**Software Requirement Specification (SRS)**

**Project Title:** InsurAI **–** Corporate Policy Automation and IntelligenceSystem **Prepared by:** Eshita Talukdar **Date:** 19th August, 2025

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to define the requirements for *InsurAI – Corporate Policy Automation and Intelligence System*. The system will automate essential insurance policy functions such as policy creation, renewal, claims management, and provide intelligent customer support through an AI-powered assistant. This document outlines the purpose and scope of the system, provides an overall description, details system features, specifies functional and non-functional requirements of the project.

**1.2 Scope**

InsurAI is designed as a corporate insurance policy management system enhanced with AI-based automation. It will support:

* Policy lifecycle management (creation, renewal, termination).
* Claims request submission and tracking.
* AI-based assistance for answering policy-related FAQs.
* Administrative dashboards for managing policies and customers.
* Secure authentication and authorization will be provided.

**1.3 Definitions, Acronyms, Abbreviations**

|  |  |
| --- | --- |
| **Term** | **Definition** |
| AI | Artificial Intelligence |
| FAQ | Frequently Asked Questions |
| MVP | Minimum Viable Product |
| Policy Lifecycle | Creation, update/renewal, termination of an insurance policy |

**1.4 References**

* IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.
* Sommerville, Ian. Software Engineering, 10th Edition, Pearson, 2015.

**1.5 Technologies to be used**

|  |  |
| --- | --- |
| **Component** | **Technology/Tool** |
| Frontend | HTML, CSS, React |
| Backend | Spring Boot |
| Database | MySQL |
| AI/ML Services | Gemini APIs (Spring AI) |
| Authentication & Security | JWT, OAuth2 |
| Deployment | Docker/Cloud deployment |

**1.6 Overview**

This document outlines the proposed system for automating insurance policy management. It defines the project scope, objectives, and the technologies to be used, including Spring Boot, MySQL, JWT/OAuth2, and AI integration. The overall description highlights how the system will ensure secure authentication, efficient data handling, and intelligent policy automation.

**2. Overall Description**

**2.1 Product Perspective**

InsurAI is a standalone web-based application that integrates automation and intelligent services for policy management. The system provides interfaces for both administrators and customers.

**2.2 Product Functions**

* **Customer Functions:**
  + Register/login to the portal
  + View available policies
  + Apply for policy and track renewal
  + Submit claims and track status
  + Interact with AI assistant for FAQs
* **Administrator Functions:**
  + Manage policies (add/update/remove)
  + Manage claims (approve/reject)
  + View customer details and policy allocations
  + Monitor system reports

**2.3 User Characteristics**

* Customers: Insurance policyholders with basic computer literacy.
* Administrators: Company staff with training in policy and claims processing**.**

**2.4 Constraints**

* Time constraint
* Limited resources and workforce
* AI services dependent on third-party APIs.

**2.5 Assumptions and Dependencies**

* Users have stable internet connectivity.
* AI API (Gemini) is accessible during operations.
* The system will initially run on a web environment only.

**3. Specific Requirements**

**3.1 Functional Requirements**

The system must support the following functions:

1. **Authentication & Authorization**
   * Secure login and registration using JWT and OAuth2.
   * Role-based access control for users, agents, and admins.
2. **Policy Management**
   * Create, update, and delete insurance policies.
   * Search and view policy details.
3. **Customer Dashboard**
   * View active and expired policies.
   * Check claim status and premium details.
4. **Claims Processing**
   * Submit and track claims online.
   * Notify customers about claim status updates.
5. **Payments**
   * Process premium payments securely.
   * Generate receipts for transactions.
6. **Admin Functions**
   * Manage user accounts and roles.
   * Update policy information.
   * Monitor system reports and performance.

**3.2 Non-Functional Requirements**

1. **Performance**
   * The system should handle multiple requests simultaneously with minimal delay.
2. **Scalability**
   * The architecture must allow easy scaling to support growing users and policies.
3. **Security**
   * Data encryption for sensitive information.
   * Strong authentication and authorization measures.
4. **Usability**
   * Simple and intuitive interface for customers, agents, and admins.
5. **Availability**
   * System should be available 24/7 with minimal downtime.
6. **Maintainability**
   * Code should be modular and easy to maintain or update.
7. **Time Constraint**
   * The system should be developed and deployed within the given project timeline.