Published online on the journal's webpage: **http://journal.ittelkom-pwt.ac.id/index.php/dinda**

Journal of Dinda

Data Science, Information Technology, and Data Analytics

Vol. 5 No. 1 (2025) 45 - 51 E-ISSN: 2809-8064

Comprehensive Lakehouse Data Architecture Model for College Accreditation

Nenen Isnaeni [1*], Bambang Purnomosidi Dwi Putranto [2] , Widyastuti Andriyani [3] , Siti Khomsah [4]
[1*]Department of Informatics, Faculty of Informatics, Telkom University
[2,3] Department of Master Information Technology, Faculty of Information Technology, Universitas Teknologi Digital Indonesia
[4]Data Science, Faculty of Informatics, Telkom University
[1*]nenisna@telkomuniversity.ac.id, [2]bpdp@utdi.ac.id, [3]widya@utdi.ac.id, [4]sitijk@telkomuniversity.ac.id

Abstract

Accreditation is an assessment activity that determines the feasibility of study programs at a university. College accreditation data comes from various sources and includes multiple data types: semi-structured, unstructured, or structured. Over time, the volume of data will continue to grow and develop, so there is a possibility of data redundancy and a long time to collect the data needed for accreditation activities. The solution is integrating data. This research aims to design a data architecture to facilitate the management of university accreditation data using the Lakehouse data architecture model. All data types can be stored on one platform in the Lakehouse data architecture. In this research, the identification, integration, and data transformation process for university accreditation data is carried out. The data used in this research is academic data in which there are with. The study's results provide an overview of the data flow process in the Lakehouse data architecture model to help better manage university accreditation data. This architecture also supports real-time data analysis so that the accreditation process can be carried out more effectively and efficiently.

Keywords: *accreditation, data analysis, data architecture, data lakehouse, data warehouse*

1. Introduction

The government establishes higher education accreditation as a process of assessing the quality of higher education. Regulation of the Minister of Education and Culture (Permendikbud) Number 5 of 2020 [1] states that accreditation is an assessment activity to determine the suitability of study programs and universities, as well as an assessment of the quality of education at a university.

Higher education will get a good accreditation score if all quality performance of each parameter/point of assessment can exceed the National Higher Education Standards [2]. Accreditation data comes from various sources or departments with multiple data types: structured, semi-structured, and unstructured. The volume of data continues to grow and develop over time, requiring universities to be able to adapt to these developments. According to [3], unlimited scalability and the ability to quickly adapt to increasing volumes are essential. Centralized or integrated accreditation data makes the accreditation process more effective and efficient. The latest emerging data management architecture trend is the data Lakehouse, which combines the flexibility of a data lake with that of a data warehouse [4]. Data Lakehouse is a new concept that has emerged as a solution for integrated data architecture [5]. The need to quickly generate knowledge that can be retrieved from unstructured data obtained from various distributed sources requires integration between the data warehouse and data lake to form a Lakehouse (LH) [6]. Lakehouse data architecture combines the integration of structured and unstructured data and combines two different architectures, namely data warehouse and data lake [7], [8]. The use of a data warehouse is not suitable for more complex data storage cases because the data warehouse only supports structured data storage [9]. Data warehouse architecture is not considered a solution to overcome big data challenges in higher education, which consists of various data types [10].

The Lakehouse data architecture model supports real-time analytics. In principle, the integrated engine fully supports SQL, including for complex queries and

Received: 17-01-2025 | Accepted: 12-02-2025 | Published: 13-02-2025

complicated data models, can operate directly on data in the Lakehouse without the need to make additional copies, provides low query latency with most data stored in local memory or SSD, and supports real-time [11]. This data architecture facilitates complex multilevel predictive analytics on real-time streaming data from various sources, including the web and social media [12]. The Lakehouse system augments the data lake with data management features such as ACID transactions and metadata management to efficiently optimize queries [13]. Decision-makers in educational institutions are interested in utilizing big data about learning objects, students, faculty, staff, and the institution itself. In the big data, there are opportunities to improve student success and managers to manage the institution more efficiently. Therefore, educational leaders need to consider how best to utilize data analytics effectively [10].

In some universities, accreditation data is still spread across various departments, so to access the data, you must contact their respective sections; in one particular case, if the data is interrelated between one part and another, there is often data redundancy. The data collection process is still done manually. It can be said that the data management is not well organized, as shown in Figure 1.

Data Lakehouse is a unique data storage solution for all data types, whether structured, semi-structured, or unstructured. It provides data quality and governance standards from the data warehouse [14]. Data Lakehouse offers better data management, reduced data redundancy, and efficient usage time [4]. In addition to having better data management, the costs used to manage data Lakehouse are lower [15].

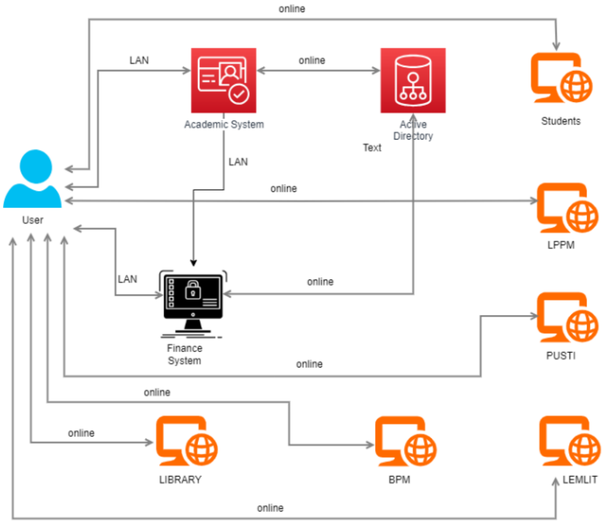


Figure 1. Old system flow

2. Research Methods

Figure 2 shows the stages of this research, starting with a literature study by reviewing several relevant literature reviews. The next step is data collection, data analysis, and data architecture design.

2.1 Literature Study

Figure 3. is the Data Lakehouse architecture taken from the book by [5]. The data lakehouse architecture has the unique ability to manage and combine all data types.

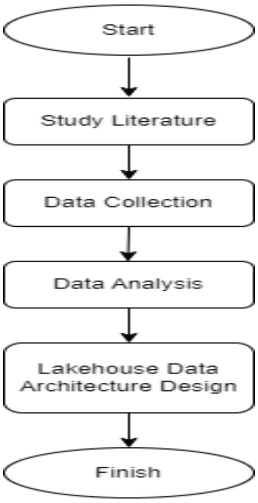


Figure 2. Research Design

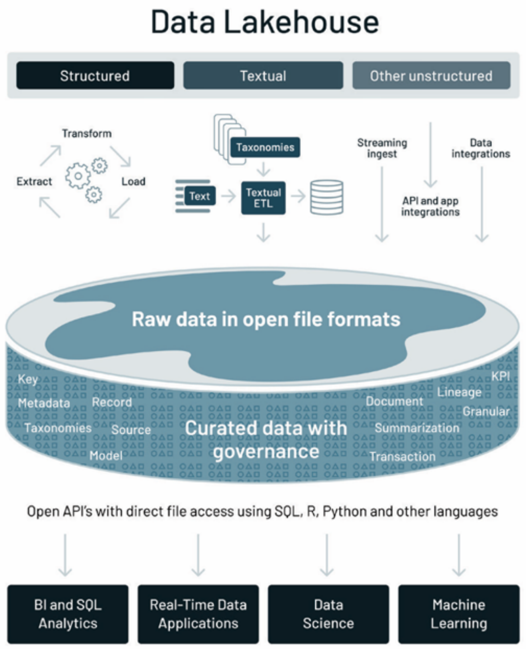


Figure 3. Data Lakehouse Architecture from [5]

from all parts. The data Lakehouse architecture integrates data from the data lake and warehouse to support data processing and analysis. The process starts with collecting data from various sources, including structured data such as relational databases, data in the form of documents or logs, and unstructured data in the form of images and videos. The data will go through an integration process, which includes Extract, Transform, and Load (ETL) steps to retrieve the data from its source, process it according to the needs, and load it into the system. The textual ETL process is used for text-based data with taxonomy mapping. In contrast, real-time data is collected through a streaming ingest mechanism, and other data is integrated through various data integration methods. After going through the integration stage, the data is stored in two main forms. First, raw data in open file formats such as ORC can be used without much preliminary modification, making it more flexible for future use. Second, data organized in Curated Data Governance is equipped with metadata and taxonomies that facilitate data analysis and management. At this stage, the system implements governance, including quality control, data security, and regulatory compliance. All data stored in the data Lakehouse can be accessed directly by users through open APIs. The data Lakehouse supports Business Intelligence (BI) and SQL Analytics for structured data analysis. It supports the development of Real-Time Data Applications, where real-time data can be utilized for applications requiring immediate updates.

2.2. Data Collection

This study uses the STIKes Kuningan Nursing Study Program accreditation data in 2020. Data collection in this study uses interviews and direct observation. The data is divided into nine criteria, as listed in Table 1.' criteria 1' is the name of the criteria containing data about the university's vision, mission, goals, and strategies. 'criteria 2' contains governance and higher education cooperation data. 'criteria 3' contains data on university students. 'criteria 4' contains human resource data in the university. 'criteria 5' contains the university's financial data, facilities, and infrastructure. 'criteria 6' is about education, 'criteria 7' is about research, 'criteria 8' is about community service, and 'criteria 9' is about university outputs and achievements.

2.3 Data Analysis

At this stage, the data that has been collected is analyzed and grouped into several criteria. The 2020 rules for LAM accreditation follow the nine criteria rules. Various types of data are taken from multiple sources, including structured, unstructured, and semi-structured data from various sources such as academic applications and websites.

Table 1. List of Accreditation Criteria

id	**name**	**information**
1	criteria 1	vision, mission, goals and strategy
2	criteria 2	governance, governance and cooperation
3	criteria 3	student
4	criteria 4	human Resources
5	criteria 5	finance, facilities and infrastructure
6	criteria 6	education
7	criteria 7	study
8	criteria 8	service to society
9	criteria 9	outcomes and achievements

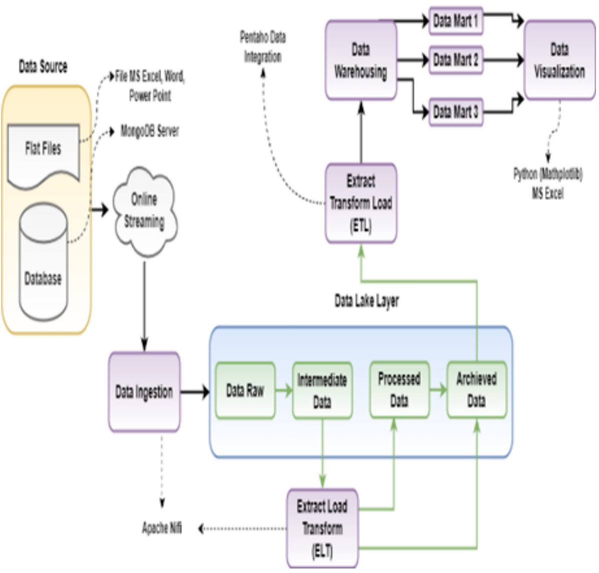


Figure 4. Data Lakehouse Architecture

2.4 Lakehouse Data Architecture Design

At this stage, the process of designing the data Lakehouse architecture is carried out, as shown in Figure 4. The data source contains a variety of structured, semi-structured, and unstructured data formats from various departments. Data that is already available is in Word, PPT, Excel, and jpg/png formats. The data will go through an online data delivery process to go to data ingestion using Apache Nifi.

- Raw data: data in the form of raw data.
- Intermediate data: there is a data filter process using **Apache N**ifi. Data that will be changed in format will go through the ELT process stage.

- Processed data: data from the ELT process is collected, and there is a change in the format/structure of the data, for example, from unstructured to structured.
- Archived data: all data is archived, both data from the ETL process and data that does not require format changes

3. Results and Discussion

3.1 ELT Process

The ELT process consists of sequences: Extract, Load, and Transform. In this process, the data will be absorbed into the data lake in raw format and then transformed when needed for analysis [16]. Data sources consist of internal college data in the form of docx, ppt, pdf, xls, mp3, mp4, jpg, and png files stored on Google Drive, while external data sources consist of JSON files taken from the college academic website. Figure 5 shows the ELT process from the data source to the data lake. Stages that occur in the ELT process in the data lake. In the data ingestion stage, raw data from Google Drive is inserted, not through further data processing. In obtaining quality data, the intermediate stage extracts data from Google Drive. Furthermore, at the Load stage, the processed data is loaded into the data storage. This process starts with data identification, followed by changing the data format to make it suitable, data cleaning, and data validation to ensure quality before Finally, being loaded into the data Lakehouse. In the Transform process, data cleaning, removal of duplicate data, and uniform XML formatting occur. The data is stored in the processed data directory in preparation for further analysis. Data that has been processed is ready for analysis (Processed Data). Archived Data serves as a backup in case recovery is required in the future.

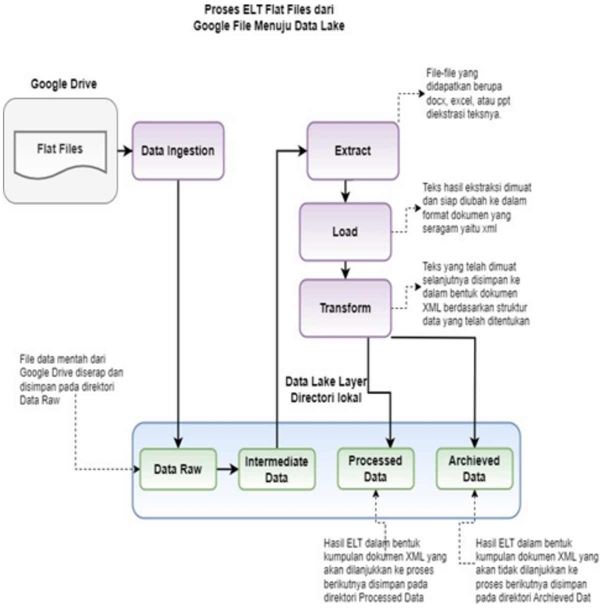

Proses ELT Flat Files dari
Google File Menuju Data Lake
Google Drive
Flat Files
Data Ingestion
Extract
Load
Transform
File-file yang didapatkan berupa docx, excel, atau ppt diekstrasi teksnya.
Teks hasil ekstraksi dimuat dan siap diubah ke dalam format dokumen yang seragam yaitu xml
Teks yang telah dimuat selanjutnya disimpan ke dalam bentuk dokumen XML berdasarkan struktur data yang telah ditentukan
File data mentah dari Google Drive diserap dan disimpan pada direktori Data Raw
Data Lake Layer
Directori lokal
Data Raw
Intermediate Data
Processed Data
Archieved Data
Hasil ELT dalam bentuk kumpulan dokumen XML yang akan dilanjukkan ke proses berikutnya disimpan pada direktori Processed Data
Hasil ELT dalam bentuk kumpulan dokumen XML yang akan tidak dilanjukkan ke proses berikutnya disimpan pada direktori Archieved Dat

Figure 5. ELT Process

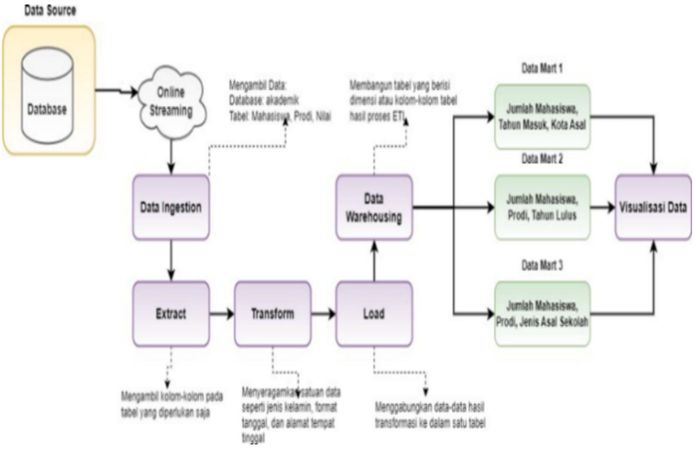

Data Source
Database
Online Streaming
Mengambil Data:
Database: akademik
Tabel: Mahasiswa, Prodi, Nilai
Data Ingestion
Extract
Transform
Load
Data Warehousing
Membangun tabel yang berisi dimensi atau kolom-kolom tabel hasil proses ETL
Mengambil kolom-kolom pada tabel yang diperlukan saja
Menyeragamkan satuan data seperti jenis kelamin, format tanggal, dan alamat tempat tinggal
Menggabungkan data-data hasil transformasi ke dalam satu tabel
Data Mart 1
Jumlah Mahasiswa, Tahun Masuk, Kota Asal
Data Mart 2
Jumlah Mahasiswa, Prodi, Tahun Lulus
Data Mart 3
Jumlah Mahasiswa, Prodi, Jenis Asal Sekolah
Visualisasi Data

Figure 6 ETL Process from Database Data Source to Data Warehouse

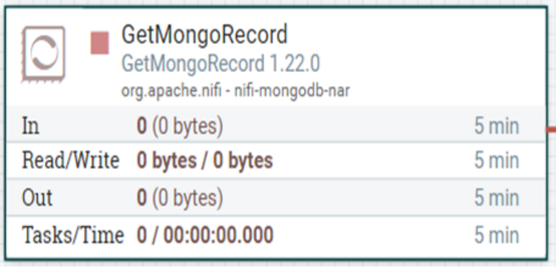
3.2. ETL Process

The external data used in this research comes from the university's academic website. A data source is the source of data stored in the database. According to [17], Extract-Transform-Load (ETL) is a process of extracting data sourced from different data, transforming it, and loading it into an integrated repository such as RDBMS-OLAP or data lake.

Figure 6 shows the ETL process from the source database to the data warehouse. This source database refers to the academic database based on MongoDB and the data warehouse with the academic_dw database, which is also based on MongoDB. This ETL process consists of several main stages: Data Ingestion, Extract, Transform, Load, Data Warehousing, and Data Visualization. The stages of the process that occur are as follows:

A. Data Ingestion

The data ingestion starts with determining the relevant data sources and then identifying their data formats and structures. Data will be extracted from its source using specific techniques and methods such as ETL (Extract, Transform, Load), ELT (Extract, Load, Transform), or CDC (Change Data Capture). The extracted data is processed and cleaned from anomalies or errors using data transformation. This data ingestion stage is handled using Apache Nifi software. In this


GetMongoRecord
GetMongoRecord 1.22.0
org.apache.nifi - nifi-mongodb-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 bytes	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Figure 7. Processor Get Mongo Record

The GetMongoRecord processor shown in Figure 7 carries out Apache software and data ingestion.

The GetMongoRecord processor in NiFi retrieves specific data from MongoDB collections using queries. The retrieved data, such as student, grade, and program data, can be sorted, restricted, and transformed before being used for further analysis or processed in the next stage of the data flow.

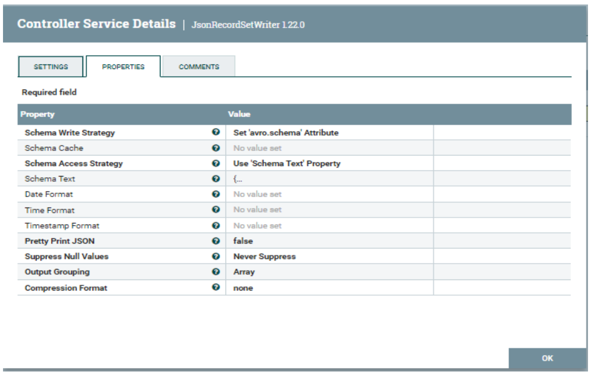
B.Extract

Apache NiFi plays a vital role in the data extraction process. Using a RecordWriter such as JsonRecordSetWriter, NiFi can retrieve data from various sources regularly and feed it into the data warehouse for further analysis. The setup of this service is shown in Figure 7.

The SplitJSON processor in Apache NiFi, shown in Figure 9, splits large JSON data into smaller parts based on specific patterns. The process allows for more granular data processing. This processor accepts the whole JSON document as input and then breaks it down according to a predefined scheme.

C.Transform

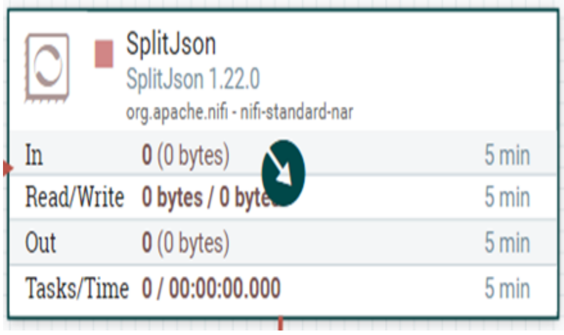
The transformation process in data warehousing transforms raw data into an analysis-ready form. Apache NiFi uses processors such as EvaluateJSONPath to evaluate JSONPath expressions on JSON data, manipulating the data more flexibly before it is created in the data warehouse. The configuration of EvaluateJSONPath involves several aspects, including:


Controller Service Details | JsonRecordSetWriter 1.22.0

Required field

Property	Value
Schema Write Strategy	Set 'avro.schema' Attribute
Schema Cache	No value set
Schema Access Strategy	Use 'Schema Text' Property
Schema Text	{...
Date Format	No value set
Time Format	No value set
Timestamp Format	No value set
Pretty Print JSON	false
Suppress Null Values	Never Suppress
Output Grouping	Array
Compression Format	none

Figure 8 Mongo DB Controller Service Details


SplitJson
SplitJson 1.22.0
org.apache.nifi - nifi-standard-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 byte	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Figure 9 Processor Split JSON

1. JSONPath Expression: used to specify the part of the JSON document to be evaluated. This expression can define elements in JSON, such as objects, arrays, or values.

2. Destination: This option selects where the evaluation results will be stored, such as data stream attributes, content, or attributes of a record.

3. Return Type: used to set the type of evaluation result, such as String, JSON, or Integer, depending on the expected data type.

D. ReplaceText

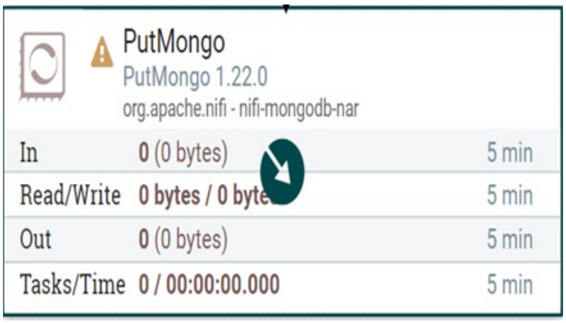
The ReplaceText processor in Apache NiFi is used to find and replace specific text in a data stream. This processor works to clean data, change formatting, or remove unwanted information. Configuring ReplaceText involves several aspects, including:

- Search Value: The text value or pattern to search for in the data.
- Replacement Value: The text or expression to be used as a replacement for the found value.
- Evaluation Mode: The evaluation mode used, such as replacing all matches, only the first one, or using regular expressions.
- Character Set: The charset or character set used to encode and decode text.

E. Merge Content

The MergeContent processor in Apache NiFi is used to merge multiple data streams into one. The configuration of MergeContent involves several aspects, including:

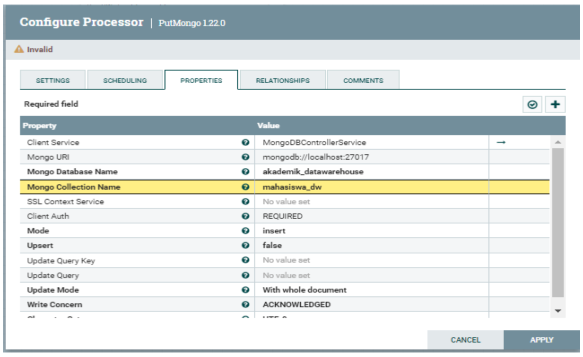
- Merge Format: Select the merge format, such as "Binary" to merge raw data or "Text" to merge text data.
- Minimum Number of Entries: The minimum number of entries required before a merge helps control when a merge should occur.
- Maximum Number of Entries: The maximum number of entries before the merge is triggered so the size of the resulting data stream can be set



PutMongo
PutMongo 1.22.0
org.apache.nifi - nifi-mongodb-nar

In	0 (0 bytes)	5 min
Read/Write	0 bytes / 0 byte	5 min
Out	0 (0 bytes)	5 min
Tasks/Time	0 / 00:00:00.000	5 min

Figure 10 Processor Put MongoDB



Configure Processor | PutMongo 1.22.0

Invalid

Required field

Property	Value
Client Service	MongoDBControllerService
Mongo URI	mongodb://localhost:27017
Mongo Database Name	akademik_datawarehouse
Mongo Collection Name	mahasiswa_dw
SSL Context Service	No value set
Client Auth	REQUIRED
Mode	insert
Upsert	false
Update Query Key	No value set
Update Query	No value set
Update Mode	With whole document
Write Concern	ACKNOWLEDGED

Figure 11 Put Mongo Configures

• Merge Strategy: The merge strategy used, such as merging all data streams or only adjacent ones.

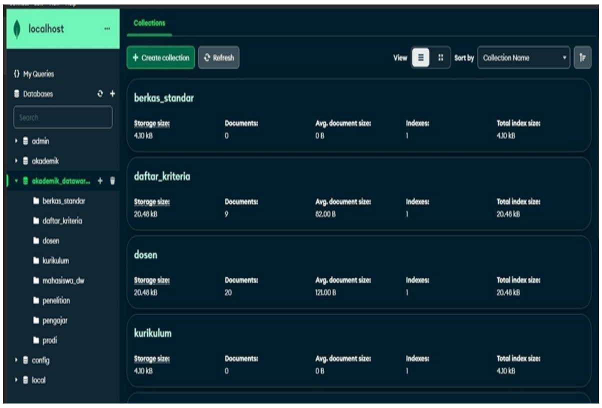
F. Load

Clean and structured data is loaded into the data warehouse tables in this stage. This data consists of facts (numerical data) and dimensions (attributes that describe facts). The PutMongo processor on Apache NiFi carries out this process. The appearance of this process is shown in Figure 10.

As in Figure 10, the PutMongo processor in Apache NiFi stores data from the NiFi data stream into the MongoDB database. Data submitted in JSON format will be written into a predefined MongoDB collection. PutMongo configuration involves several aspects, including:

- Connection String: Connection information to the MongoDB database, including host, port, and other required options.
- Collection Name: The name of the MongoDB collection where the data will be stored.
- Write Concern: The level of certainty that the data has been written successfully into the database.
- Batch Size: The number of documents to be written in one batch operation.

G. Data Warehousing



berkas_standar

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
4.10 kB	0	0 B	1	4.10 kB

daftar_kriteria

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	9	82.00 B	1	20.48 kB

dosen

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
20.48 kB	20	121.00 B	1	20.48 kB

kurikulum

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
4.10 kB	0	0 B	1	4.10 kB

Figure 12 Container management system docker

Data from various sources to support business analysis is collected and integrated in this stage. The ultimate goal is to provide a platform for better decision-making. The processed data is stored in the academic_ws database using MongoDB.

H. Data Visualization

The Database Management System used is MongoDB, which contains document-based data. This database management system uses the Container Management System device, namely docker. Figure 12 is a view of the docker management system. "berkas_standar" is a database that contains accreditation standard data, "daftar_kriteria" is a database that contains a list of criteria, "lecturer" is a database that contains lecturer data and "curriculum" is a curriculum database.

This research did not reach the system implementation stage. The focus of this research is on the development of data architecture for universities using a data lakehouse architecture with the final result in the form of the formation of a prototype data lakehouse for university accreditation data by taking data samples from one of the private universities in Indonesia.

4. Conclusion

Based on the research results, it can be concluded that the Lakehouse data architecture model on university accreditation data has been successfully designed to assist the university in developing a plan for developing an accreditation data management system in the future. In the data Lakehouse architecture, storage is integrated into one platform, making it easier and faster to analyze data and make decisions/policies for the university. In addition, integrated data helps universities to minimize expenses. Some of the components used in this research are data sources, data ingestion, data storage, data processing, data analytics, and data visualization. This research provides an overview of the data process in a comprehensive data Lakehouse architecture model, which helps data management to be better organized. Data Lakehouse architecture can be developed by considering better data security aspects, such as the application of encryption and stricter access control.

References

[1] Kemendikbud, "Peraturan Menteri Pendidikan dan Kebudayaan tentang Akreditasi Program Studi dan Perguruan Tinggi," *Menteri Pendidik. dan Kebud. Republik Indones.*, vol. 29, no. 5, pp. 1–31, 2020.

[2] A. P. Studi *et al.*, "LAM - PTKes," 2019.

[3] E. Zagan and M. Danubianu, "Data Lake Approaches: A Survey," *2020 15th Int. Conf. Dev. Appl. Syst. DAS 2020 - Proc.*, pp. 189–193, 2020, doi: 10.1109/DAS49615.2020.9108912.

[4] A. Nambiar and D. Mundra, "An Overview of Data Warehouse and Data Lake in Modern Enterprise Data Management," *Big Data Cogn. Comput.*, vol. 6, no. 4, 2022, doi: 10.3390/bdcc6040132.

[5] B. Inmon, M. Levins, and R. Srivastava, *Building the Data Lakehouse*. 2021.

[6] A. A. Harby and F. Zulkernine, "From Data Warehouse to Lakehouse: A Comparative Review," *Proc. - 2022 IEEE Int. Conf. Big Data, Big Data 2022*, no. January 2023, pp. 389–395, 2022, doi: 10.1109/BigData55660.2022.10020719.

[7] V. Bureva, "Short remarks on index matrices Short remarks on Data Warehouse (DW), Data Lake (DL) and Data Lakehouse (DLH)," pp. 81–105, 2020.

[8] A. L. Antunes, E. Cardoso, and J. Barateiro, "Incorporation of Ontologies in Data Warehouse/Business Intelligence Systems - A Systematic Literature Review," *Int. J. Inf. Manag. Data Insights*, vol. 2, no. 2, 2022, doi: 10.1016/j.jjimei.2022.100131.

[9] D. Jain, "Lakehouse: A Unified Data Architecture," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 3, pp. 881–887, 2021, doi: 10.22214/ijraset.2021.33376.

[10] S. K. Joseph, "Data Lake Model to Modern Educational Organizations," *Int. Res. J. Eng. Technol.*, no. July, pp. 268–276, 2020, [Online]. Available: https://www.dropbox.com/s/35s2aussfhgikv6/20200000%0AData%0ALake%0AModel%0Ato%0AModern%0AEducational%0AOrganizations%0AIRJET-V7I748.pdf?dl=0

[11] B. Chattopadhyay *et al.*, "Shared Foundations: Modernizing Meta's Data Lakehouse".

[12] R. Liu, H. Isah, and F. Zulkernine, "A Big Data Lake for Multilevel Streaming Analytics," *2020 1st Int. Conf. Big Data Anal. Pract. IBDAP 2020*, 2020, doi: 10.1109/IBDAP50342.2020.9245460.

[13] P. Jain, P. Kraft, C. Power, T. Das, I. Stoica, and M. Zaharia, "Analyzing and Comparing Lakehouse Storage Systems," *Int. Conf. Innov. Data Syst. Res.*, 2023, [Online]. Available: https://github.com/lhbench/lhbench.

[14] M. Armbrust, A. Ghodsi, R. Xin, M. Zaharia, and U. Berkeley, "Lakehouse: A New Generation of Open Platforms that Unify Data Warehousing and Advanced Analytics," in *Proceedings of CIDR*, 2021.

[15] S. Park, C. S. Yang, and J. W. Kim, "Design of Vessel Data Lakehouse with Big Data and AI Analysis Technology for Vessel Monitoring System," *Electron.*, vol. 12, no. 8, 2023, doi: 10.3390/electronics12081943.

[16] S. Azzabi, Z. Alfughi, and A. Ouda, "Data Lakes: A Survey of Concepts and Architectures," *Computers*, vol. 13, no. 7, p. 183, 2024, [Online]. Available: https://www.mdpi.com/2073-431X/13/7/183

[17] D. Mazumdar, J. Hughes, and J. Onofre, "The Data Lakehouse: Data Warehousing and More," 2023, [Online]. Available: http://arxiv.org/abs/2310.08697