A black rectangle with a black background

Description automatically generated with low confidenceLogo, company name

Description automatically generated

DAEN 690

Project Report

Bhavana Macha

Prakruti Rajendra Kothari

Sachin Mote

Sahithi Oddiraju

Saiprem Vemulapalli

Ravinder Singh

Spring 2023

Integrating Data Catalogs with content search technologies

**About the Cover**

Dr. Isaac Gang is an Associate Professor at the George Mason University College of Engineering and Computing, Volgenau School of Engineering, MS Data Analytics Engineering (DAEN) program.

He joined the DAEN faculty in the Fall of 2020 from Texas A&M University-Commerce (TAMUC) where he served as an Assistant Professor of Computer Science as well as the department’s Outreach Coordinator. Before coming to TAMUC, Dr. Gang was an Assistant Professor of Computer Science and Engineering at the University of Mary Hardin-Baylor (UMHB) and an Adjunct Professor of Computer Science at the University of Southern Mississippi’s School of Computing before joining UMHB.

Dr. Gang is a former DOE grant winner, former President and Board Member of the Association of Computer Educators in Texas (ACET), Industry Advisory Board (IAB) Coordinator, and the Director of CS For All.

His current and primary teaching responsibilities at Mason largely involves Data Analytics Engineering graduate courses along with a mix of CS and AIT graduate courses. He is an affiliate faculty member of GMU’s C4I & Cyber Center.

Dr. Gang’s primary research agenda involves Big Data Analytics (emphasis on data bias and data governance), Cyber Security (ransomware, steganography, and cyberbullying), and Image/Signal Processing.

Contents

Table of Contents

[Abstract 5](#_Toc125979012)

[Section 1: Problem Definition 7](#_Toc125979013)

[1.1 Background 7](#_Toc125979014)

[1.2 Problem Space 7](#_Toc125979015)

[1.3 Research 7](#_Toc125979016)

[1.4 Solution Space 8](#_Toc125979017)

[1.5 Project Objectives 8](#_Toc125979018)

[1.6 Primary User Stories 8](#_Toc125979019)

[1.7 Product Vision 9](#_Toc125979020)

[1.7.1 Scenario #1 9](#_Toc125979021)

[1.7.2 Scenario #2 9](#_Toc125979022)

[Section 2: Datasets 9](#_Toc125979023)

[2.1 Overview 9](#_Toc125979024)

[2.2 Field Descriptions 9](#_Toc125979025)

[2.3 Data Context 10](#_Toc125979026)

[2.4 Data Conditioning 11](#_Toc125979027)

[2.5 Data Quality Assessment 11](#_Toc125979028)

[2.6 Other Data Sources 11](#_Toc125979029)

[2.7 Storage Medium 11](#_Toc125979030)

[2.8 Storage Security 11](#_Toc125979031)

[2.9 Storage Costs 12](#_Toc125979032)

[Section 3: Algorithms & Analysis / ML Model Exploration & Selection 12](#_Toc125979033)

[3.1 Solution Approach 12](#_Toc125979034)

[3.1.1 Systems Architecture 12](#_Toc125979035)

[3.1.2 Systems Security 12](#_Toc125979036)

[3.1.3 Systems Data Flows 12](#_Toc125979037)

[3.1.4 Algorithms & Analysis 12](#_Toc125979038)

[3.2 Machine Learning 12](#_Toc125979039)

[3.2.1 Model Exploration 13](#_Toc125979040)

[3.2.2 Model Selection 13](#_Toc125979041)

[Section 4: Visualizations / ML Model Training, Evaluation, & Validation 13](#_Toc125979042)

[4.1 Overview 13](#_Toc125979043)

[4.2 Visualizations 13](#_Toc125979044)

[4.3 Machine Learning 13](#_Toc125979045)

[4.3.1 Model Training 13](#_Toc125979046)

[4.3.2 Model Evaluation 14](#_Toc125979047)

[4.3.3 Model Validation 14](#_Toc125979048)

[Section 5: Findings 14](#_Toc125979049)

[Section 6: Summary 14](#_Toc125979050)

[Section 7: Future Work 14](#_Toc125979051)

[Appendix A: Glossary 15](#_Toc125979052)

[Appendix B: GitHub Repository 16](#_Toc125979053)

[Overview 16](#_Toc125979054)

[GitHub Repository Link 16](#_Toc125979055)

[GitHub Repository Contents 16](#_Toc125979056)

[Appendix C: Risks 17](#_Toc125979057)

[Sprint 1 Risks 17](#_Toc125979058)

[Sprint 2 Risks 17](#_Toc125979059)

[Sprint 3 Risks 17](#_Toc125979060)

[Sprint 4 Risks 17](#_Toc125979061)

[Sprint 5 Risks 18](#_Toc125979062)

[Appendix D: Agile Development 19](#_Toc125979063)

[Scrum Methodology 19](#_Toc125979064)

[Sprint 1 Analysis 19](#_Toc125979065)

[Sprint 2 Analyis 19](#_Toc125979066)

[Sprint 3 Analysis 20](#_Toc125979067)

[Sprint 4 Analysis 20](#_Toc125979068)

[Sprint 5 Analysis 20](#_Toc125979069)

[Works Cited 21](#_Toc125979070)

**No table of figures entries found.**

**This page intentionally left blank**

Abstract

Abstract

**INSTRUCTIONS**

[NOTE: The project abstract is a separately graded assignment in the course. The final approved project abstract is to be copied word-for-word from the other assignment into this report.]

Write one paragraph of no more than 300 words that summarizes your project. Here are the typical kinds of information found in most abstracts which you should use as an outline as you develop your abstract.

1. The context or background information for your research; the general topic under study; the specific topic of your research.
2. The central questions or statement of the problem your research addresses.
3. What’s already known about this question, what previous research was conducted or shown.
4. The main reason(s), the exigency, the rationale, the goals for your research — why is it important to address these questions? Are you, for example, examining a new topic? Why is that topic worth examining? Are you filling a gap in previous research? Applying new methods to take a fresh look at existing ideas or data? Resolving a dispute within the literature in your field?
5. Your research and/or analytical methods.
6. Your main findings, results, or arguments.
7. The significance or implications of your findings or arguments.

Your abstract should be intelligible on its own, without a reader’s having to read your entire paper.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

**This page intentionally left blank**

Report

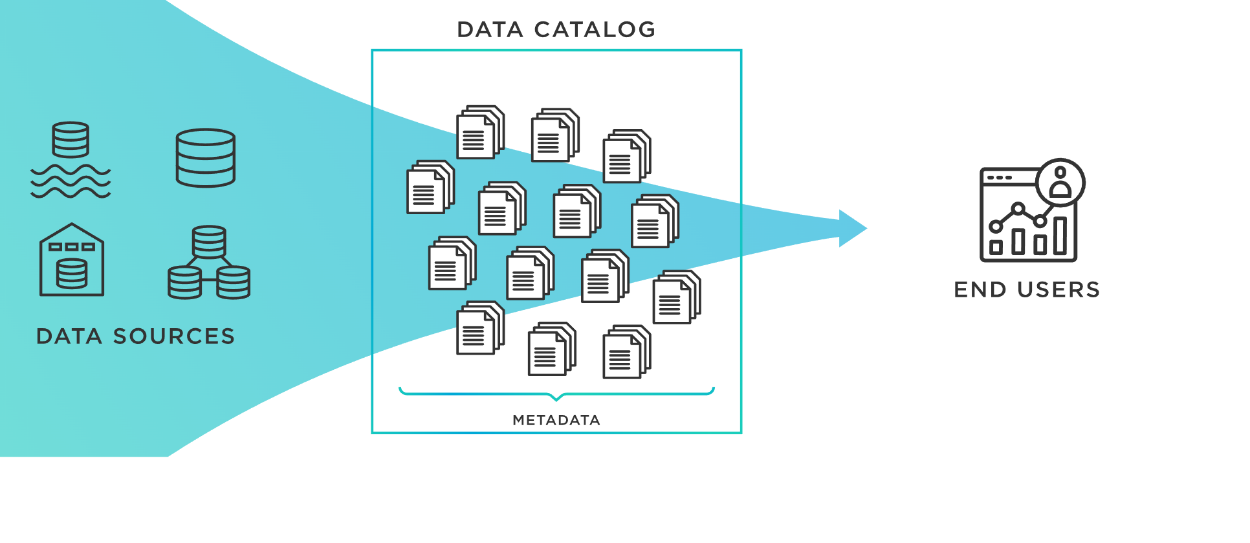
# Problem Definition

## Background

As technologies grew, we saw the huge demand in data management and exploration. As per today's figure 2.5 quintillion data is generated every day and searching in this big amount of data could be difficult and cumbersome. some of the popular old methods which were used in searching the data were.[1]

* Identify Key Data Sources: Identifying the primary data sources is the first stage in a search for data without data catalogs. Understanding the many tools and systems used by the company to create, store, and process data is necessary for this. Knowing the location and structure of data sources will help you focus your search and find information in particular regions.
* Utilize Search Engines: Many businesses use search engines like Google or Bing to look up information online. Search engines can also be used to look for information inside your company. You may use keyword searches on shared drives, email inboxes, and collaborative tools to find information. It's crucial to keep in mind that this strategy might not produce reliable results, particularly when looking for private or sensitive information.
* Collaborate with Colleagues: Working with colleagues is an additional method for finding data without data catalogs. Colleagues who have been with the company for a while can be well-versed in where the data is kept. Also, locating and accessing data contained in databases can be made easier by collaborating with coworkers who have specialized expertise, such as database administrators.
* Analyze Application Logs: Any application can produce logs to keep track of user behavior. To determine how data is used within the company, these logs can be examined. You can determine which users have accessed which data sources and when by looking through the application logs. This method can assist you in finding information sources that you might have overlooked during your search.

All these search methods help you to find data but are much slower and sometime not reliable. To increase the speed and accurately find the data we need to work on much faster methods which is searching the data using data catalogs. Data catalogs are useful in search because they provide a centralized repository of information about the available data assets within an organization or system. By using a data catalog, users can quickly and easily search for data assets based on various criteria such as data type, metadata, keywords, and data lineage.



Features like faceted search and best fit tags help in increasing the efficiency of the data catalogs. The "best fit tags" feature in data catalogs is a way to automatically suggest relevant tags or labels for data based on its content and metadata. The data is examined to determine its essential characteristics, such as the data type, format, and subject matter. Next, depending on a match between these traits and an already-existing taxonomy of tags or labels, it recommends the most acceptable ones.

For instance, the "best fit tags" tool can recommend tags like "sales," "customer data," "revenue," and "sales performance" if you submit a dataset containing customer sales data. Thus, based on their unique requirements, users can discover and filter the data using these tags. The "best fit tags" function may have the ability to speed up the data cataloging process, which is one potential advantage. The system can automatically recommend pertinent tags based on the data itself, saving users the effort and risk of error that comes with having to manually tag their data. This can make sure that the data is consistently categorized and labeled, which will eventually make it simpler to find and use. It's important to remember that the "best fit tags" function is not perfect. The system might not always recommend the most precise or pertinent tags, depending on the complexity of the material and the quality of the metadata. Nonetheless, data catalogs can assist in making their data more accessible and helpful to a larger range of consumers by continuously enhancing the algorithms used to recommend tags.[2]

We have another feature which is helpful, search facets are categories or attributes that are used to categorize data and make it easier to find and search for. Users can refine and narrow down their search results to a more focused set of information that matches their criteria by applying these facets to their search results as filters.

Let's take the scenario where you are searching for a specific dataset in a data catalog that contains data on patterns of the world's climate. You can use the data catalog's search facets to filter your search results and locate the data you need. There may be various search aspects, some of which include:

* Data Type: You might be able to restrict your search results using this aspect based on different sorts of data, such text, photographs, or videos.
* Date Range: This facet may allow you to filter your search results by a specified time period, such as the past year, past 5 years, or a custom date range.
* File Format: With this feature, you might be able to filter your search results based on the types of data files they contain, such as CSV, Excel, or JSON.
* Author: With this facet, you might be able to narrow down your search results by the person or organization who created the dataset, for example.

By using these search Facet, we can swiftly narrow down our search results to find the dataset you require. For instance, we may restrict the datasets that appear in our search results to those that were produced within the last year, are in CSV format, and were created by a specific researcher.

Overall, search facets are a potent technique for improving the effectiveness and usability of data libraries. Search facets give users the ability to filter and hone their search results, which can save time and boost productivity by enabling users to get the information they need quickly and simply.[3]

## Problem Space

Here we will talk about three problem which we could encounter while integrating data catalogs with content search technologies are.[4]

• Data Consistency: Ensuring data consistency across the data catalog and the content search technology is one area of concern. While the content search technology indexes the actual data content, the data catalog may include metadata that specifies the data's characteristics, such as its kind, size, and format. It is crucial to check that the metadata in the catalog matches the content that has been indexed by the search technology. Consistency issues could result in erroneous search results, which could affect data retrieval and discovery.

• Scalability: Ensuring scalability when combining a data catalog with content search technology is another area of concern. There are more catalog items and search indexes as data sets grow and complexity. This may influence the search technology's performance, slowing down search requests or even causing system failures. Designing a scalable architecture that can manage growing data volumes while retaining search performance is crucial.

• Data Security: The incorporation of content search technologies with a data catalog can raise security issues. Sensitive information, including data location, access restrictions, and permissions, may be present in the data catalog. Technology for content search may also index confidential information. Making sure that the data catalog and search technology are secure and that only authorized people have access is crucial. This may entail putting in place safe authentication and permission procedures, encrypting confidential information, and keeping an eye on access logs for illegal access attempts.

## Research

During our research we came across came widely popular data catalogs which are being used in real word with various other popular tools:

* **AWS Glue Data Catalog:** the AWS Glue Data Catalog is a fully managed metadata repository that stores and manages data about data assets like databases, tables, and their schemas. It offers a consolidated and uniform method of storing metadata across numerous AWS data sources, simplifying data management and processing. By offering a consistent view of all data assets, the AWS Glue Data Catalog enables users to find and comprehend the data available in various data sources, including databases, Amazon S3, Amazon RDS, and Amazon Redshift. It is intended to be extremely performant, scalable, and available, and it supports concurrent access from various people and systems.

The data catalog offers a schema registry for various data sources, assisting in ensuring data accuracy and consistency across various AWS services. Names of databases, tables, columns, data types, and other features that describe the data are among the metadata included in the catalog. Users can execute tasks including creating and updating tables, searching metadata, and controlling access by gaining access to the Amazon Glue Data Catalog through the AWS Management Console, the AWS CLI (Command Line Interface), or the AWS SDKs (Software Development Kits). The Amazon Glue ETL (Extract, Transform, Load) service, which enables users to transform and transport data across different data stores, can be used in conjunction with the AWS Glue Data Catalog. It is simpler to manage complex data transformations and migrations with Amazon Glue ETL because it automatically generates ETL code using the metadata kept in the Data Catalog. Overall, AWS Glue Data Catalog is an effective solution that can assist businesses in effectively managing metadata across various data sources and services, facilitating data understanding, access, and processing.[5]

* **Google Cloud Data Catalog:** A fully managed and scalable metadata management service, Google Cloud Data Catalog enables businesses to quickly find, comprehend, and manage their data assets across the Google Cloud Platform (GCP), as well as other cloud and on-premises settings. Google data Catalog enables users to swiftly search and discover data assets based on their qualities, keywords, and tags. It also helps users to develop and manage an exhaustive and up-to-date inventory of data assets, including data stores, tables, views, schemas, and metadata. Some of the popular features of Google cloud data catalogs are:
  + Centralized Metadata Management: Users can quickly find and comprehend their data assets thanks to Data Catalog, which offers a consistent view of metadata from diverse data sources across GCP and other cloud and on-premises environments.
  + Automated Data Discovery: Data Catalog makes it simple for users to locate the data they require by automatically scanning data sources and classifying and tagging data assets based on their characteristics and usage patterns.
  + Collaborative Data Governance: By enabling fine-grained access control and enabling users to develop, share, and reuse tags, policies, and annotations, Data Catalog enables users to work together on metadata management.
  + Integration with GCP Services: Data Catalog is tightly integrated with GCP services such as Big Query, Cloud Storage, and Pub/Sub, enabling users to quickly access and use their data assets within GCP.[6]

## 1.3.1 Project Tools

The tools being used for this project are Momentum in conjunction with Elasticsearch and OpenMetadata

* **Momentum:** Momentum Tool by Accure is a no-code data engineering and AI/ML platform that enables engineers, scientists, analysts, and automation engineers to efficiently solve machine learning problems and automate business processes. It offers a drag-and-drop user interface that enables users to create and use ML models without having to know any programming or code. A/B testing, model version management, data drift detection, and other tools that assist users in managing and keeping an eye on their ML models are also included in Momentum.

Engineering features automatically are done using Momentum AI. Data scientists can maintain their focus on raising model accuracy thanks to this. Box charting, Pearson's Chi-squared, and correlation coefficients are other tools you can use for feature engineering. You could use SMOTE to generate synthetic data to balance your classes if your dataset contains unbalanced classes and down sampling will result in data losses. Momentum offers a highly scalable SMOTE implementation that can handle billions of rows of data.[7]

* **Open Metadata:** OpenMetadata is an open-source metadata management system that aids businesses in the management of metadata throughout their data and analytics ecosystems. Data utilization, data lineage, and data quality are all examples of metadata, which is information about data. With the help of OpenMetadata, businesses can manage and store metadata from a variety of sources, including databases, data warehouses, and data lakes. Additionally, it offers a collection of APIs and connectors that make it simple for programmers and data engineers to access and use information in their processes and applications. Via the provision of visibility into data lineage and data quality, one of the main advantages of OpenMetadata is that it aids companies in ensuring the reliability of their data. This can aid businesses in making wiser decisions, enhancing data governance, and fostering more teamwork.

OpenMetadata can be used by organizations of all sizes and in all industries. It is a valuable tool for organizations that are looking to improve their data management practices. Here are some of the benefits of using OpenMetadata:

* + Improved data discovery: All data assets, including their connections to one another, may be found in one location thanks to OpenMetadata. Users will find it simpler to locate the data they require as a result.
  + Improved data governance: Tools for data governance are available from OpenMetadata, including data lineage and data quality management. This facilitates better data management for enterprises.
  + Improved data quality: Tools for managing data quality, including data profiling and data cleaning, are provided by OpenMetadata. This aids businesses in raising the caliber of their data. [8]
* **Elastic Search:** The Apache Lucene search engine library provides the foundation for the distributed, open-source search and analytics engine Elasticsearch. It offers a scalable search solution with quick and effective search capabilities and is built to handle enormous volumes of data. For search capabilities as well as for logging, monitoring, and analytics, Elasticsearch is frequently utilized as a key element in an application's design. Structured, unstructured, and geographic data types can all be stored, indexed, and searched for using Elasticsearch. For doing complex search queries, aggregations, and analytics, it offers a potent collection of APIs. Elasticsearch is incredibly flexible and can be tailored to meet the unique requirements of a given application or business.

Elasticsearch can be set up on a group of computers to offer fault tolerance and high availability. It can be used independently or combined with other application stack components like Kibana, Logstash, and Beats, which are a part of the Elastic Stack.[9]

## Solution Space

Integrating data catalogs with content search technologies can provide organizations with a powerful tool for discovering, understanding, and using their data. It can improve the search functionality of data catalogs and help users find the information they need quickly and accurately. Another benefit of integrating data catalogs with content search technologies is that it can improve the accuracy of search results. The metadata in a data catalog can be used to disambiguate terms, understand the context of a search, and provide more relevant results.

Data Consistency: will be Creating a common data model to guarantee consistency between the content search technology and the data catalog. Determining metadata attributes and how they relate to the content might be a part of this. To ensure metadata uniformity across the data catalog and search technology, using automated techniques. To find any discrepancies, this may entail comparing metadata attributes, data types, and format kinds. And doing data quality tests to find and fix discrepancies between the metadata in the catalog and the content that has been indexed by the search technology.

Scalability: we will use a distributed architecture that manages enormous data volumes and splits the burden across several nodes. Using a distributed file system for storing and a distributed search index for indexing and querying are two examples of how to do this:

1. Use indexing and caching strategies to enhance search performance. To decrease the number of requests made to the search technology, this may entail pre-fetching data and caching search results.
2. To enhance search performance and lower the chance of system crashes, distribute search queries among several search nodes using load balancing techniques.

Data Security: Security measures should be used to ensure that only authorized users can access the data catalog and search technologies. And protecting sensitive data in the data catalog and search technologies by using encryption techniques.

## Project Objectives

The objectives of the project are divided into 3 sub-objectives. Like straightforward keywords, tags assist users in organizing the data objects. By categorizing the data and functionally organizing it, users may make it easier to discover. Our first task is to determine the best-fit tags. Best fit tags help with search optimization, and this could be achieved by feeding the metadata tags to search engines and inputting the metadata tags into data catalogs. The resulting output would be a bunch of data tags and finding the best descriptor among them is the goal.

The next task is to integrate the search engine and data catalogs. By joining them, the distinct features of both data catalogs and the search engine could be leveraged to our advantage.

The final objective is to provide access indicators to the data. This would control access to data and increase data privacy. The access indication could be achieved by mimicking a data catalog and access rules. The access rules that would be included for this project are ABAC, RBAC & XACML.

By utilizing policies to specify what may (or cannot) happen, ABAC enables you to create fine-grained access based on any attribute (not just role and not just user attributes). Role-based security sometimes referred to as role-based access control (RBAC), is a method that limits system access. Setting rights and permissions is necessary to grant authorized user's access. Finally, Extensible Access Control Markup Language is an XML-based attribute-based access control policy language intended for expressing security policies and information access requests. These are the access control languages that are included in restricting access in the project.

## Primary User Stories

We created the following major user story to serve as our project's guideline based on the user context and value proposition:

* **Incorporating faceted search Feature:**

It is a search interface that allows users to filter results based on multiple criteria, also known as facets, such as categories, tags, date ranges, and others. This type of search can help in integrating data catalogs with content search technologies by providing users with a more intuitive and flexible way of accessing and discovering information.[10]

By incorporating facets into a search interface, users can quickly narrow down the results to a specific subset that meets their criteria, reducing the amount of time they spend searching and increasing the chances that they'll find what they're looking for. The integration of data catalogs and content search technologies can also provide a more complete view of the information that is available, making it easier for users to understand the context of the information and how it relates to other data sets. Additionally, the use of facets can help improve the accuracy and relevance of search results, since it allows users to refine their searches based on specific criteria. This can also lead to a better understanding of user behavior and preferences, which can be used to further improve the search experience.

* **Incorporating Search suggestions Feature:**

Search suggestions can help integrate data catalogs with content search technologies by providing users with a more intuitive and efficient search experience. When a user starts typing a query into a search bar, search suggestions can appear in real-time and provide relevant terms, phrases, or questions related to the user's search. This can save users time and effort in finding the information they need.[11]

The search suggestions can be based on the information contained within the data catalog. This can help ensure that the search suggestions are relevant to the user's needs and can lead them to the right information. The integration of the data catalog with content search technologies can also provide better search results by combining structured data from the catalog with unstructured content. Also, search suggestions can help improve the data catalog quality by providing feedback on what users are searching for. This can help organizations to identify gaps in their data catalogs and make updates to ensure that the catalog provides the information that users need.

## Product Vision

Our product vision is to create a unified data discovery platform that empowers users to find and utilize relevant data assets quickly and easily. Our platform will leverage machine learning algorithms to analyze the content of datasets, making it easier for users to search for data based on keywords, concepts, or context. It will help organizations to be more efficient, productive, and innovative, by providing them with the data insights they need to make informed decisions.

**For:** MITRE Corporation

**Who:**  Whoever wants to get useful insights from the dataset

**Our Product:** Data Catalog Integrated with Content Search Engine Technology

### Scenario #1

Improved search result: By giving the search engine context, integrating a data catalog with content search technology can dramatically enhance the search results. A data catalog can aid a search engine's comprehension of the data's information, such as the types, sources, and relationships, enabling the search engine to produce more accurate and pertinent results. Consider a scenario where a user is looking for a client's phone number in a database that contains information about the customer in many tables. If so, the search engine may determine which table has the customer data using the data catalog, which will help it focus its search results.

### Scenario #2

Data Governance: Additionally, it can aid organizations in enforcing data governance guidelines. The lineage, quality, and access controls of data assets can all be managed centrally through a data catalog. Users can only search for data that they are authorized to access by integrating the data catalog with content search tools. Consider a scenario where a company has a policy that limits access to confidential customer information. The content search engine may then make sure that only authorized users can look for such data by identifying which data sources include sensitive customer data.

# Datasets

## Overview

This section briefly discusses the dataset that we used and where we acquired the dataset from. Our Primary dataset is Synthea Covid 19 Dataset. The dataset is in text format.

|  |  |  |  |
| --- | --- | --- | --- |
| Dataset Title | Source | Size | Type |
| MITRE Synthetic Health Data | MITRE | 21 GB | Text |

MITRE Synthetic Health Data is an open-source patient population simulation made accessible by The MITRE Corporation, is the source of the data stored within Synthetic Mass. This dataset consists of ten thousand synthetic patient’s records based out in Massachusetts with COVID-19 in the CSV format.[18]. The Synthea COVID-19 data set is a longitudinal set of synthetic COVID-19 patients and their EHR records. This data inventory has different files that contain patient personal information, allergies, immunizations, payers, care plans, observations etc.

## Field Descriptions

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Title | URL | Authors | Publication Date: | Type | Tags |
| MITRE Synthetic Health Data | <https://synthea.mitre.org/downloads> | MITRE | 11/01/2020 | Text | “Health", "Covid” |

We used 5 csv files from the MITRE Synthetic Covid 100k health dataset. The “Patient.csv”, “Observations.csv”, “Encounter.csv”, “Procedures.csv” and “Careplans.csv” were merged into one csv file with around 49 columns.

### Patient Dataset

The Patients dataset is a list of patients, as well as their healthcare expenses and coverage. Each patient will have a row with their personal information. It has 25 fields, with the patient ID displaying the unique as stated. This collection contains PII data for patients in the form of string, float 64, and integer data types.

|  |  |  |
| --- | --- | --- |
| Field Name | Description | Data Type |
| ID | This is the unique patient ID that is used to identify patients and patient information. This ID is initially generated when a patient is admitted, or the record is made. | String |
| BIRTHDATE | This is the birth date of the patient. The format is mm/dd/yyyy. | Date |
| SSN | This is the social security number of the patient. This field does not have null. | Number |
| DRIVERS | This is the driver's license of the patient. It is a specific identification number assigned to a driver by the issuing government agency. It has 9 characters. | String |
| PASSPORT | This is the passport number of the patient. It is a unique identification number, and it has null values for the patient who does not have a passport number. | String |
| FIRST | This is the first name of the patient. | String |
| LAST | This is the last name of the patient. | String |
| MARITAL | This is the marital status of the patient. It is ‘M’ for married, ‘S’ for Single. It has blank fields for patients whose marital status is unknown. | Character |
| RACE | This is the patient’s race. It could be Black, White, Asian, Native etc. | String |
| ETHNICITY | This is the patient’s ethnicity. This indicates if the patient is Hispanic or non-Hispanic. This field does not have null values. | String |
| GENDER | This field indicates the patient’s gender. This field either has ‘M’ for Male or ‘F’ for Female. | Character |
| BIRTHPLACE | This field has the patient’s birthplace. | String |
| ADDRESS | This field has the exact place where the patients are living. | String |
| CITY | This field has city where patient is living | String |
| STATE | This field is the state where the patients are living. | String |
| COUNTY | This field is the County where the patients are living. | String |
| ZIP | This field is the ZIP code (postal code) for the place where the patients are living. | String |
| LAT | This field is the longitude of the place where the patients are living. | Float64 |
| LON | This field is the latitude of the place where the patients are living. | Float64 |
| HEALTHCARE\_EXPENSES | This field contains the total expenses of the individual patient in dollars. | Float64 |

### Procedure Dataset

|  |  |  |
| --- | --- | --- |
| Field Name | Description | Data Type |
| DATE | This field is the start date of the procedure for the patient | Date |
| PATIENT | This is the unique patient ID that is used to identify patients | String |
| ENCOUNTER | This field is a unique ID for the encounter | String |
| CODE | This field gives the code for the patient. | Number |
| DESCRIPTION | This field gives the description about the patient's condition | String |
| BASE\_COST | This field gives base cost for the patient’s procedure. | Number |
| REASONCODE | This field gives reason code for the patient’s procedure. | Number |
| REASONDESCRIPTION | This field gives description about the reason. | String |

### CarePlan’s Dataset

Care Plan dataset gives details about the patient’s care Plan with its start and end date.

|  |  |  |
| --- | --- | --- |
| Field Name | Description | Data Type |
| ID | This is the unique patient ID used to identify patients and patient information. | String |
| START | This is the Start date of the patient’s CarePlan | Date |
| STOP | This is the Stop date of the patient’s CarePlan | Number |
| PATIENT | This is the unique patient ID that is used to identify patients | String |
| ENCOUNTER | This field is a unique ID for the encounter It can be null | String |
| CODE | This field gives the code for the CarePlan | Number |
| DESCRIPTION | This field gives the description about the patient’s CarePlan | String |

### Observation Dataset

|  |  |  |
| --- | --- | --- |
| Field Name | Description | Data Type |
| DATE | This field gives the date of the observation. | Date |
| PATIENT | This is the unique patient ID that is used to identify patients | String |
| ENCOUNTER | This field is a unique ID for the encounter It can be null | String |
| CODE | This field gives the code for the Patient | Number |
| DESCRIPTION | This field gives the description about the patient’s Observation | String |

### Encounter’s Dataset

|  |  |  |
| --- | --- | --- |
| Field Name | Description | Data Type |
| ID | This is the unique patient ID that is used to identify patient's encounters. | String |
| START | This is the Start date of the patient’s Encounter. | Date |
| STOP | This is the Stop date of the patient’s Encounter. | Date |
| PATIENT | This is the unique patient ID that is used to identify patients. | String |
| ORGANIZATION | This field gives ID about the organization | String |
| PROVIDER | This field gives the code for the Patient’s encounter provider | String |
| PAYER | This field gives the ID of the Payer | String |
| ENCOUNTERCLASS | This field gives the description about the Encounters class | String |
| CODE | This field gives the code for the encounter | Number |
| DESCRIPTION | This field gives the details about the encounter’s description | String |
| BASE\_ENCOUNTER\_COST | This field gives the cost about the encounters. | Number |
| TOTAL\_CLAIM\_COST | This field gives the total claim cost of the encounter | Number |
| PAYER\_COVERAGE | This field gives the cost covered by the payer | Number |
| REASONCODE | This field gives reason code for the encounter. | Number |
| REASONDESCRIPTION | This field gives description about the reason. | String |

## Data Context

The main idea of the project is to integrate the best fit tags, tracking tags, scalability, and security features of the search engine elastic search to the data catalog open metadata. In this project, the primary datasets we used are MITRE Synthetic Health Data, sourced from MITRE.

The team has decided to use the above dataset for two primary reasons. Since the questions requested for a search engine are text-based, the above datasets are more relevant for this project. The features are text based, which are like search engine queries, and they allow us to validate the best fit tags and tracking tags for the dataset. To make the dataset searchable, it would need to be indexed and labeled with metadata. Adding categories depending on the themes covered in each document, as well as metadata such as the author, publication date, and source of the text excerpt, might be included. Lastly, using a search engine such as Elasticsearch, the dataset might be integrated into a data catalog. This would allow users to search the dataset for relevant documents based on keywords, tags, or other metadata.

## Data Conditioning

Our Primary dataset Synthea Covid 19 Dataset is a non-realistic but synthetic data repository from Synthea (MITRE) generated and maintained as an open source. The repository contains data that has been maintained in various files based on their usage such as patients, allergies, medications, and observations etc. For project, the data gathered must be refined and processed in a way that can be used better with data catalog and search engine.

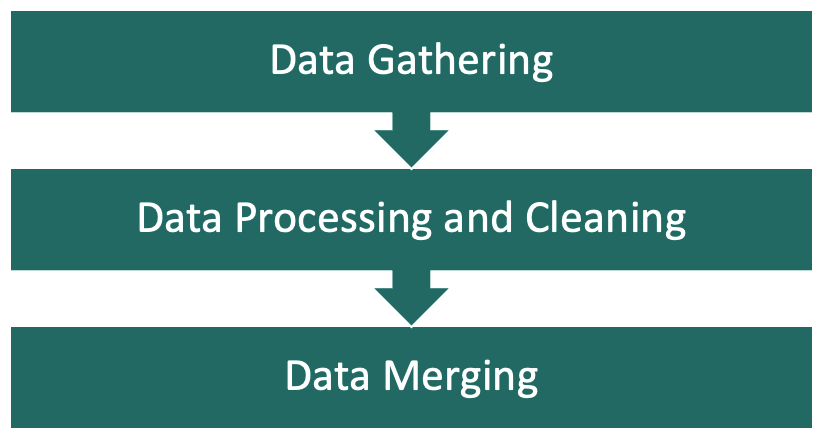


Fig 2.  Data Conditioning Process for Synthea COVID-19 Dataset

From a total of 16 datasets from the repository having more than 100k records, the team has shortlisted a few datasets which are Encounters, Organizations, Patients and Conditions and planned to work further on these datasets to form a single dataset which has a combination of records from all the 4 chosen datasets. The team planned to use Py Spark and SQL in Databricks to handle these files and perform join Operations to get a final dataset which will be used for tasks and operations performed in the future. While considering the idea of creating a new dataset with subsets of data, the team had assumed and identified to work on some features in data such as COVID-19 as a health condition, information of patients who were diagnosed with COVID-19 and organizations where the patients were admitted or treated. The new dataset with identified features will be used to feed the data catalog, generate tags and work on accesses part of this project.

Data Conditioning for BBC News dataset

The BBC news dataset is an open-source self-updating dataset. The dataset is updated automatically through a kernel that functions with fixed frequency and the output of the kernel updates the dataset. The RSS news feed is collected from the BBC news official news site. We are going to be using this dataset for integration with the elastic search engine. This is a single dataset of 4.41 MB. Since this is a small dataset there is not much data conditioning required. The dataset does not have any missing values or skewed values therefore it is good enough for achieving the project objective. Using this dataset, the team aims to generate metadata tags that determine the sentiment of the news. We are going to achieve generating tags using Open Metadata and integrate them with the search engine (elastic search).

## Data Quality Assessment

### Completeness:

The first data set in consideration for this project is Covid 100k Synthetic data generated by MITRE Organization. This data set contains the data in csv files related to Procedure, Observations, Care plans, Encounter and Patient The measure of Completeness was manually calculated from the merged dataset and except for the patient ID column all the columns had some missing values. For this project we are considering using the data sets conditions, encounters. In the dataset file conditions, we are filtering out data based on the code 840539006 as it is the data related to the covid file. There are no null values available in the rest of the fields. The encounters data set is not complete as it has values missing in several fields in reason code and reason description. The second dataset that we considered was BBC News dataset. This dataset is complete and does not have any null values.

### Missing Data:

In the MITRE merged dataset only Patient ID column does not have any missing data rest all the columns have some missing value. The count of missing value for each column is shown in Fig 3. There was no missing data observed in the datasets BBC News.

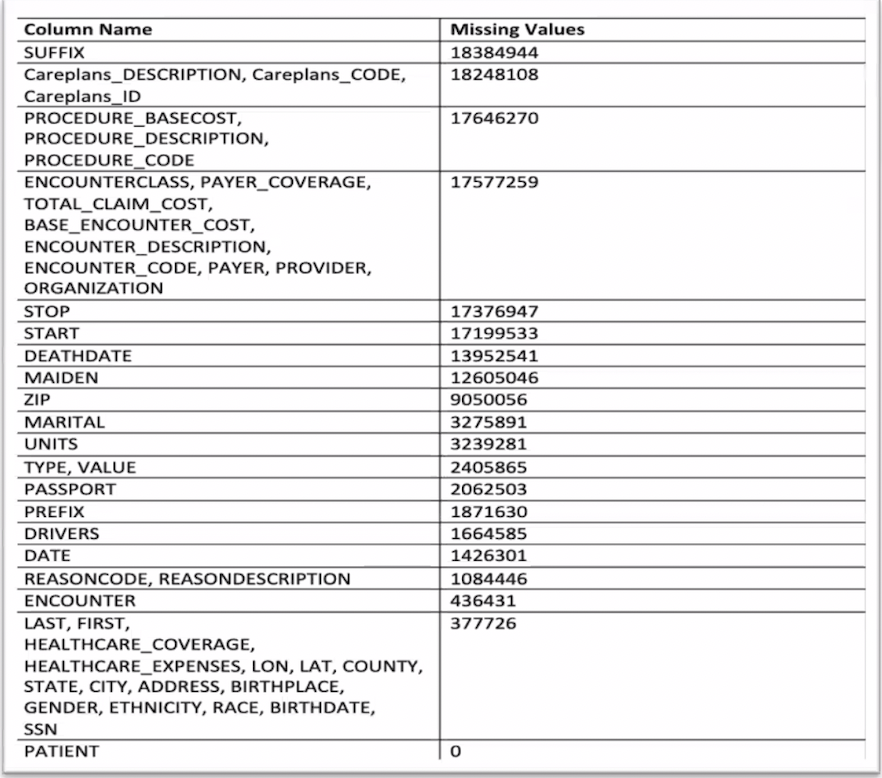


Fig 3. Missing value count for MITRE dataset.

### **Uniqueness:**

For the MITRE Dataset, many columns in the Merged dataset have numerous unique values. On the other hand, other columns, including ENCOUNTERCLASS, RACE, SUFFIX, PREFIX, ETHNICITY, GENDER, BASE ENCOUNTER COST, TYPE, TOTAL CLAIM COST, MARITAL, and STATE, contain a small number of distinct values, suggesting that some of them represent discrete categorical variables.

The number of unique values for the BBC news is as follows. The tile field contains 13539 values. Pub Date stands for publishing date which contains the date values between 5Sep17 and 25Feb23. The field guide contains 12947 unique values. The RSS Feed description contains 13321 unique values. There are several repeated values for patient, providers, code, description in the encounters data set as the medical data is recorded multiple times. However, the Id column is unique. In the conditions data set the data set is unique and there are no repeated values.

### **Accuracy:**

Most of the data seems to be accurate except for the field guide in BBC news which is URL provided for the specific article. Some articles might be missing because of the extensive timeline of 5 years from 2017 to 2023. As the data generated for the covid data is synthetic, the data sets encounters and covid data set is accurate.

### **Atomicity:**

The level of atomicity is very high in the MITRE Merged dataset because most of its columns cannot be further decomposed into smaller attributes and the atomicity of the data is assessed as high for and BBC news data set as there are no fields missing.

### **Conformity:**

Most of the dataset's properties have uniform formatting, ensuring the data's consistency and standardization. However, a few columns have data in text and numbers which needs to be processed.

### **Overall Quality:**

The Overall quality of datasets BBC news is good as they do not have much missing values and most values provided are within the context of data. The overall data quality of the dataset conditions and encounters is moderate as some of the columns have missing values and the missing reason has not been justified.

## Other Data Sources

### BBC News Dataset

BBC News dataset is available on Kaggle, a collection of 2,225 news articles published by the BBC between 2004 and 2005. The dataset includes news articles in five distinct categories, including business, entertainment, politics, sport, and tech. The data is in CSV format and can be used for a variety of natural language processing tasks, such as text classification, sentiment analysis, and topic modeling. Additionally, the dataset is small, which makes it an excellent choice for training and testing models quickly. Each article in the collection includes a title, a brief synopsis, and the article's full content. The articles were published between 2004 and 2005 and are identified by their category. The dataset can be used for text categorization, sentiment analysis, and topic modeling, among other natural language processing applications.

### Ask a manager 2021 Salary Survey Dataset

Ask A Manager is an advice website for workplace-related questions. Earlier in 2021, the blog site published a Manager Salary Survey where respondents anonymously shared their current salary among other biographical details. The survey and the resultant data are interesting in several ways. One reason is that the responses are anonymous, hence there is a minimal risk of exposing participants to personal information. Also, the dataset is large (N=26538) and publicly available.

## Storage Medium

For this project we will are considering using the amazon s3 bucket. We will store the data files in S3 bucket which will act as an initial storage for our data. We will also be using snow pipe to integrate snowflake which will be useful in storing and fetching data from S3 bucket to the snowflake database, and S3 bucket also provides us with the functionality where we can use S3 events to trigger actions based on changes to the data files. Enormous amounts of data can be handled with ease by Amazon S3, which is very scalable. This makes it perfect for tasks involving the management and storage of massive amounts of data because Amazon S3 is built for 99.99% durability, our data will always be accessible whenever we need it and will be highly resilient to failures. For projects that demand high availability and reliability, this is essential. A cost-effective storage option is Amazon S3, which charges according to the volume of data kept and the level of access needed. It is therefore a cost-effective choice for undertakings with tight budgets.

Amazon S3 integrates seamlessly with other AWS (Amazon Web Services) services, such as Amazon Elasticsearch, Amazon Athena etc. making it easy to build a complete data storage and search solution.

## Storage Security

Amazon S3 provides a variety of security features and controls to help you protect your data stored in the service.

We have access to several tools through Amazon S3, including Identity and Access Management (IAM) rules, Bucket policies, Access Control Lists (ACLs), and pre-signed URLs (Uniform Resource Locator), to manage who has access to our data. We may also impose fine-grained access controls on individual items or buckets using these protocols, as well as give or revoke permissions to individuals, groups, or roles.

We have a lot of options with Amazon S3 to manage network-level access to our data. We can use Amazon S3 Block Public Access to ensure that S3 buckets or objects are not publicly accessible, create Access Points to enable network regulations and routing rules, or use Virtual Private Cloud (VPC) to isolate our S3 resources.

## Storage Costs

Storing data in your S3 buckets costs money. The price you pay is determined by the size, duration, and storage class of your items and how often you access them during the month. To track access patterns and automate the movement of objects between access tiers, you must pay a monthly monitoring and automation fee per item stored in the S3 Intelligent-Tiering storage class. There are no retrieval fees in S3 Intelligent-Tiering, and there are also no additional tiering fees when items are moved across access levels.

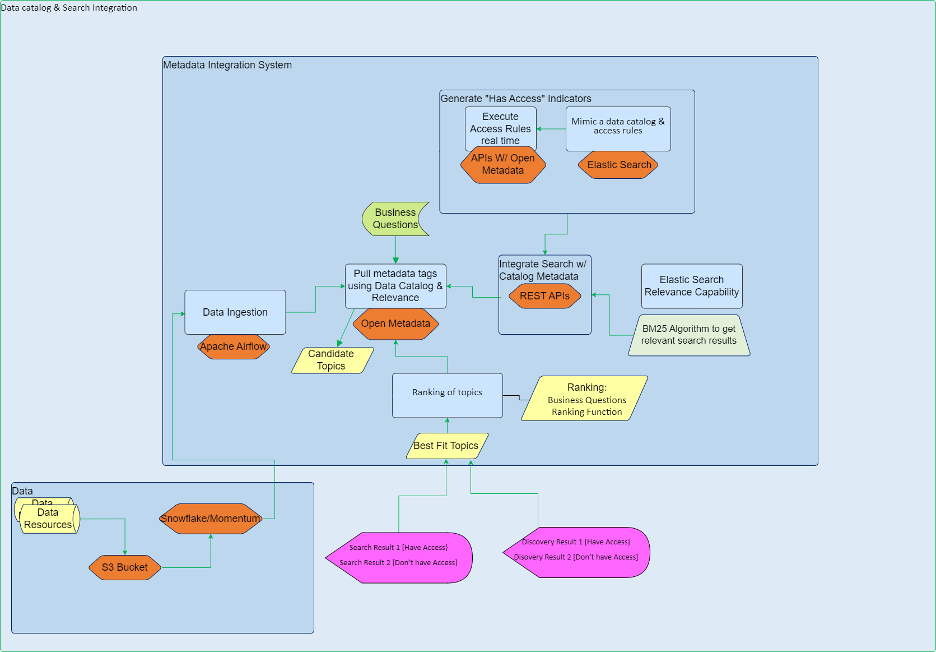
For this project, the S3 standard package would suffice. It costs $0.023 per GB per month, for the first 50TB of data. The cost increases as the data storage capacity increases but that is irrelevant since this project does not use more than 50TB of data.

# Algorithms & Analysis

## Solution Approach

Our Project on Integrating data catalogs with content search technologies focuses on integration between the different tools which can improve the efficiency and effectiveness of data discovery and analysis.

### Systems Architecture



The data resources we have used for this project are Kaggle and MITRE. These datasets are stored in an object storage service called Amazon s3 bucket. S3 bucket makes it easier to reduce expenses, organize data, and set up precise access controls to satisfy unique business, organizational, and compliance requirements using cost-effective storage classes and simple administration tools. Then from the S3 bucket, the data is ingested into Snowflake/Momentum. Then using Apache airflow, the data is ingested into Open Metadata. With the help of Open Metadata, you can manage metadata from beginning to end, enabling you to maximize the value of your data assets not only for established use cases like data governance and discovery, but also for new ones like data quality, observability, and human collaboration. Open Metadata is used in this project to produce data tags and using a ranking algorithm the best-fit tags for the data are determined. RESTful APIs are used to integrate data catalog with the search engine. The data catalog in this case is Open Metadata and the search engine is Elastic search. The elastic search uses a bag-of-words retrieval algorithm called BM25 scores a set of documents according to the query keywords that exist in each one, regardless of where in the text they appear. It is a group of scoring functions with marginally differing elements and constraints.

### Systems Security

For our project we are using various tools and all those come with the data security option which will be useful for our project. Here we will talk about all the security features offered by the various tools:

Amason S3 - S3 allows for server-side encryption, which encrypts your data automatically before it is stored there. Data can also be encrypted on the client side before being uploaded to S3. SSL/TLS-based encryption in transit is also supported by S3. Amazon S3 provides detailed access logs that we can use to monitor and audit access to our data.[19]

Snowflake - Snowflake uses multi-factor authentication (MFA) and a variety of authentication protocols to ensure that only authorized users can access data. Snowflake also offers role-based access control (RBAC) to enforce data access policies. [20]

Open metadata - To prevent sensitive data from being accessed or disclosed by unauthorized parties, open metadata technologies can include data masking features. The masking of confidential information, such as personally identifiable information (PII), financial data, and medical data, falls under this category. Open metadata tools can integrate with other security tools, such as vulnerability scanners and intrusion detection systems, to provide a comprehensive security solution for metadata and associated assets.[21]

Apache Airflow: Airflow supports secure connections to databases, cloud services, and other resources using protocols such as SSL and SSH. Airflow logs all user actions and system events, making it possible to track and audit system activity.[22]

### Systems Data Flows

The system data flow is the representation of information flow through various processes in the project. In these cases, starting from data resources to elastic search engines the data flows through various applications undergoing data cleaning, and data optimization. Beginning from the amazon s3 bucket the data is ingested into Open Metadata through Apache Airflow. The elastic search engine is integrated with APIs with Open Metadata to generate access indicators, which is the secondary goal of the project. The data is then integrated with Open Metadata using RESTful APIs. The data tags produced using Open Metadata and the search engine combinedly help with search optimization.

### Algorithms & Analysis

#### 3.1.4.1 BM25:

BM25 (Best Match 25) is a ranking algorithm used in information retrieval to measure the relevance of a document to a query. It is the default algorithm used by Elastic search. The BM25 algorithm's fundamental principle is to determine a relevance score for each document in a collection based on the term frequency (TF) and inverse document frequency (IDF) of the query terms within the document, and this score determines how the documents are ordered. Term Frequency ranks the search results based on how often the query terms appear in a document and how important those terms are to the document. Inverse Document Frequency is a calculation that scores how rare your word is in the corpus. The rarer the word is, the higher its score.

Elasticsearch rates the documents in a dataset based on the relevance of the search. It returns a data list that has been sorted in order of relevance. By including and removing variables that will change the balance between recall and precision, we can alter the score.

The accuracy of a search query and the search results are measured by search relevancy. We can ask, "How relevant is this document?" rather than, "Is this document relevant? "It depends on the precision and recall of two essential criteria. Recall functions like a unit of measurement. We want to make sure that the search results for a dataset contain all the pertinent documents. On the other hand, precision is comparable to a quality indicator. To make sure all the information in the search results is pertinent. Picture a sea filled with vibrant aquatic life, including fish, crabs, jellyfish, sharks, dolphins, and even SpongeBob as well. You're seeking purple fish in that sea. We would only retrieve fish that was a solid shade of purple in a high-precision scenario. That could seem appealing, but what if there were fish with purple spots or fish that were partially purple and partially yellow? Technically, those fish are still relevant to our search. On the other hand, in a high recall scenario, we would fetch anything purple or that is a fish. Sharks, dolphins, purple crabs, and jellyfish would all fall under this category. Even though "fish" was what we meant, the other creatures are still important because they are purple. But it would also include common fish of various hues just because they are fish. That outcome is not desirable. We fetched meaningful results in both instances; however, the results were either over or under-captured. Because enhancing one might make the other worse, precision and recall are frequently at odds with one another. Yet, the equilibrium between the two can assist us in producing a score that we can utilize to get pertinent results.

How it works:

Text analyzers are used by Elasticsearch to transform large amounts of text into optimum searchable data. They are utilized while inputting data as well as when performing data queries. Character Filter, Tokenizer, and Token Filter are the three components that make up Elasticsearch's text analyzer. The text analyzer can be highly customized, which means that every component of its architecture can be altered to fit a specific use-case.

The character filter modifies, adds, and removes textual components. For example, it can replace instances of specific strings and eliminate HTML characters. Text is divided into tokens, usually referred to as terms, by the tokenizer. The whitespace tokenizer (default tokenizer) separates the text anytime it comes across a whitespace. Some tokenizers, like the letter tokenizer, on the other hand, divide text anytime they come across a non-letter character. The token filter is a related character filter. Tokens may be added, subtracted from, or changed. There are some nice filters that can add synonyms, like the synonym token filter.

Terms can be mapped to many documents using the inverted index architectural design, which is far more efficient than the iterative approach for document search.

# Rule base access control and policies

## Overview

For sensitive data to be secure and private, access control in data catalogs is essential. In essence, a data catalog is a database that houses metadata on the data assets in a company, such as details about their location, format, ownership, and usage. Anyone who obtains access to a data catalog may be able to access all the data assets included in it if access control is not implemented there. As a result, sensitive data may be accessed without authorization or used inappropriately. Without access control, it is also challenging to trace who has accessed which data assets, making it challenging to identify and investigate security problems or data breaches.

Based on their job or need-to-know, companies can use access control in data catalogs to limit access to sensitive data assets to just approved people or groups. This might aid in avoiding security events such as data breaches and illegal data usage.

Let's Take an example like our project work, that a large healthcare organization has a data catalog that contains metadata about all the patient data they have collected over the years, including information on patient demographics, medical conditions, treatments, and prescriptions. This catalog is accessible to all employees within the organization. Any person inside the company could possibly access this sensitive patient data if access controls weren't in place. Data breaches, illegal data use, and other security issues would become quite likely as a result. For instance, a worker without the right clearance might view the medical records of friends or family members, or they could use the information maliciously or for personal advantage.[23]

Diagram

Description automatically generated

**Role-based access control (RBAC)** might be used by the healthcare company to impose access control on their data catalog and avoid similar mishaps. For instance, according to the demands of their respective jobs, they may establish several roles, such as doctors, nurses, administrators, and IT personnel, each with varying levels of access to patient data. The company might also put in place access controls based on the least privileged principle, which limits user access to only that which is necessary for them to carry out their job responsibilities.

With RBAC and least privilege access control in place, only authorized users within the organization can access sensitive patient data. This significantly reduces the risk of data breaches and other security incidents, and helps the organization comply with data privacy regulations. Now we will talk about how we have incorporated all these Features in our work using the Open metadata platform.[24]

## Organization &User

From small startups to huge corporations, OpenMetadata integrated with Elastic search is used by a range of businesses. Several well-known companies, such as Airbnb, Netflix, and Twitter, use this technique. To manage metadata across their data architecture and improve data discovery and analysis, these firms use OpenMetadata combined with Elastic search. These businesses may improve data governance, data quality, and decision-making by fusing the strength of Elastic search with the OpenMetadata metadata management tools.

Users may search and discover metadata throughout their whole data infrastructure utilizing Elastic search's robust search capabilities due to the integration with OpenMetadata. This facilitates users' access to and use of the data they require, leading to quicker and more effective decision-making.

The platform offers a central location for information management, making it simpler for businesses to manage metadata across their data infrastructure, according to users of OpenMetadata linked with Elastic search. Better data governance and data quality are the results of this. Because the platform combines the metadata management capabilities of OpenMetadata with the data exploration and analysis capabilities of Elastic search, it also enables enterprises to explore and analyze data more efficiently.

Since OpenMetadata is an open-source platform when connected with Elastic search, users have the freedom to modify it to meet their own requirements. As a result, it is a more adaptable and affordable option than exclusive metadata management and search solutions.

## Roles & Policies

Providing a rich collaborative experience for users of data assets within an organization is one of the primary objectives of Open Metadata. It is crucial to have some sort of governance in place as we develop product functionality that involves different consumers. Having 100s of users in an organization updating metadata of data assets can lead to potential inaccurate and low-quality metadata. This further hampers the discoverability and usage of data assets. Therefore, it becomes essential to restrict user rights using access control systems to enable the product's growth to accommodate different personas.

Our solution revolves around a simple model to assign a role to users that conform to the same persona and providing them with a set of permissions known as policies. Fig 6. shows relationship between user, Roles, and Policies [25]. Each user can be assigned one or more Roles and each role has a defined policy, and these policies are composed of a set of rules. Rules allow/deny access to metadata operations such as updating, tags, descriptions, owners, and lineage.

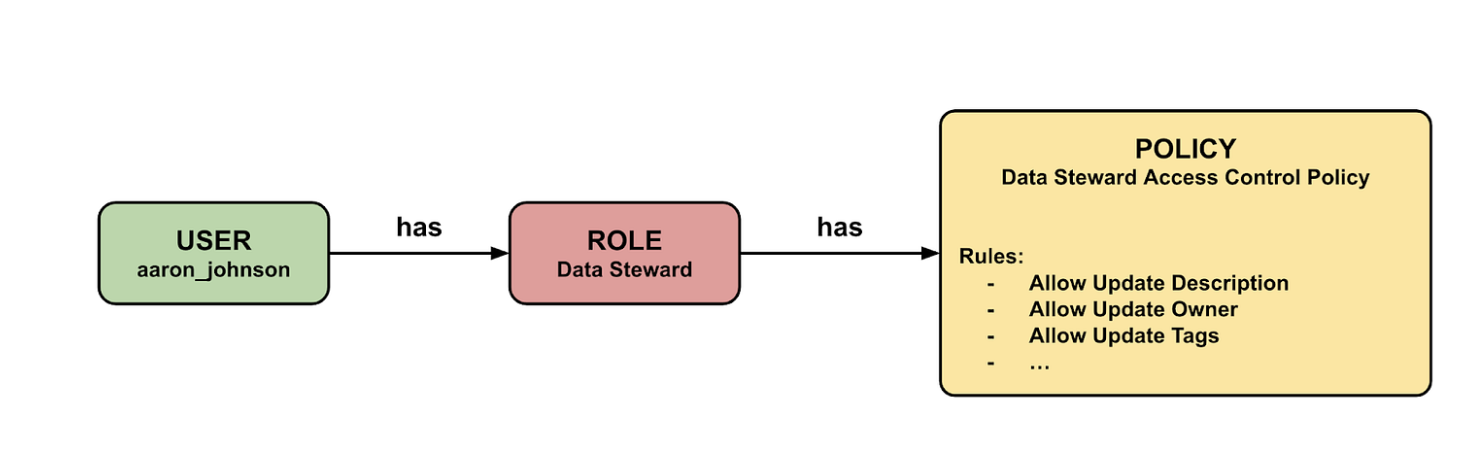
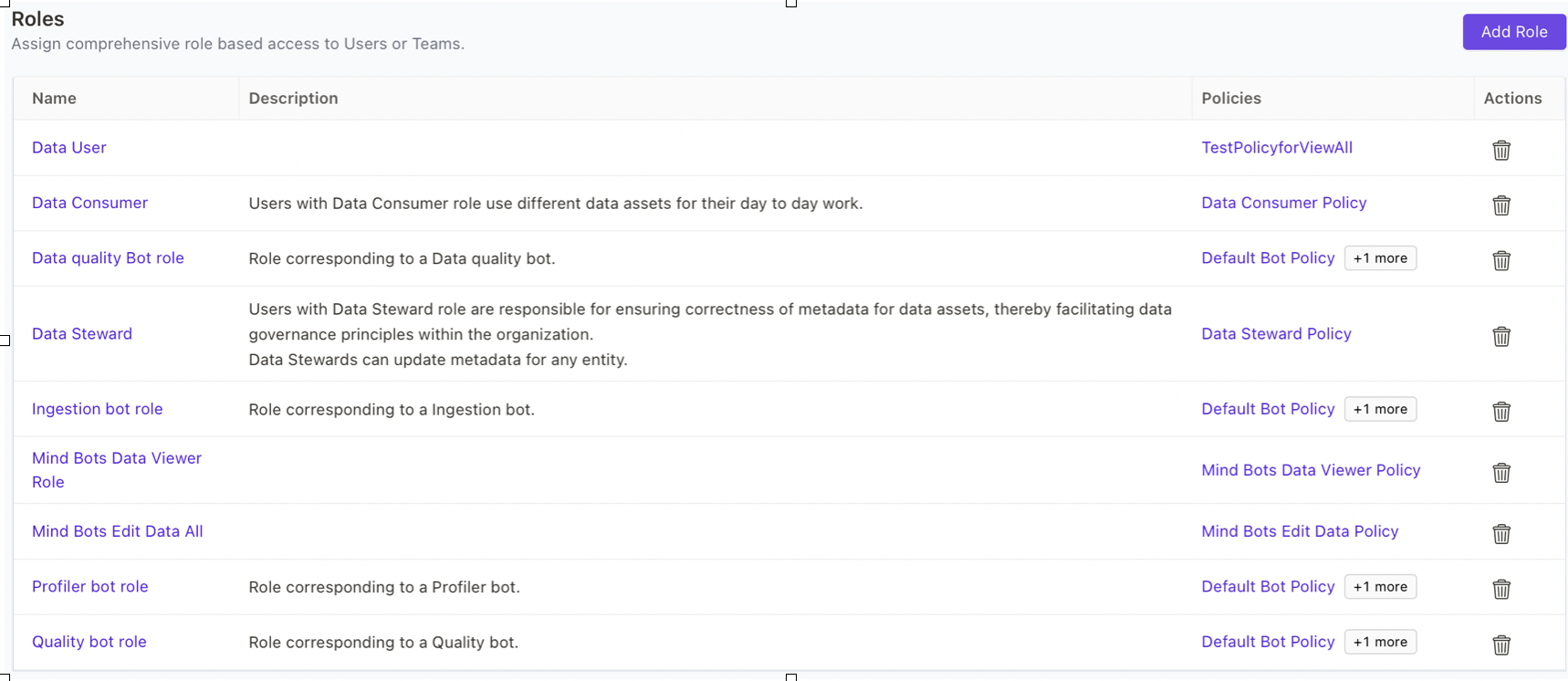


Fig 6. User, Role, and Policy relationship

For our implementation, we created various Roles that can be assigned to the users. As seen in Fig 7. from a single page, we can add a new role, its description and can also add/update the permissions. Some of the common roles include Data Steward which ensures correctness of metadata for data assets, Data Consumer and Open Metadata Admin.

Fig 7. Roles

Policies help to ensure that metadata is accurate, complete, and consistent, which in turn helps the organization to make better decisions. Once policies are defined, they can be enforced through automated tools and workflows. Fig 8. shows different policies that we implemented with their description and in which role they are used.

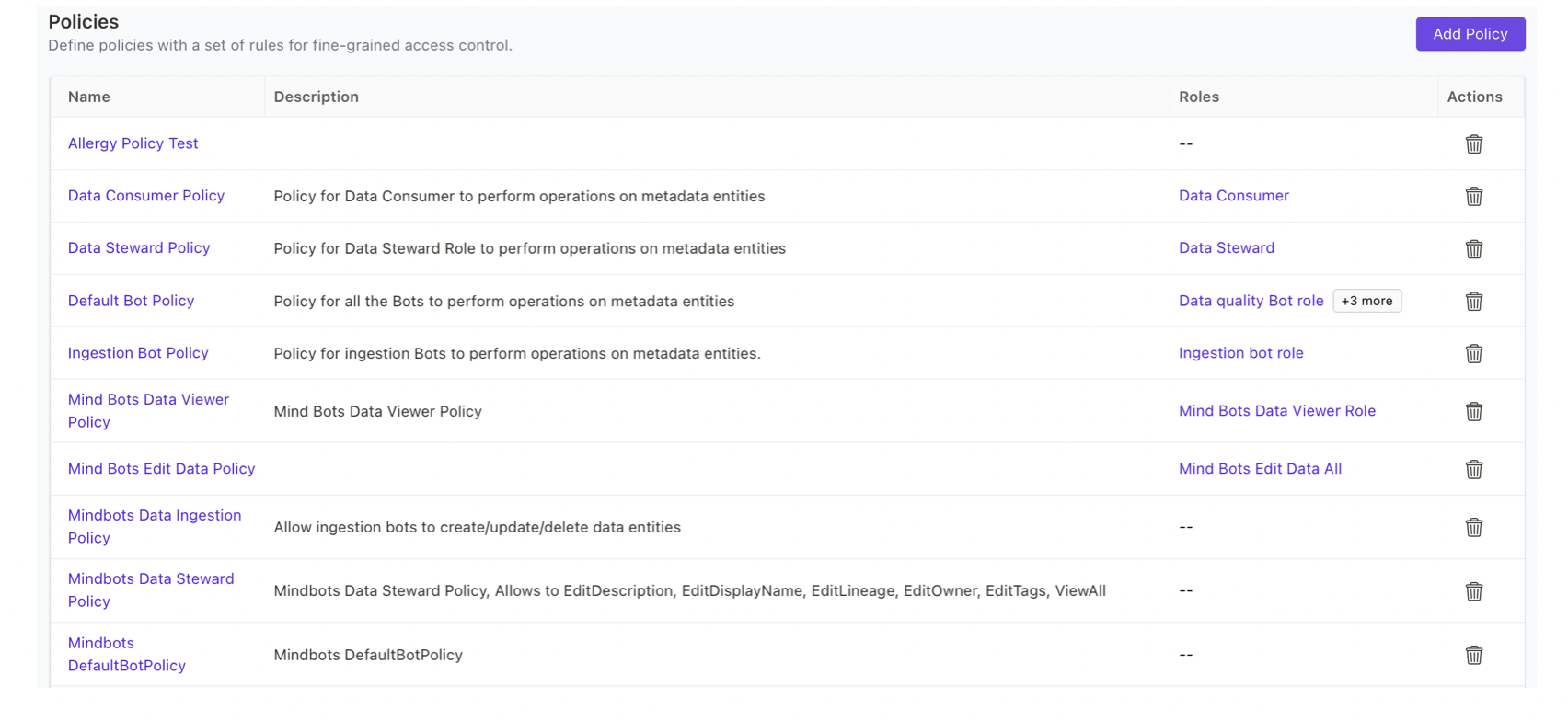


Fig 8. Policies

## Visualizations

In this section we got visualization about metadata and access controls.

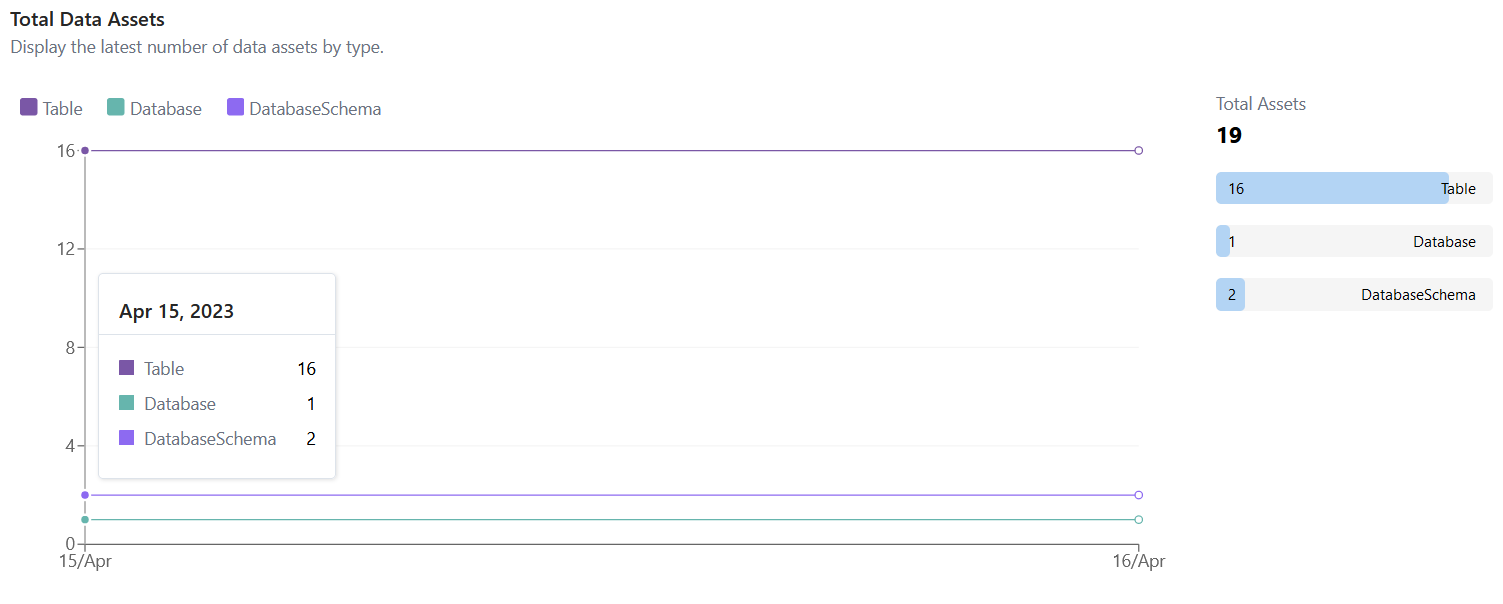


Fig 9. Data Assets

Fig 9 shows the details of the data assets in the Database as seen in OpenMetadata.

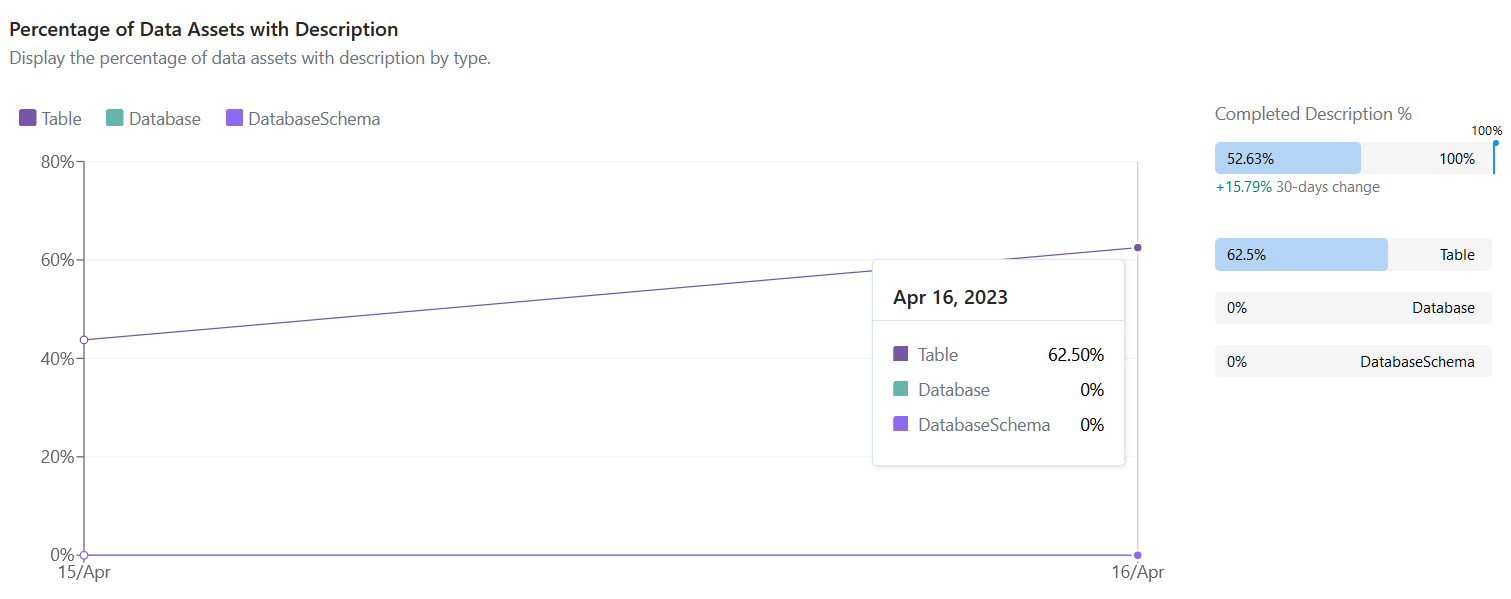
 Fig 10. Dataset Description

Fig 10 shows the percentage of data description completed. Data description is needed as it is the main factor to match the tags in elastic search.

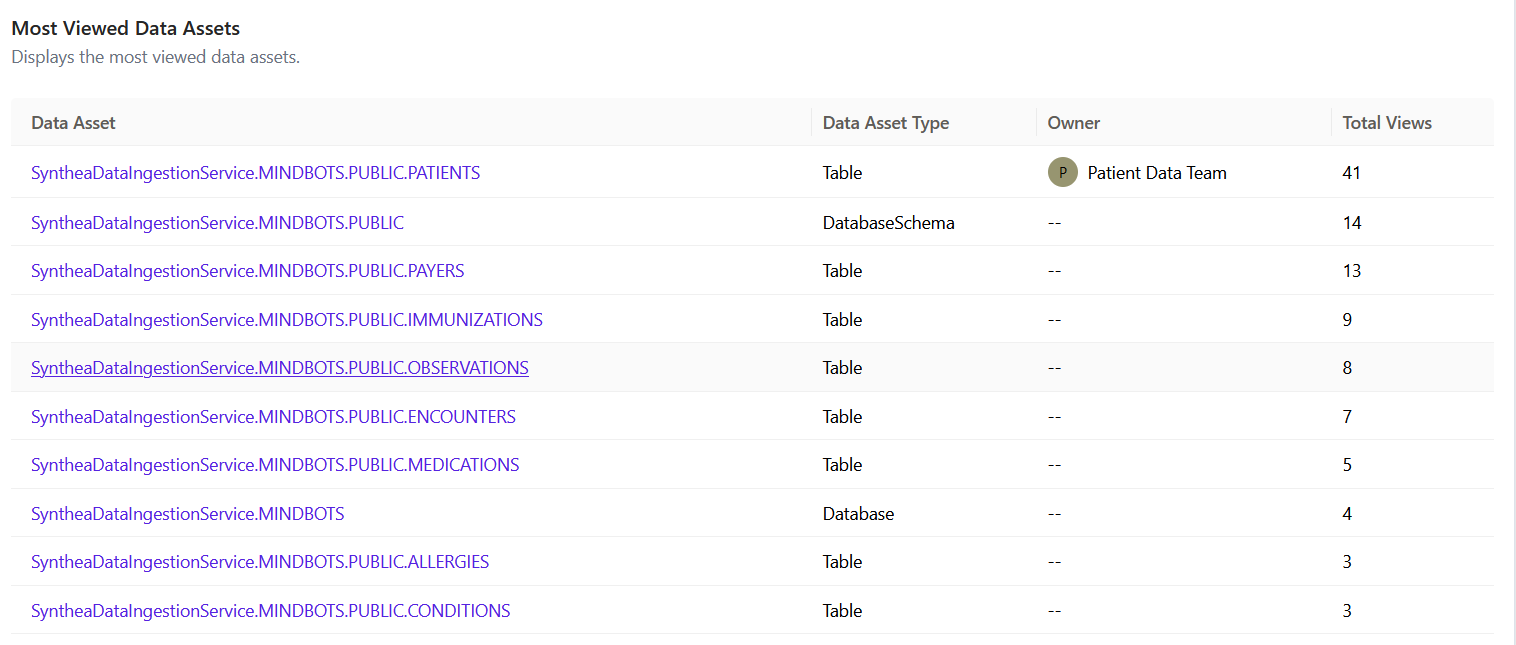
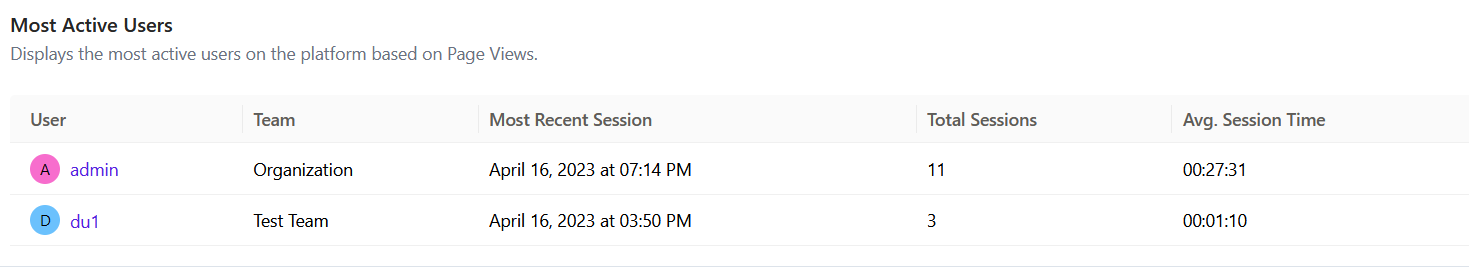
 Fig 11. Data Assets Views

Fig 11 shows us the most viewed data assets. This feature can be used if the views are in large scale and if the organization needs to restrict specific number of accesses for performance boosting.

 Fig 12. Active Users

The visuals in figure 12 show us the most active users, hence helpful in knowing active consumers.

# Findings

**INSTRUCTIONS**

Discuss the major findings of the project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

# Summary

**INSTRUCTIONS**

Summarize the overall project and results for the reader. What did you discover, prove, disprove, etc.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

# Future Work

**INSTRUCTIONS**

This is critical section of the report. Propose future follow-on work or next step(s) for the project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Appendix

Appendix A: Glossary

|  |  |
| --- | --- |
| Term | Definition |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**INSTRUCTIONS**

Place all terms which require definitions in the Appendix A: Glossary.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Appendix B: GitHub Repository

Overview

**INSTRUCTIONS**

Provide a GitHub Link and the README.MD content. Do not just provide a link to the GitHub repository but provide a narrative paragraph which introduces the project. This section should mirror the look and feel of a well-documented professional GitHub site.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

GitHub Repository Link

GitHub Repository Contents

Appendix C: Risks

Sprint 1 Risks

**INSTRUCTIONS**

Include the risk table associated with the Sprint. Below the risk table provide a narrative description of how the risks and mitigation plans were identified, what the team got correct, what the team could have done differently, how accurate was the team in identifying the risks, did the team encounter any unanticipated risks, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 2 Risks

**INSTRUCTIONS**

Include the risk table associated with the Sprint. Below the risk table provide a narrative description of how the risks and mitigation plans were identified, what the team got correct, what the team could have done differently, how accurate was the team in identifying the risks, did the team encounter any unanticipated risks, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 3 Risks

**INSTRUCTIONS**

Include the risk table associated with the Sprint. Below the risk table provide a narrative description of how the risks and mitigation plans were identified, what the team got correct, what the team could have done differently, how accurate was the team in identifying the risks, did the team encounter any unanticipated risks, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 4 Risks

**INSTRUCTIONS**

Include the risk table associated with the Sprint. Below the risk table provide a narrative description of how the risks and mitigation plans were identified, what the team got correct, what the team could have done differently, how accurate was the team in identifying the risks, did the team encounter any unanticipated risks, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 5 Risks

**INSTRUCTIONS**

Include the risk table associated with the Sprint. Below the risk table provide a narrative description of how the risks and mitigation plans were identified, what the team got correct, what the team could have done differently, how accurate was the team in identifying the risks, did the team encounter any unanticipated risks, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Appendix D: Agile Development

Scrum Methodology

**INSTRUCTIONS**

Provide a narrative of the team efforts in using a scrum methodology for a data analytics engineering project. Describe how easy/difficult was it to adapt to the Scrum methodology. Did the team conduct a daily scrum? If not, how often did the team conduct a scrum. Describe how easy/difficult it was to use the YouTrack tool to manage the project. Don’t be limited to just these questions. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 1 Analysis

**INSTRUCTIONS**

Provide a narrative of the team’s efforts during this Sprint. Be sure to include – but not be limited to – how the team identified the User Stories, how well the team performed with the various tasks, how easy/difficult it was for the team to manage their activities during the Sprint, what did the team do correct, what could/should the team have done differently, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 2 Analyis

**INSTRUCTIONS**

Provide a narrative of the team’s efforts during this Sprint. Be sure to include – but not be limited to – how the team identified the User Stories, how well the team performed with the various tasks, how easy/difficult it was for the team to manage their activities during the Sprint, what did the team do correct, what could/should the team have done differently, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 3 Analysis

**INSTRUCTIONS**

Provide a narrative of the team’s efforts during this Sprint. Be sure to include – but not be limited to – how the team identified the User Stories, how well the team performed with the various tasks, how easy/difficult it was for the team to manage their activities during the Sprint, what did the team do correct, what could/should the team have done differently, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 4 Analysis

**INSTRUCTIONS**

Provide a narrative of the team’s efforts during this Sprint. Be sure to include – but not be limited to – how the team identified the User Stories, how well the team performed with the various tasks, how easy/difficult it was for the team to manage their activities during the Sprint, what did the team do correct, what could/should the team have done differently, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Sprint 5 Analysis

**INSTRUCTIONS**

Provide a narrative of the team’s efforts during this Sprint. Be sure to include – but not be limited to – how the team identified the User Stories, how well the team performed with the various tasks, how easy/difficult it was for the team to manage their activities during the Sprint, what did the team do correct, what could/should the team have done differently, etc. Think of this writeup as a “lessons learned” that you would like to pass along to any project team thinking of doing a similar project.

**DELETE THIS TEXT BOX AFTER YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS.**

Reference

1  Zaidi, E., et al.Data catalogs are the new black in data management and analytics (2017). <https://www.gartner.com/en/documents/3837968/data-catalogs-are-thenew-black-in-data-management-and-a>

2  Debowski, H., et al.: Using semantic technologies to manage a data lake: data catalog, provenance and access control, p. 17 (2020)

3 Labadie, C., et al.: Fair enough? Enhancing the usage of enterprise data with data catalogs. In: 2020 IEEE 22nd Conference on Business Informatics (CBI), vol. 1, pp. 201–210, June 2020.

4 M. D. Wilkinson et al., “The FAIR Guiding Principles for scientific data management and stewardship,” Sci. Data, vol. 3, p. 160018, Mar. 2016, Doi: 10.1038/sdata.2016.18.

5. <https://www.oracle.com/big-data/data-catalog/what-is-a-data-catalog/>.

6. Wells, D. (2019). Introduction to Data Catalogs.

7. Hendler, J. (2014, December 1). Data integration for heterogenous datasets. Big data.

Retrieved February 12, 2023, from <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4276119/>

8. Marisa. (2022, July 5). Open Metadata 0.11 release. Medium. Retrieved February 12, 2023, from [https://blog.open-metadata.org/openmetadata-0-11-release-8b82c85636a](https://blog.open-metadata.org/openmetadata-0-11-release-8b82c85636a )

9. <https://www.elastic.co/what-is/elasticsearch>

10. Antti Ala-Ilkka | October 14, 2019, | from: <https://www.addsearch.com/blog/faceted-search>

11. Moutjo Sen |Aug 30,2018 | From: <https://medium.com/@mourjo_sen/a-detailed-comparison-between-autocompletion-strategies-in-elasticsearch-66cb9e9c62c4>

12. Haushofer, B., & Klas, W. (2010). A survey of techniques for achieving metadata interoperability. ACM Computing Surveys (CSUR), 42(2), 1-37.

13 A. Kumar, A. Bandyopadhyay, H. Bhoomika, I. Singhania, and K. Shah, “Analysis of network traffic and security through log aggregation,” International Journal of Computer Science and Information Security (IJCSIS), vol. 16, no. 6, 2018.

14.  R. Kuc and M. Rogozinski, ´ Elasticsearch Server. Packt Publishing Ltd, 2016.

15. C. Gormley and Z. Tong, Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine.” O’Reilly Media, Inc.”, 2015.

16. P. McDermott, ‘‘Building open government,’ Government Inf. Quart., vol. 27, no. 4, pp. 401–413, Oct. 2010.

17. W3C. (2020). Data Catalog Vocabulary (DCAT)–Version 2. W3C

18. Walonoski J, Klaus S, Granger E, Hall D, Gregorowicz A, Neyarapally G, Watson A, Eastman J. Synthea™ Novel coronavirus (COVID-19) model and synthetic data set. Intelligence-Based Medicine. 2020 Nov; 1:100007. <https://doi.org/10.1016/j.ibmed.2020.100007>

19. Security best practices for Amazon S3

https://docs.aws.amazon.com/AmazonS3/latest/userguide/security-best-practices.html

 20. snowflake support <https://baffle.io/snowflake/?utm_term=snowflake%20data%20security&utm_campaign=AWS+Search+New&utm_source=adwords&utm_medium=ppc&hsa_acc=9421251531&hsa_cam=18723977187&hsa_grp=144020711978&hsa_ad=631220416021&hsa_src=g&hsa_tgt=kwd-871199414817&hsa_kw=snowflake%20data%20security&hsa_mt=e&hsa_net=adwords&hsa_ver=3&gclid=CjwKCAjwiOCgBhAgEiwAjv5whEmxl9q--ktXugX0zvLvM-Ida5HNwjd2x6WY9AdX4F3Nd5lZwCfINRoCPf4QAvD_BwE>

21. Enable security https://docs.open-metadata.org/deployment/security

22. Apache Security by Apache airflow <https://airflow.apache.org/docs/apache-airflow/stable/administration-and-deployment/security/index.html>

23. Data Catalog: A Key to Unlocking Business Insights [https://levelup.gitconnected.com/data-catalog-a-key-to-unlocking-business-insights-750d261a350e]

24. Role-Based Access Control [ https://auth0.com/docs/manage-users/access-control/rbac#:~:text=Role%2Dbased%20access%20control%20(RBAC,assigning%20permissions%20to%20users%20individually.]

25. Matt. (2023, April 3). *Building Access Control for OpenMetadata*. Medium. Retrieved April 20, 2023, from <https://blog.open-metadata.org/building-access-control-for-openmetadata-5b842a2abd90>

**This page intentionally left blank**