controls
- Input validations
- Integration testing
- Documentation review
- Manifest update

Phase 5: Operations and Deployment

This is the final stage of initial development, where the software is put into production and is deployed. Code is released from Engineering after final testing, at which point the release is provided to the Operations team. This is the final checkpoint on architectural compliance, application and hosting security. No further modifications can be made after Engineering releases the code to the Operations team without formal review. Deployment to production environments is accomplished through Change Management.

Development, Testing and Production environments must be physically separate instances on different servers.

Phase 6: Maintenance

This final phase is ongoing through the rest of the software's life-cycle and includes the management of changes, corrections, patches, additions, migrations, decommissioning, and

more, all of which are required to undergo a formal Change Management process.

Document control

This policy is only controlled in its live, digital format. Any other format or export of this policy is an uncontrolled version of this document

Document Owner	@Art Machado	
Author(s)	@Art Machado @Paul Gordon @angelina.kilmer	
Required Approver(s) and Approval Date	@Art Machado - VP Information Security	Apr 29, 2025
Review cycle	ANNUAL	
Next review date	Feb 24, 2026	

Version History

Date	Author(s)	Version	Changes
Apr 29, 2025	@Art Machado @angelina.kilmer	3.1	Update of policy language
Feb 25, 2025	@Art Machado @Paul Gordon @angelina.kilmer	3.0	Annual review
Nov 1, 2024	@angelina.kilmer	2.3	Changed Policy classification from Confidential to Public
Mar 13, 2024	@Art Machado, Sarah	2.2	Annual review, AI considerations

	Zwicker, @Paul Gordon		
Aug 3, 2023	@John Cole	2.1	Linkage updates and policy references
Feb 23, 2023	@Art Machado, Sarah Zwicker	2.0	Annual Review + logo updates
Jul 14, 2022	@Art Machado	1.2	Added enhanced details to clarify various policy points.
Apr 14, 2022	Sarah Zwicker & @Art Machado	1.1	Initial Review and Approval
Mar 31, 2022	@Art Machado & Sarah Zwicker	1.0	Original version