Template - Requirements Specifications Document

# **Introduction** – A health Care insurance company that want to understand customer behavior and market trends that is critical for driving revenue growth.this project focuses on building big data pipeline to help health insurance company analyze competitor and customer data from various sources.

## **Purpose** :- The purpose of this project is to develop a scalable and robust Big Data pipeline that enables a healthcare insurance company to gain actionable insights from competitor and customer data. These insights will be used to personalize offers, calculate customer royalties, identify profitable policies, and ultimately improve revenue and market competitiveness.

## **Intended Audience and Use** – This project can be used by Business analysts, Data engineers, Data scientists, Product Managers, Health insurance company

## **Product Scope** - The product scope includes building a Big Data pipeline to collect, process, and analyze healthcare insurance data from multiple sources.It aims to generate insights on customer behavior, policy trends, and claim patterns. The solution will support strategic decisions by enabling personalized offers, royalty calculations, and revenue optimization.

## **Definitions and Acronyms**:- it uses ETL(Extract, transform, Load) to process for data pipeline, Big data for large data sets, HDFS(Hadoop Distributed File System) to store large data set in distributed system, spark for big data processing

# **Overall Description –** This product involves building a Big Data analytics pipeline to support a healthcare insurance company in improving revenue and understanding customer behavior through actionable insights derived from competitor and customer data. This is a new standalone solution designed to integrate seamlessly with cloud services like AWS Redshift, S3, and Databricks. It ingests, processes, and analyzes structured and unstructured data from third-party and web-scraped sources.

## **User Needs** –

## **Business Analysts:** Require access to cleansed, query-ready data for analysis and dashboarding.

## **Health insurance company:** Need summarized reports to identify profitable customer segments and policy strategies.

## **Data Engineers:** Need to automate data ingestion, cleaning, and transformation processes for scalable workflows.

## **Data Scientists:** Need a rich dataset to build models for predicting claim trends and customer lifetime value.

## **Assumptions and Dependencies** – The data will be available in structured format for initial ingestion. The environment is cloud-based (AWS), assuming availability of S3, Redshift, and EMR Studio. Dependencies include Databricks for notebooks, PySpark for processing, and GitHub/Jira for code/version control and task tracking. It is assumed that users have access rights to Redshift, S3 buckets, and other related services.

# **System Features and Requirements:-**It requirements for data ingestion, cleaning, transformation, and storage using AWS and PySpark tools. It defines how the system interacts with cloud infrastructure, ensures data quality, supports business queries, and adheres to performance, security, and scalability standards.

## **Functional Requirements -** Enable ingestion, cleaning, and transformation of healthcare datasets using PySpark. Store cleaned datasets (Patients, Subscribers, Claims, Group\_Subgroup) in Redshift. Create analytical queries and outputs for defined business requirements. Maintain version control via GitHub and track tasks in Jira.

## External Interface Requirements –

## **User Interface:** No dedicated UI; compatible with BI tools.

## **Hardware Interface:** Operates fully in AWS cloud.

## **Software Interface:** Uses S3, Redshift, EMR/Databricks, GitHub, and Jira.

## **Communications Interface:** AWS service integrations (e.g., S3 to Redshift).

## **System Features** - Null value detection and replacement, duplicate removal. Schema creation and loading of cleaned data to Redshift. Generate output tables for use cases like claim frequency, subgroup analysis, and profit trends.

## **Nonfunctional Requirements –**

## **Performance:** Pipeline execution as fast as possible for datasets.

## **Safety:** Enforced role-based access control.

## **Security:** Data encryption and IAM-based policies.

## **Usability:** Code should be clean, documented, and reusable.

## **Scalability:**Architecture supports data growth and future integration.