Template - Requirements Specifications Document

# Introduction - *This introduction is very important as it sets expectations that we will come back to throughout the SRS.*

## Purpose -*Define the purpose of these requirements here.*

The purpose is to outline the functional and non-functional requirements for developing a Big Data analysis platform for a Health Care insurance company. The platform aims to enhance the company's revenue by analyzing competitors' data and customer behavior, enabling customized offers and royalty calculations for policyholders.

* 1. Intended Audience and Use - *Define who in your organization will have access to the SRS and how they should use it. This may include developers, testers, and project managers.*

Software Requirement Specification:

Developers: To understand the functional and technical requirements for implementation.

Testers: To create test cases and validate the functionality.

Project Managers: To ensure the project stays on track and meets its objectives.

Data Analysts; To analyze the interpret data for business insights.

Stakeholders: to understand the scope and goal of the project.

## Product Scope - *What are the benefits, objectives, and goals we intend to have for this product? This should relate to overall business goals, especially if teams outside of development will have access to the SRS.*

The product will provide a comprehensive data pipeline and analysis system to enhance the Health Care insurance company's ability to understand customer behavior and improve revenue through targeted offers and royalty calculations.

Benefits include:

-Understanding of customer needs.

Improved ability to customize insurance offers.

Increased revenue through strategic business decisions.

Integration with existing data sources and systems for comprehensive analysis.

## Definitions and Acronyms -*Clearly define all key terms, acronyms, and abbreviations used in the SRS. This will help eliminate any ambiguity and ensure that all parties can easily understand the document.*

SRS: Software Requirements Specification

AWS: Amazon Web Services

S3: Simple Storage Service

EMR: Elastic MapReduce

ETL: Extract, Transform, Load

API: Application Programming Interface

DBMS: Database Management System

Pyspark: Python API for Spark

Jira: Project management tool

GitHub: Source code repository

# Overall Description - *Your next step is to give a description of what you’re going to build. Why is this product needed? Who is it for? Is it a new product? Is it an add-on to a product you’ve already created? Is this going to integrate with another product? Understanding and getting your team aligned on the answers to these questions on the front end makes creating the product much easier and more efficient for everyone involved.*

## User Needs - *Describe who will use the product and how. Understanding the various users of the product and their needs is a critical part of the SRS writing process.*

The primary users of the product include:

Insurance Analysts: Need insights into customer behavior and competitors' data to strategize offers and royalties.

Data Scientists: Require robust data pipelines for analyzing large datasets.

Business Managers: Need actionable insights to drive revenue and improve customer satisfaction.

IT Administrators: Require reliable and secure data integration and storage solutions.

## Assumptions and Dependencies - *What are we assuming will be true? Understating and laying out these assumptions ahead of time will help with headaches later. Are we assuming current technology? Are we basing this on a Windows framework? We need to take stock of these technical assumptions to better understand where our product might fail or not operate perfectly.*

The system will use AWS services such as S3,Redshift and EMR.

The platform will integrate with databricks for data processing.

The development environment will include Jira for project management and Github for version control.

Users will have the necessary permissions to access AWS and Databricks resources.

The product will rely on current technologies compatible with AWS and Databricks.

# System Features and Requirements -*In order for your development team to meet the requirements properly, we must include as much detail as possible. This can feel overwhelming but becomes easier as you break down your requirements into categories.*

## Functional Requirements - *Functional requirements are essential to your product because, as the name implies, they provide some sort of functionality. Asking yourself questions such as “does this add to my tool’s functionality?” or “what function does this provide?” can help with this process. You may also have requirements that outline how your software will interact* *with other tools*

Data Ingestion: Collect data from various sources including web scraping and third-party APIs.

Data Cleaning: Implement data cleaning processes to handle null values, duplicates, and incorrect formats.

Data Storage: Store raw and cleaned data in AWS S3 and Redshift.

Data Analysis: Perform analysis to generate insights such as disease claims, subscriber demographics, and policy profitability.

Report Generation: Create reports and visualizations using Databricks for stakeholders.

## External Interface Requirements - *You may also have requirements that outline how your software will interact with other tools There are several types of interfaces you may have requirements for, including:*

### User-Interface: web-based interface for accessing reports and visualizations.

### Hardware Interface:Standard computing devices (PC,laptop) with internet access.

### Software Interface: Integration with third party APIs and internal databases.

### Communications: Secure communications over HTTPs for data transfer.

## System Features - *System features are a type of functional requirements. These are features that are required in order for a system to function.*

Dashboard: Customizable dashboard to display key metrics and insights.

Search Functionality: Ability to search and filter data based on various criteria.

User roles and permission: Role-based access control for different user types.

## Nonfunctional Requirements - *Nonfunctional requirements, which help ensure that a product will work the way users and other stakeholders expect it to, can be just as important as functional ones. These may include:*

## Performance Requirements: The system should handle concurrent users with response times under 2 seconds.

## Safety Requirements: Regular data backups and disaster recovery plans.

## Security Requirements: Data encryption, secure access controls, and regular security audits.

## Usability Requirements: Intuitive and accessible user interface with support for users with disabilities.

## Scalability Requirements: Ability to scale with increasing data volumes and user load.