Software Design Document (SDD)

Development Team

2024-06-26

Software Design Document (SDD)

1. Scope

1.1 Identification

SDD-001: Software Identification - The software **shall** be identified by the following information: - **Software Name**: [Software Name] - **Software Identifier**: [SW-001] - **Version**: 1.0 - **Release**: Initial Release - **Classification**: [Unclassified/Classified Level]

1.2 Software Overview

SDD-002: Software Purpose - The software **shall** provide [primary software functionality] - The software **shall** support [key operational capabilities] - The software **shall** integrate with [existing systems or infrastructure]

SDD-003: Software Context - The software **shall** be part of the [system name] system - The software **shall** interface with [other software components] - The software **shall** run on [specified hardware platform]

1.3 Document Overview

SDD-004: Document Purpose - This document **shall** describe the software design for [software name] - This document **shall** serve as the basis for software implementation - This document **shall** support software testing and maintenance

2. Referenced Documents

2.1 Government Documents

SDD-005: Military Standards - MIL-STD-498: Software Development and Documentation - MIL-STD-961E: Defense and Program-Unique Specifications Format and Content

2.2 Project Documents

SDD-006: Requirements Documents - Software Requirements Specification (SRS) - System/Subsystem Specification (SSS) - Interface Requirements Specification (IRS)

3. Design Overview

3.1 Design Philosophy

SDD-007: Design Principles - The software **shall** follow object-oriented design principles - The software **shall** implement separation of concerns - The software **shall** use design patterns where appropriate - The software **shall** support modularity and reusability

SDD-008: Architecture Approach - The software **shall** use microservices architecture - The software **shall** implement RESTful API design - The software **shall** support containerization - The software **shall** enable horizontal scaling

3.2 Design Constraints

SDD-009: Technology Constraints - The software **shall** be developed using [specified programming language] - The software **shall** use [specified framework] - The software **shall** run on [specified operating system] - The software **shall** use [specified database system]

SDD-010: Performance Constraints - The software **shall** respond to requests within 3 seconds - The software **shall** support 1000 concurrent users - The software **shall** use no more than 80% of available resources - The software **shall** maintain performance under load

3.3 Design Methods and Tools

SDD-011: Design Methods - The software **shall** use UML for design modeling - The software **shall** implement test-driven development - The software **shall** use continuous integration practices - The software **shall** follow agile development methodology

SDD-012: Design Tools - The software **shall** use [specified IDE] for development - The software **shall** use [specified version control system] - The software **shall** use [specified build tools] - The software **shall** use [specified testing frameworks]

4. System Architecture

4.1 System Overview

SDD-013: System Architecture - The system **shall** consist of the following major components: - **Web Layer**: User interface and presentation logic -

Application Layer: Business logic and application services - **Data Layer**: Data access and persistence - **Integration Layer**: External system integration

SDD-014: Component Relationships - The components **shall** communicate through well-defined interfaces - The components **shall** be loosely coupled - The components **shall** support independent deployment - The components **shall** enable horizontal scaling

4.2 System Context

SDD-015: External Dependencies - The system **shall** depend on [external systems] - The system **shall** integrate with [third-party services] - The system **shall** use [external databases] - The system **shall** communicate via [network protocols]

SDD-016: System Boundaries - The system **shall** have clear boundaries with external systems - The system **shall** implement security controls at boundaries - The system **shall** provide monitoring and logging at boundaries - The system **shall** support boundary testing

5. Detailed Design

5.1 Module Design

5.1.1 User Management Module **SDD-017: User Authentication Module** - **Module ID**: AUTH-001 - **Purpose**: Handle user authentication and authorization - **Responsibilities**: - User login and logout - Password management - Session management - Access control enforcement

SDD-018: User Profile Module - **Module ID**: PROFILE-001 - **Purpose**: Manage user profile information - **Responsibilities**: - Profile creation and updates - Preference management - Account settings - User preferences

5.1.2 Data Management Module **SDD-019: Data Access Module** - **Module ID**: DATA-001 - **Purpose**: Handle data access and persistence - **Responsibilities**: - Database operations - Data validation - Transaction management - Data caching

SDD-020: Data Processing Module - **Module ID**: PROCESS-001 - **Purpose**: Process and transform data - **Responsibilities**: - Business logic implementation - Data calculations - Data transformation - Business rule enforcement

5.1.3 Communication Module **SDD-021: API Module** - **Module ID**: API-001 - **Purpose**: Provide RESTful API services - **Responsibilities**: - API endpoint management - Request/response handling - API documentation - API versioning

SDD-022: Integration Module - **Module ID**: INTEGRATION-001 - **Purpose**: Handle external system integration - **Responsibilities**: - External API communication - Data synchronization - Error handling - Retry mechanisms

5.2 Interface Design

5.2.1 User Interface Design **SDD-023: Web Interface Design** - **Interface ID**: WEB-UI-001 - **Design Approach**: Responsive web design - **Technology Stack**: HTML5, CSS3, JavaScript, React - **Design Principles**: - Mobile-first design - Accessibility compliance - User experience optimization - Performance optimization

SDD-024: Mobile Interface Design - **Interface ID**: MOBILE-UI-001 - **Design Approach**: Progressive Web App (PWA) - **Technology Stack**: HTML5, CSS3, JavaScript, Service Workers - **Design Principles**: - Touch-friendly interface - Offline capability - Fast loading times - Native app-like experience

5.2.2 API Interface Design **SDD-025: REST API Design** - **Interface ID**: REST-API-001 - **Design Approach**: RESTful API design - **Technology Stack**: JSON, HTTP/HTTPS, JWT - **Design Principles**: - Resource-based URLs - HTTP method semantics - Stateless operations - Standard HTTP status codes

SDD-026: Database Interface Design - **Interface ID**: DB-API-001 - **Design Approach**: Data access layer abstraction - **Technology Stack**: SQL, ORM, Connection Pooling - **Design Principles**: - Connection pooling - Transaction management - Query optimization - Data validation

5.3 Data Design

5.3.1 Database Design **SDD-027: Database Schema** - **Database Type**: [Relational/NoSQL] database - **Schema Design**: Normalized database schema - **Key Features**: - Primary and foreign key relationships - Indexing strategy - Data constraints - Referential integrity

SDD-028: Data Models - **User Model**: User account and profile information - **Data Model**: Core business data entities - **Audit Model**: System audit and logging data - **Configuration Model**: System configuration data

5.3.2 Data Flow Design **SDD-029: Data Flow Architecture** - **Input Data Flow**: User input and external data sources - **Processing Data Flow**: Business logic and data transformation - **Output Data Flow**: Reports, notifications, and external systems - **Storage Data Flow**: Database operations and caching

SDD-030: Data Security Design - **Encryption**: Data encryption at rest and in transit - **Access Control**: Role-based data access control - **Audit Trail**:

Comprehensive data access logging - **Data Backup**: Automated backup and recovery procedures

6. Human-Machine Interface Design

6.1 User Interface Design

SDD-031: Interface Layout - The interface **shall** use a consistent layout design - The interface **shall** provide intuitive navigation - The interface **shall** support responsive design - The interface **shall** comply with accessibility standards

SDD-032: User Experience Design - The interface **shall** provide clear visual hierarchy - The interface **shall** use consistent color schemes - The interface **shall** provide helpful error messages - The interface **shall** support user customization

6.2 User Interaction Design

SDD-033: Interaction Patterns - The interface **shall** use standard interaction patterns - The interface **shall** provide immediate feedback - The interface **shall** support keyboard navigation - The interface **shall** implement progressive disclosure

SDD-034: Accessibility Design - The interface **shall** comply with WCAG 2.1 AA standards - The interface **shall** support screen readers - The interface **shall** provide keyboard alternatives - The interface **shall** use sufficient color contrast

7. Requirements Traceability

7.1 Design to Requirements Traceability

SDD-035: Functional Requirements Traceability - Each functional requirement **shall** be traced to design components - Design components **shall** implement specific requirements - Requirements **shall** be validated through design review - Design changes **shall** be tracked against requirements

SDD-036: Non-Functional Requirements Traceability - Performance requirements **shall** be addressed in design - Security requirements **shall** be implemented in design - Reliability requirements **shall** be considered in design - Maintainability requirements **shall** be supported by design

7.2 Design Verification

SDD-037: Design Review Process - Design **shall** be reviewed by technical stakeholders - Design **shall** be validated against requirements - Design **shall** be assessed for feasibility - Design **shall** be approved before implementation

8. Notes

8.1 Acronyms and Abbreviations

- **SDD**: Software Design Document
- **API**: Application Programming Interface
- **CSS**: Cascading Style Sheets
- **HTML**: HyperText Markup Language
- **JSON**: JavaScript Object Notation
- **JWT**: JSON Web Token
- **ORM**: Object-Relational Mapping
- **PWA**: Progressive Web App
- **REST**: Representational State Transfer
- **SQL**: Structured Query Language
- **UML**: Unified Modeling Language
- **WCAG**: Web Content Accessibility Guidelines

8.2 Definitions

- **Module**: A self-contained component of the software
- **Interface**: A boundary between software components
- **Architecture**: The overall structure of the software system
- **Design Pattern**: A reusable solution to common design problems
- **Component**: A modular part of the software system

8.3 Background Information

This Software Design Document follows MIL-STD-498 guidelines and provides a comprehensive framework for software design. The design is structured to support implementation, testing, and maintenance throughout the software lifecycle.