Software Requirements Specification Document (SRSD)

Project: Web-Based System for Insurance Companies (WBSIC)

Version: 1.0

Date: October 2025

Prepared by: Musanze DevOps Team

1. Introduction

Purpose: Defines the software requirements for WBSIC, describing the system functionality, performance, interfaces, and constraints.
Scope: The WBSIC is a comprehensive web-based system that handles customer, policy, claim, payment, and reporting modules using React.js and Node.js, integrated with DevOps tools for CI/CD and automation.
Intended Audience: Developers, Testers, System Administrators, and Project Managers.
References: Jira, React.js, Node.js, Docker, Kubernetes, IEEE SRS Template.

2. Overall Description

Product Perspective: Multi-tier web system including frontend (React), backend (Node.js), database (MongoDB/PostgreSQL), and automated CI/CD with Jenkins, Docker, and Kubernetes.
Product Features: Customer management, Policy management, Claims processing, Payments & Billing, and Reporting.
User Classes: Customers, Staff, Admins, DevOps Engineers.
Environment: Node.js v18+, MongoDB, Docker, Jenkins, Kubernetes, Linux Ubuntu Server.

3. System Features

Customer & User Management: Registration, authentication, roles.
Policy Management: CRUD policies, approvals.
Claims Processing: Submission, approval workflow, notifications.
Payments & Billing: Payment gateway, invoices, reminders.
Reporting & Analytics: Dashboards, metrics, export to CSV/PDF.

4. Non-Functional Requirements

Performance: Handle 2M daily requests, <500ms latency.
Security: HTTPS, JWT, encryption.
Scalability: Kubernetes auto-scaling.
Availability: 99.9% uptime.
Maintainability: Modular CI/CD.
Usability: Responsive UI.
Portability: Dockerized deployment.

5. External Interface Requirements

User Interface: Responsive React UI with admin dashboards.
Hardware: Cloud-hosted (AWS/Azure/GCP).
Software Interfaces: REST APIs, Payment Gateway, SMTP/Twilio.
Communication: HTTPS REST API, WebSocket notifications.

6. System Architecture

Frontend: React.js SPA
Backend: Node.js + Express
Database: MongoDB
Infrastructure: Docker, Jenkins, Kubernetes, Helm, Ansible
Tracking: Jira Scrum with 5 teams.

7. Constraints

Must handle 10M users, automated Jenkins deployment, GDPR compliance, persistent Docker volumes.

8. Future Enhancements

- AI-based fraud detection
- Chatbot integration
- Multi-language and multi-currency
- Mobile app version.

9. Team Responsibilities

Group-1: Customer & User Management
Group-2: Policy Management
Group-3: Claims Processing
Group-4: Payments & Billing
Group-5: Reporting & Analytics Dashboard