# Requirements Document

## Agentic SDLC Automation Project

### 1. Introduction

#### 1.1 Purpose

This document outlines the comprehensive requirements for the Agentic SDLC Automation Project. It serves as the primary reference for the design, development, testing, and implementation of the system.

#### 1.2 Scope

The Agentic SDLC system will automate key aspects of the software development lifecycle, including JIRA monitoring, requirements analysis, story creation, test generation, and release management. The system will employ AI agents to perform these tasks with appropriate human oversight.

#### 1.3 Definitions, Acronyms, and Abbreviations

* **SDLC**: Software Development Life Cycle
* **JIRA**: Issue tracking and project management software
* **UAT**: User Acceptance Testing
* **API**: Application Programming Interface
* **MVP**: Minimum Viable Product
* **Agent**: AI component designed to perform specific functions autonomously

#### 1.4 References

* JIRA API Documentation
* Organizational SDLC Policies
* Story Template Standards (see attached template)
* Test Framework Documentation

#### 1.5 Overview

This document is organized into functional requirements, non-functional requirements, system interfaces, and constraints. It will guide the development team in creating a system that meets stakeholder needs while ensuring quality and compliance.

### 2. Overall Description

#### 2.1 Product Perspective

The Agentic SDLC system will integrate with existing JIRA infrastructure to automate routine aspects of the software development lifecycle. It will operate as a supervisor-agent architecture where specialized agents handle distinct aspects of the process under the coordination of a supervisor agent.

#### 2.2 Product Functions

At a high level, the system will: - Monitor JIRA boards for new “To Do” items - Analyze requirements for completeness and clarity - Request clarification from stakeholders when needed - Generate standardized stories and tasks - Create automated tests - Execute tests and report results - Manage user acceptance testing - Coordinate releases

#### 2.3 User Characteristics

The system will interact with: - **Development Teams**: Engineers, architects, and technical leads - **Product Managers**: Defining requirements and accepting deliverables - **Project Managers**: Tracking progress and resource allocation - **QA Engineers**: Defining and validating tests - **Business Requestors**: Initiating requirements and providing clarification - **System Administrators**: Configuring and maintaining the system

#### 2.4 Constraints

* Must integrate with existing JIRA instances
* Must adhere to organizational security policies
* Must maintain audit trails for all automated actions
* Must provide override capabilities for human users
* Must operate within existing infrastructure constraints

#### 2.5 Assumptions and Dependencies

* JIRA API availability and stability
* Access to appropriate permissions for JIRA operations
* Standardized story templates are defined and available
* Testing frameworks are accessible via API

### 3. Functional Requirements

#### 3.1 JIRA Monitoring

##### 3.1.1 Board Monitoring

* The system shall continuously monitor specified JIRA boards
* The system shall detect new items added to “To Do” columns
* The system shall identify the project context of each item
* The system shall track changes to monitored items

##### 3.1.2 Event Handling

* The system shall trigger appropriate workflows upon detecting new items
* The system shall maintain a queue of items requiring processing
* The system shall prioritize items based on configurable rules
* The system shall handle concurrent processing of multiple items

#### 3.2 Requirements Analysis

##### 3.2.1 Completeness Check

* The system shall analyze new items against the standard template
* The system shall identify missing required fields
* The system shall evaluate the clarity and specificity of requirements
* The system shall score requirements based on completeness metrics

##### 3.2.2 Clarification Management

* The system shall formulate specific questions for incomplete requirements
* The system shall communicate with requestors via JIRA comments
* The system shall track outstanding clarification requests
* The system shall incorporate responses into the requirement analysis
* The system shall escalate to human intervention after configurable timeouts

#### 3.3 Story Generation

##### 3.3.1 Template Application

* The system shall apply the standard story template to create structured stories
* The system shall populate template fields from requirement details
* The system shall generate unique identifiers for created stories
* The system shall link stories to original requirements

##### 3.3.2 Task Breakdown

* The system shall decompose stories into appropriate tasks
* The system shall assign standard task types based on story characteristics
* The system shall estimate effort based on historical data
* The system shall identify dependencies between tasks

##### 3.3.3 Project Organization

* The system shall tag all generated items with appropriate project IDs
* The system shall maintain traceability between requirements and stories
* The system shall organize related stories and tasks into epics when appropriate
* The system shall ensure no cross-contamination between projects

#### 3.4 Testing Framework

##### 3.4.1 Test Generation

* The system shall create automated tests based on acceptance criteria
* The system shall generate unit, integration, and system test templates
* The system shall link tests to stories and requirements
* The system shall estimate test coverage
* The system shall identify high-risk areas requiring additional testing

##### 3.4.2 Test Execution

* The system shall execute tests against developed code
* The system shall capture and analyze test results
* The system shall generate detailed failure reports
* The system shall track test status across test cycles
* The system shall identify regression issues

##### 3.4.3 Test Management

* The system shall manage test environments
* The system shall schedule test executions
* The system shall track test execution history
* The system shall provide test coverage metrics

#### 3.5 User Acceptance Testing

##### 3.5.1 UAT Coordination

* The system shall manage the UAT process for completed items
* The system shall notify appropriate stakeholders of items ready for UAT
* The system shall capture and track UAT feedback
* The system shall categorize feedback by type and severity

##### 3.5.2 Feedback Processing

* The system shall analyze UAT feedback for actionability
* The system shall route feedback to appropriate teams
* The system shall track feedback resolution
* The system shall verify feedback implementation

#### 3.6 Release Management

##### 3.6.1 Release Planning

* The system shall track items eligible for release
* The system shall group items into logical release packages
* The system shall identify dependencies affecting release readiness
* The system shall generate release notes from completed items

##### 3.6.2 Release Coordination

* The system shall schedule releases based on organizational policies
* The system shall coordinate with deployment systems
* The system shall track release status
* The system shall notify stakeholders of release activities
* The system shall capture release metrics

#### 3.7 Supervisor Functions

##### 3.7.1 Workflow Coordination

* The system shall maintain overall workflow context
* The system shall delegate tasks to appropriate specialized agents
* The system shall handle agent communication
* The system shall resolve conflicts between agent recommendations

##### 3.7.2 Exception Handling

* The system shall detect and respond to process exceptions
* The system shall implement recovery procedures for failures
* The system shall escalate to human intervention when required
* The system shall maintain process integrity during recovery

##### 3.7.3 Human Oversight

* The system shall provide configurable approval checkpoints
* The system shall present relevant information for human decisions
* The system shall implement override mechanisms
* The system shall record all human interventions

### 4. Non-Functional Requirements

#### 4.1 Performance

##### 4.1.1 Response Time

* The system shall detect new JIRA items within 5 minutes of creation
* The system shall complete requirement analysis within 15 minutes
* The system shall generate stories within 10 minutes of completed analysis
* The system shall execute test suites within timeframes appropriate to test size

##### 4.1.2 Throughput

* The system shall support processing of at least 100 concurrent items
* The system shall handle at least 500 JIRA items per day
* The system shall support at least 50 simultaneous users

##### 4.1.3 Capacity

* The system shall store up to 5 years of historical data
* The system shall support up to 100 active projects
* The system shall handle attachments up to 100MB in size

#### 4.2 Security

##### 4.2.1 Authentication

* The system shall implement role-based access control
* The system shall integrate with organizational authentication systems
* The system shall enforce password policies
* The system shall support multi-factor authentication for administrative functions

##### 4.2.2 Authorization

* The system shall restrict access based on user roles
* The system shall enforce separation of duties
* The system shall maintain permission inheritance hierarchies
* The system shall prevent unauthorized access to sensitive data

##### 4.2.3 Audit

* The system shall maintain comprehensive audit logs
* The system shall record all automated actions
* The system shall track user interactions with the system
* The system shall preserve audit data for at least 1 year

#### 4.3 Reliability

##### 4.3.1 Availability

* The system shall be available 99.5% of the time during business hours
* The system shall implement fault tolerance for critical components
* The system shall recover from failure within 15 minutes
* The system shall provide degraded operation during partial failures

##### 4.3.2 Recoverability

* The system shall perform regular state backups
* The system shall recover from data corruption without loss
* The system shall restore to the last known good state after failures
* The system shall implement transaction integrity

##### 4.3.3 Error Handling

* The system shall detect and log all errors
* The system shall implement graceful degradation
* The system shall notify administrators of critical errors
* The system shall provide diagnostic information for troubleshooting

#### 4.4 Usability

##### 4.4.1 User Interface

* The system shall provide intuitive dashboards for monitoring
* The system shall implement consistent navigation patterns
* The system shall display clear error and status messages
* The system shall support accessibility standards

##### 4.4.2 Documentation

* The system shall provide comprehensive online help
* The system shall include user guides for each role
* The system shall offer contextual assistance
* The system shall maintain up-to-date documentation

#### 4.5 Maintainability

##### 4.5.1 Configurability

* The system shall support configuration without code changes
* The system shall implement rule-based behavior customization
* The system shall allow template modifications
* The system shall permit workflow adjustments

##### 4.5.2 Extensibility

* The system shall implement a modular architecture
* The system shall support plugin development
* The system shall provide well-documented APIs
* The system shall facilitate integration with new systems

#### 4.6 Scalability

* The system shall scale horizontally to support increased load
* The system shall support distributed processing
* The system shall optimize resource usage under varying loads
* The system shall maintain performance during peak periods

### 5. System Interfaces

#### 5.1 JIRA Integration

* The system shall connect to JIRA via REST API
* The system shall implement webhook receivers for real-time updates
* The system shall handle JIRA authentication securely
* The system shall maintain compatibility with JIRA versions in use

#### 5.2 Version Control Integration

* The system shall integrate with Git repositories
* The system shall track branch creation and merges
* The system shall associate code changes with stories
* The system shall support multiple version control platforms

#### 5.3 Testing Framework Integration

* The system shall connect to automated testing tools
* The system shall support multiple test frameworks
* The system shall capture and process test results
* The system shall integrate with test management systems

#### 5.4 Communication Systems

* The system shall send notifications via email
* The system shall post updates to collaboration platforms
* The system shall support custom notification channels
* The system shall manage notification preferences

### 6. Implementation Constraints

#### 6.1 Development Environment

* The system shall be developed using approved technologies
* The system shall follow organizational coding standards
* The system shall implement continuous integration
* The system shall utilize approved development tools

#### 6.2 Deployment Environment

* The system shall operate in the organizational cloud environment
* The system shall comply with infrastructure security requirements
* The system shall support containerized deployment
* The system shall implement automated deployment

#### 6.3 Legal and Regulatory

* The system shall comply with data protection regulations
* The system shall maintain appropriate data residency
* The system shall implement required compliance controls
* The system shall support audit requirements

### 7. MVP Requirements

The following requirements represent the Minimum Viable Product to be delivered within the 2-month timeline:

#### 7.1 MVP Functional Requirements

* Basic JIRA board monitoring and event detection
* Simple requirements completeness checking
* Question generation for incomplete requirements
* Story creation using standard templates
* Basic test generation capabilities
* Test execution and result reporting
* Simple UAT coordination
* Basic release planning

#### 7.2 MVP Non-Functional Requirements

* Essential security controls
* Basic error handling
* Minimum performance requirements
* Core JIRA integration
* Fundamental audit logging
* Basic user interfaces for monitoring

### 8. Approval

This requirements document requires review and approval by the following stakeholders: - Project Sponsor - Product Owner - Technical Lead - QA Lead - Security Officer

### Appendices

#### Appendix A: Story Template

[Reference to attached JIRA Story Template]

#### Appendix B: Acceptance Criteria Standards

[Reference to organizational acceptance criteria standards]

#### Appendix C: Technical Environment Specifications

[Reference to technical environment documentation]