**Requirements Specifications Document**

**Introduction**

For a healthcare insurance company, analyzing data can lead to making better decisions, creating insurance offers that people want, and figuring out how to make more money while keeping customers happy. This project is all about using big data to find those answers. We’re going to collect a lot of data, from what kind of insurance claims people make, to how old they are when they make them, and even which hospitals are the busiest.

**a. Purpose**

The main reason we’re doing this is to build a special application that will help the company look through all this data and find useful bits of information. This is going to help the company offer better deals to customers, reward people who stick with us for a long time, and overall, make more money by being smarter about what the hospital offers.

**b. Intended Audience and Use**

This document is for everyone who’s going to help make this project happen. That includes the people who write the code, those who make sure everything’s running smoothly, the bosses who keep an eye on progress, and even the big decision-makers in the company. It’s a guidebook that will help everyone stay on the same page, understand what we’re building, and how we’re going to get there.

**c. Product Scope**

This application is going to help us understand our customers like - what they need, what they’re worried about when it comes to health, and how we can offer them something that no one else is. By getting to know our customers better, we can make them happier, and a happy customer is good for business.

**d. Definitions and Acronyms**

* **Big Data:** Refers to exceptionally large volumes of data that cannot be efficiently processed or analyzed using traditional database and software techniques. Big Data encompasses structured, semi-structured, and unstructured data that requires advanced tools and methodologies for insights and decision-making.
* **AWS S3 (Amazon Simple Storage Service):** A scalable cloud storage service provided by Amazon Web Services. It offers industry-leading durability, availability, and scalability for storing and retrieving any amount of data from anywhere on the web.
* **AWS Redshift:** A fully managed, petabyte-scale data warehouse service offered by Amazon Web Services. It enables fast analysis of large datasets using SQL and integrates seamlessly with data lakes and BI tools.
* **Databricks:** An analytics platform powered by Apache Spark. It provides a unified environment for data engineering, collaborative data science, full-lifecycle machine learning, and business analytics through a highly collaborative and interactive workspace.
* **PySpark:** The Python API for Apache Spark, an open-source, distributed computing system that provides an interface for programming entire clusters with implicit data parallelism and fault tolerance. PySpark enables users to interface Spark with Python, facilitating data analysis and processing in a scalable and efficient manner.
* **Jira:** A project management and issue tracking software developed by Atlassian. It is designed to improve team collaboration by organizing tasks, tracking their progress, and managing deadlines and deliverables.
* **GitHub:** A Git repository hosting service that provides a web-based graphical interface. It offers source code management (SCM) and distributed version control functionalities of Git alongside its own features, facilitating code sharing and collaboration across teams.

**2. Overall Description**

**a. User Needs**

The users of this project are the employees at the healthcare insurance company. They need a way to make sense of heaps of data from various sources. Specifically, they're looking for insights on:

* Which disease has a maximum number of claims.
* Find those Subscribers having age less than 30 and they subscribe any subgroup
* Find out which group has maximum subgroups.
* Find out hospital which serve most number of patients
* Find out which subgroups subscribe most number of times
* Find out total number of claims which were rejected
* From where most claims are coming (city)
* Which groups of policies subscriber subscribe mostly Government or private
* Average monthly premium subscriber pay to insurance company.
* Find out Which group is most profitable
* List all the patients below age of 18 who admit for cancer
* List patients who have cashless insurance and have total charges greater than or equal for Rs. 50,000.
* List female patients over the age of 40 that have undergone knee surgery in the past year

**b. Assumptions and Dependencies**

We're moving forward with a couple of key assumptions:

1. **Tech Readiness**: We’re counting on the availability and reliability of AWS services like S3, Redshift, along with Databricks for data processing and PySpark for analytics. These are the building blocks of our project.
2. **Data Quality**: The success of this project hinges on the quality and completeness of the data we're working with. We assume that the data collected from various sources will be robust and comprehensive enough for meaningful analysis.

**3. System Features and Requirements**

**a. Functional Requirements**

1. **Data Gathering**: This application should needs to be able to pull in information from a csv files and load it into amazon s3.
2. **Connection to other platforms**: This application should be able to connect to platforms like Amazon s3 and Redshift to read and write the data.
3. **Cleaning the Mess**: Not all data comes neat and tidy. Sometimes it's messy, with missing pieces or bits that don’t make sense. This application needs to make sure the data is spotless and ready to use. This means fixing errors, filling in the blanks, or throwing out the stuff that just doesn’t belong.
4. **Finding Patterns**: The whole point of gathering all this data is to spot trends and patterns. Maybe it’s finding out which illnesses lead to the most insurance claims, or figuring out if younger people prefer different kinds of insurance. Whatever it is, this application should be able to spot these trends quickly and accurately.
5. **Unloading the report**: Lastly, this application needs to be able to store the generated output to the redshift.

b. **External Interface Requirements**

1. **Hardware Compatibility:** Our application needs to work with the computers and servers we already have. It should run smoothly, whether it's on a desktop in the office, a laptop at home, or even a tablet.
2. **Software Connections:** Our application should be able to talk to other systems like S3 and redshift, grab the data it needs, and then put its findings back in a way that these systems can understand.

c. **System Features**

1. **Data Collection**: We need to make sure our application can automatically gather data from amazon S3
2. **Data Cleaning Engine**: This feature is all about making the data look pretty and usable. It'll take the messy, jumbled data and clean it up by fixing errors, filling in missing pieces, and getting rid of anything we don't need.
3. **Trend Spotting**: Here's where things get really interesting. Our application will not just look at the data; it'll actually transform it to get the required use cases.
4. **Data unloading**: Lastly, this application unloads the results to redshift that can be used to make business decisions.
5. **Scalability**: As the data grows, our application will grow with us. It's built to handle more data, more users, and more complexity as it is on a scalable system.

d. **Nonfunctional Requirements**

1. **Performance**: Our application needs to be fast. It should be able to handle big loads of data and spit out results quickly, even when there is large amount of data.
2. **Reliability**: This application should not have any bugs or downtime. It should be able to recover quickly and should not lose any data.
3. **Security**: Since the data we are dealing with sensitive information, our application's security must be ironclad.
4. **Scalability**: As the data grows, our application needs to be able to grow as well. That means being able to handle more data, more users, and more complex analyses without stuttering or slowing down.