

# Redback Operations - Requirements Gathering & Options Paper

## Background

Redback Operations is currently without a permanent scalable contemporary data solution.

While workarounds currently exist, such as storing data in GitHub folders or on the personal computers of company members this is far from best-practice and doesn't represent a sustainable longer-term solution.

While previous work has been dedicated to a data warehouse solution and the 'Project 4 - Data Warehousing Team' is underway this trimester, both have given results with a focus on meeting short-term requirements in an ad-hoc fashion. These endeavours brought success for what was required under the constraints of a two-trimester unit and given the nature of the capstone program with the considerable budgetary and limited working hours. It is commendable the progress made so far is of such a high quality.

The purpose of this document is to present information on how data is currently used in Redback Operations and present a set of options which, if executed aim to improve the way data is to be used in the company.

Currently, data is sought out only when it is needed to be accessed, processed, utilized, and then most likely saved or discarded once it is no longer required. Within this process, includes a series of unnecessary steps which can be simplified or reduced, made more efficient, replaced or removed all together with a proper data model, data architecture, engineering, and data-lifecycle.

The first step, and that being the focus of this document and Project 4 is to establish what's required from a potential Data Lakehouse. A proper best-practice data solution should not be acquiring hardware or software without considering and migrating across all records from existing storage solutions and continuing from there. Nor does it involve opening the gates for Junior and Senior company members to spin-up storage space, create and populate their own folders for storage without any guidance or overall modelling and data-lifecycle to ensure efficient use of resources. A Redback Operations data solution must begin with a cycle of planning as with any other project before jumping straight into a solution.

While a data warehouse or data storage solution is a key component of the overall goal, to properly implement a data solution in line with an industry standard would mean a complete overhaul of the existing data processes for Redback (in some cases implementing for a lack of current data processes). This would consist of a body of work more akin to a 'Data Transformation' than a Data warehouse implementation.

While planning and implementing a project of this size given the timeframe would be ambitious for any company. A data transformation would require an introduction of many fundamental aspects typical of a high-functioning data department. This better ensures a solution foundation rooted in industry standards, including appropriate data architecture, data modelling, overall data framework, insights management and data security.

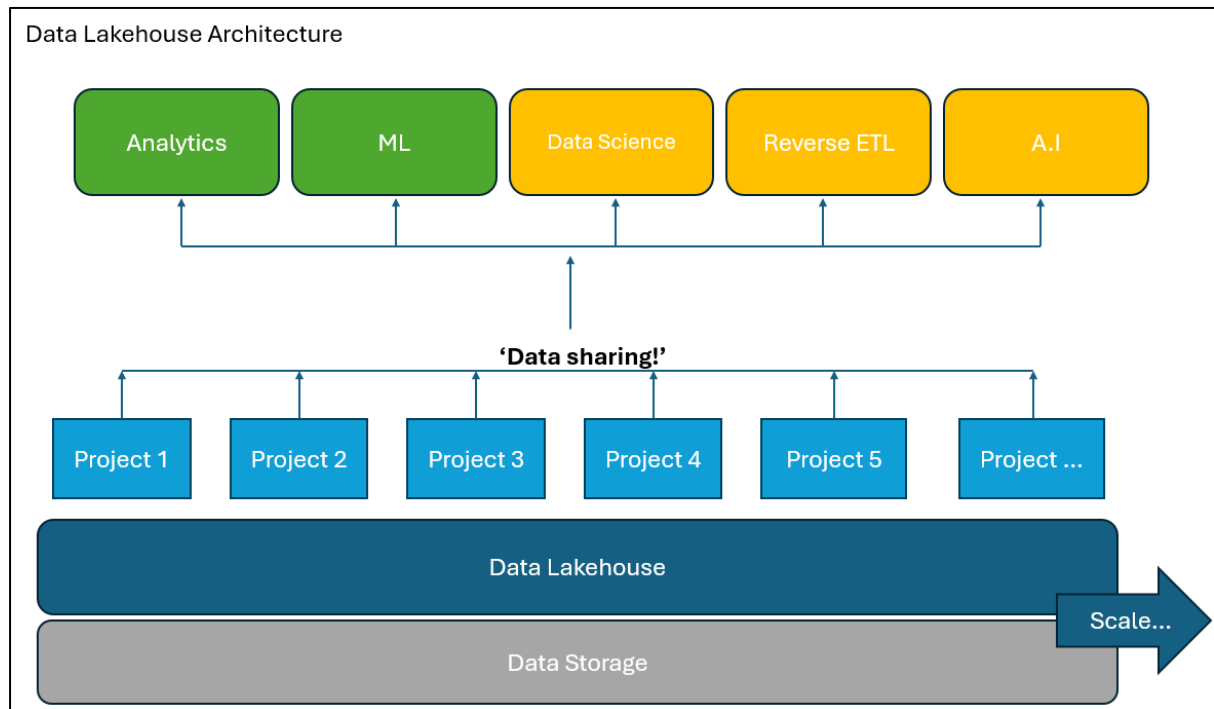
## Table of Contents

Background.....	2
A Note on Data Warehouse .....	4
The Plan .....	5
Requirements Gathering .....	6
Options .....	10
IOMETE.....	11
N8n .....	11
Dremio.....	12
Microsoft Fabric/Azure .....	12
MongoDB.....	13
Apache Hudi.....	13
Data Storage .....	14

## A Note on Data Warehouse

'Data Warehouse Team' is this company's project allocated to a data storage solution, as has been the project objective since its inception.

The project name has been appropriate for general understanding given that practically yes, this project aims to implement a data storage solution, however; if any recommendation of this document is to be approved and bring the project up to industry standards and best practices, the projects objective would be looking to acquire and implement a state-of-the-art Data Lakehouse or similar company-wide data storage solution no longer a standalone Data Warehouse for one project alone.



1. Data Lakehouse Architecture

## The Plan

A proposed data transformation requires input from a wide variety of stakeholders and would require a separate document.

A proposed plan may look like:

### Requirements Gathering

- Requirements gathering process including survey results and Requirements gathering document.

### Assess current infrastructure.

- Gather information on the current data architecture/lack of.

### Options paper, Choosing a platform.

- Storage layer
- Platform
- Options paper document.

### Design Architecture

- Data ingestion
- Storage
- Processing
- Analytics
- Data governance

### Data ingestion

- Part of the platform decision

### Data Storage

- Virtual machine
- Cloud storage

### Data Query and Analytics

- Review tool stack for data analytics, Power BI, Tableau.

### Security and governance

- Define access and encryption policies for sensitive data.

### Training and documentation

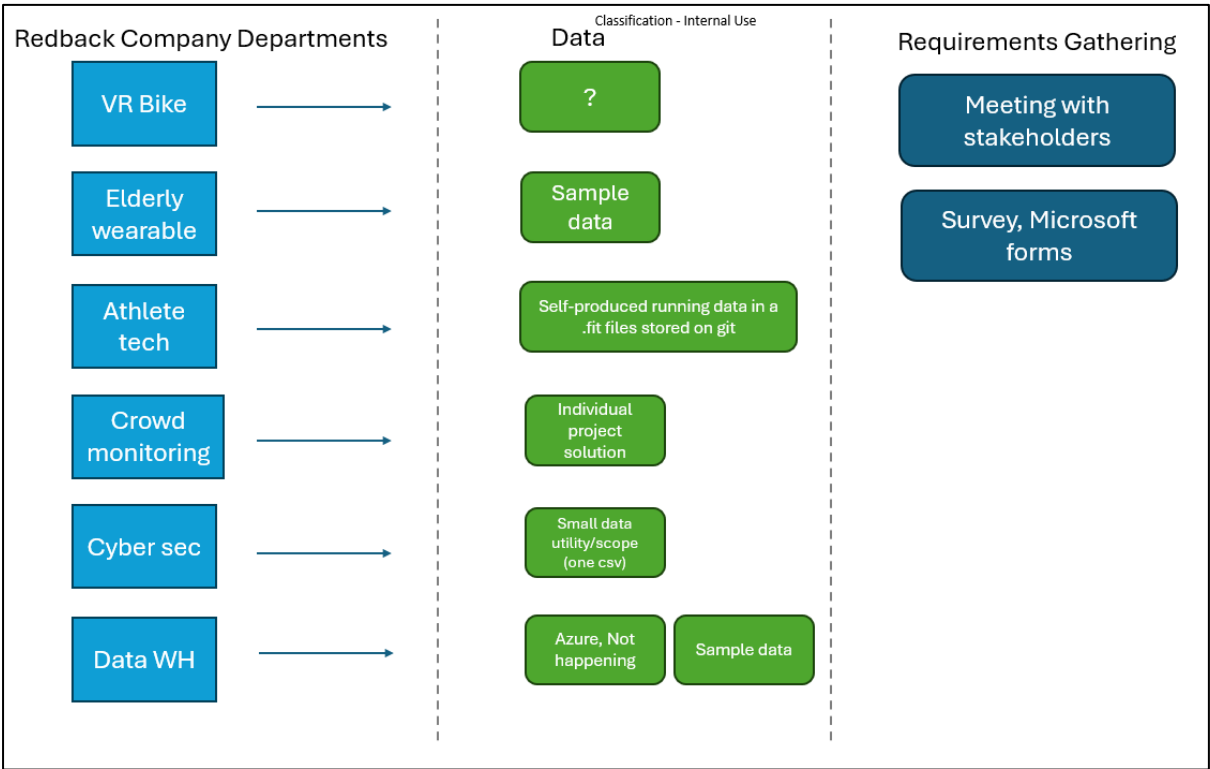
- How-to's and documentation,
- Decision register
- Project retrospective.

## Requirements Gathering

The first stage of the data transformation involves requirements gathering, searching for pain points, and understanding how the company uses its data. Background documentation and consultation with company data stakeholders indicates prior requirements gathering has not been performed beyond a surface level and has been limited to the domain of Data warehouse leaders and some other key stakeholders.

Rigorous requirements gathering allows for an overall understanding of how the company uses its data, this in-turn gives a greater understanding of what is required to improve the data solution and helps to focus the scope when going to market for a SaaS solution or whether there is even need for a solution at all.

So far, as part of the requirements gathering process we have conducted five meetings with leaders from projects in the company.

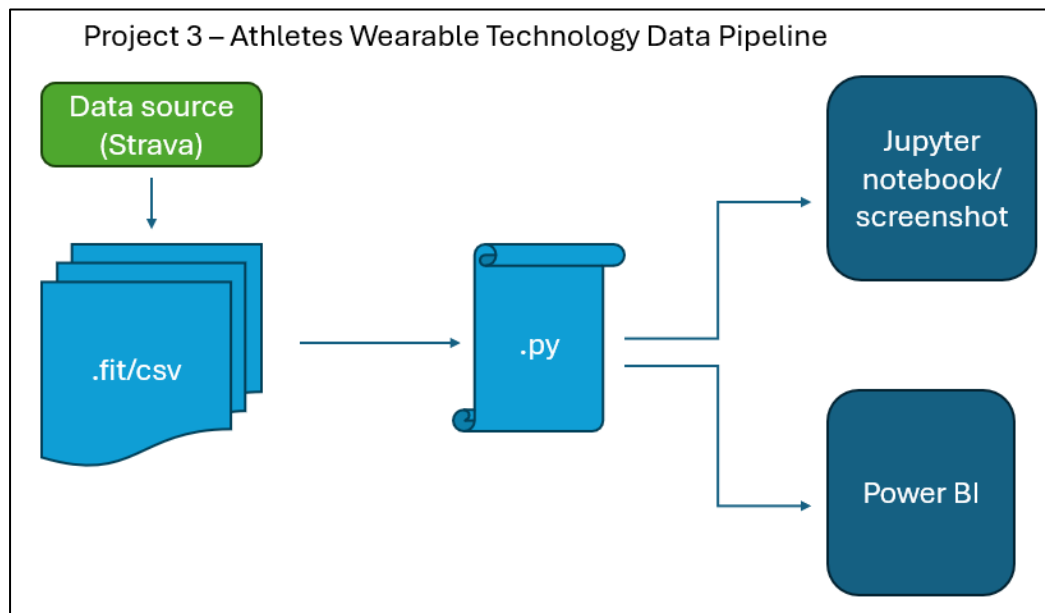


2.0 The current Redback company projects (Trimester 1 2024), their data and the upcoming requirements gathering phases.

## Pain Points

The results of a requirements gathering process at the beginning of Trimester 1 2024 indicate pain points of:

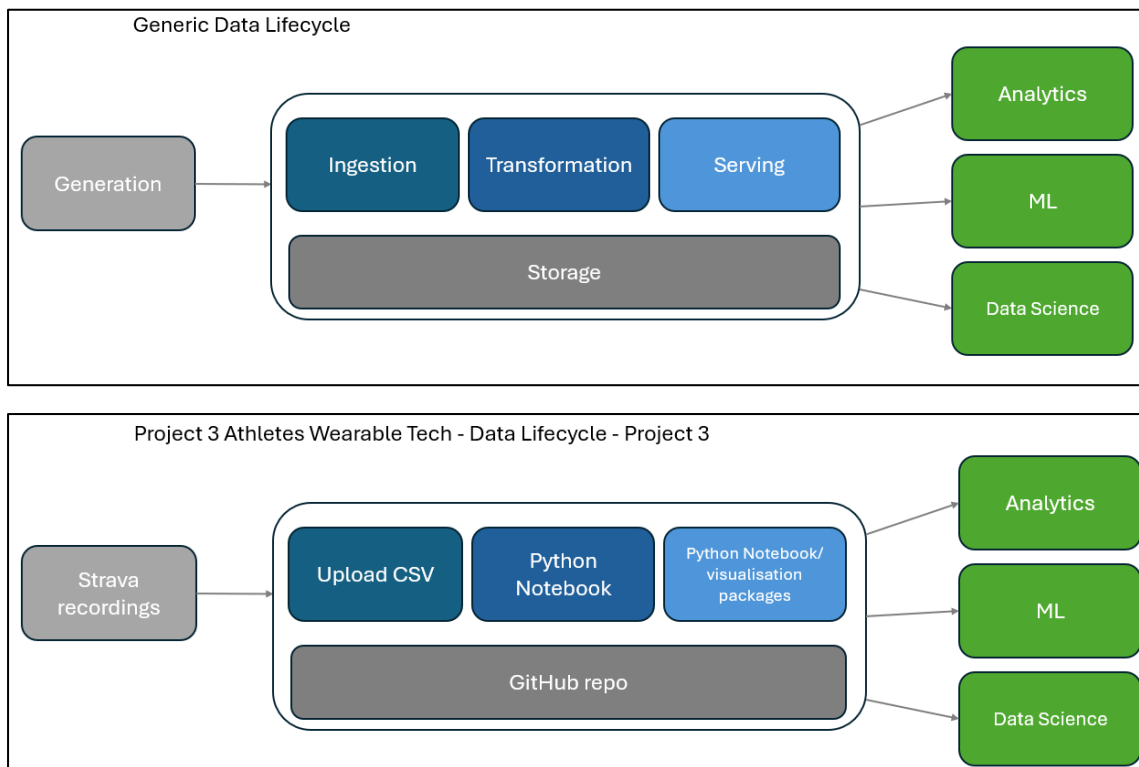
- There currently isn't an easy way to update the dataset for the various data analytics sources. Currently they must go through a GitHub fork/pull request.
- The solution needs to have the capability to store unstructured, NoSQL objects and structured data.
- Limited budget, subject to approval at all phases.
- Preferably a more common language/low code approach for upskilling, due to high student turnover given nature of the capstone unit.
- There is no central 'repository' for storing company wide data, adding steps to potential collaboration. No shared Data Lake that projects can derive insights from.
- Individual user licenses present an obstacle every trimester, given student turnover.



3.0 Project 3 Data Pipeline

## Requirements gathering is an ongoing process.

A second requirements gathering exercise in the form of a survey is planned for Trimester 1 University week 6 (08/04/2024).



### 4.1 and 4.2 Data Lifecycle comparisons.

So far, from the pain points from the interview process alone can be summarized as requirements and favorable key features for a Data Lakehouse/Data solution. These become important to critique the suitability of options when choosing Redbacks Data Lakehouse.

- Large enough to support the current data storage requirements.
- Easily scalable to incorporate future projects and data requirements.
- Unstructured file storage, to support projects with object storage requirements.
- Cost effective, with a free trial or free forever tier to begin with.
- Ease of use and minimal upskilling will be favored. Commonality taken into consideration.
- Contemporary solutions, Redback Operations projects all have state-of-the-art objectives, and a data solution should be no different.
- Preferably an enterprise-wide solution not individual user-based licensing.



## Survey

While company and project leaders have been consulted in meetings, end users will be offered a survey to gather how the majority of company members use data, mainly focused on members who aren't directly involved in a data analysis/data science project. (Consultation of the Epic 2 - Data Analysis group is ongoing)

You can find the results from the week 6 survey [Here](#)

The goal of the survey is to add to the overall requirements gathering process, namely how Redbacks company's and non-data focused users currently use data, given the preliminary requirements gathered.

QuestionsResponses12Settings

### Redback Operations Data Survey

**B I U**

This form is about how YOU use data in Redback.  
'Data' in this case means any information you use for reporting, analysis, modelling in tasks for your project.  
This will help determine changes to data modelling company wide.

Responses are anonymous.

+

📄

Tt

📄

▶

☰

What project are you a part of?

☐ Project 1 - SunCycle SmartBike

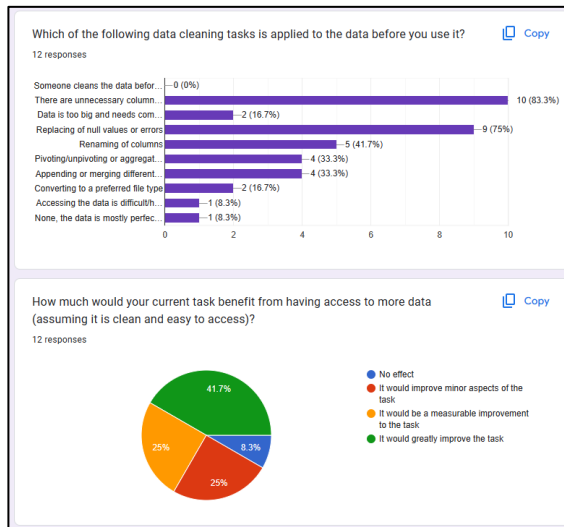
☐ Project 2 - Elderly Wearable Tech Sensor

☐ Project 3 - Athlete Wearable Tech Sensor

☐ Project 4 - Crowd Monitoring and Player Tracking

☐ Cyber Security Team

☐ Data Warehousing Team



How do you currently access data?  
&  
(Optional) if possible provide a link to the data.

9 responses

Access data via GitHub links using Python notebooks.

I clone the repository and download them to my local PC, then setting up a path so it can be easily accessed when I work on a task. (This might be working for me because I'm currently working on only 1 project so the paths do not change very frequently).

From company github

through what has been uploaded to the GitHub, sometimes it is raw data, sometimes it has already been cleaned. and some data does not show all of the data but instead shows only key findings.

Data has been converted to CSV files and included in GitHub.

Through git hub

Data is available in Company GitHub repository

By loading it to python notebook using GitHub link of the file

## Options

The following options in this document represent a result of a formal process of requirements gathering through questioning and interviewing key stakeholders in Redback to assess pain points as well as researching industry leaders and trusted comparison companies, consultation from outside sources including industry professionals and individual set-up/testing each technology.

The six options in this document are the remainder of an initial field of which each option to qualify was required to meet at least a majority of requirements before each technology was tested to examine a potential use case further.

A final recommendation is planned to be provided after the results of the Requirements survey examined and the larger requirements gathering process concluded to maximize the information in making any decision.

### Summary table of Data Lakehouse options

Software	Ease of Access	Data Storage	Price	Learning Curve	Open Source	Key Features
<i>IOMETE</i>	Ok	Cloud	Free option	OK	Yes	Security and Data catalog
<i>N8n</i>	Very Good	Cloud/VM	\$22 pu/pm	Very Easy	Yes	Strong Git compatibility
<i>Dremio</i>	Good	Cloud/VM	Free/low	OK	Backend Yes	Fast
<i>Microsoft Fabric</i>	Very Good	Cloud	\$8,409 pm	Easy	Yes	Every data tool available
<i>MongoDB</i>	Ok	Cloud/VM	Free/\$25 pm	OK	Only in paid	JSON based
<i>Apache Hudi</i>	Very Poor	Cloud/VM	Free	Very Hard	Completely	Completely free and open source

\*\*all prices USD

## IOMETE

### IOMETE

**Ease of access:** Requires a sign-up through email

**Data storage:** Requires third-party.

**Pricing:** 'Free forever'

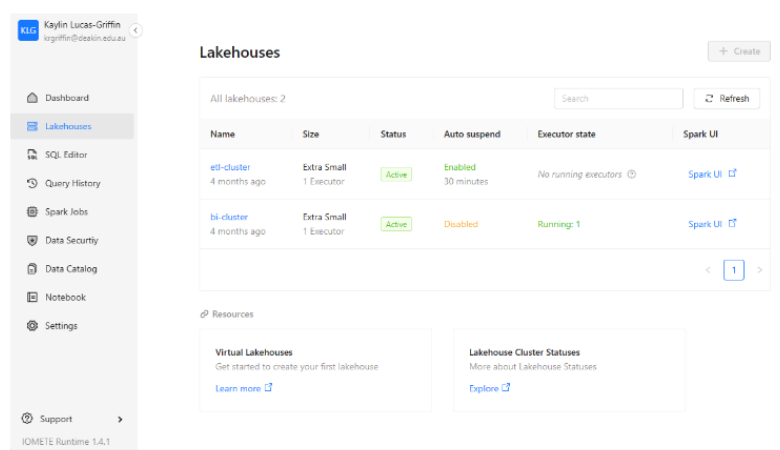
**Learning Curve:** Probably moderate, lack of established community

**Open Source:** Yes


**Licensing:** No

**Compression Tool:** No

**Key features:** Built-in security and data catalog



Iomete is a relatively new, free, open-source Data Lakehouse platform built on Apache Spark. In Operation Redbacks case, it would exist on top of a separate data storage platform. Our best use case would be underpinning with preferably a cloud-based storage solution to take advantage of Iomete's modern features, with that in mind it also has a stand-alone on-prem offering, which conveniently is 'Free forever'. GitHub integration wasn't achievable in testing, but is a big part of their overall sell, so it's assumed it is more than capable. A good all-rounder. Its key advantages are: The 'Free forever' tier and that it's brand new, with most of the required features, quick in comparison set-up time, open source and lack of licensing as far as can tell. Data Catalog is a bonus and so is built in security, provided there is scope to use it. Unfortunately, being new it's user-base is niche and would mean limited community support and steeper learning curve than some of the other options.



## N8n

### n8n

**Ease of access:** Requires a sign-up through email

**Data storage:** Requires third-party cloud/virtual machine.

**Pricing:** Free trial expiry

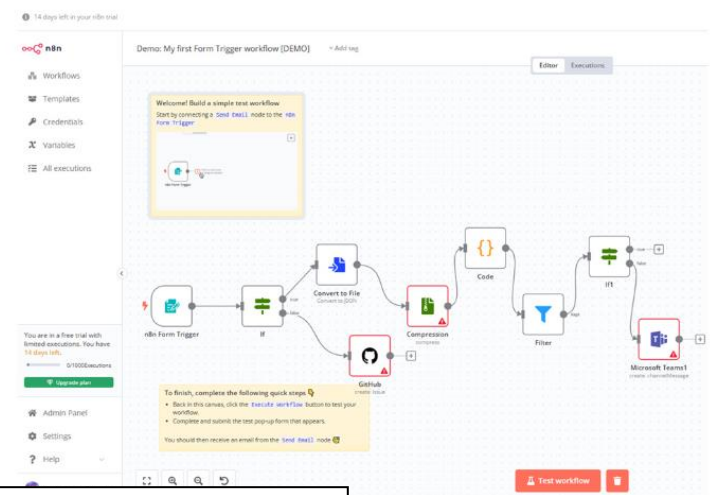
**Learning Curve:** Easy, low-code environment

**Open Source:** Yes


**Licensing:** No

**Compression Tool:** Built-in compression workflows

**Key features:** The most Git compatible, works with cloud or on-prem



N8N is a low-code visual user interface-based Data-pipeline and process automation tool with a focus on data lifecycle and ingestion. The GUI is similar to Microsoft's Power Automate, where pipelines are connected through a low-code visual workflow builder. It isn't primarily designed for data storage or data warehousing and is therefore a stretch to be considered as a single technology for Redbacks data Lakehouse platform without a separate data storage/Lakehouse tool as part of a combined stack. Because it is open-source and low-code it offers a middle ground between back-end customization and ease of use. N8n's big sell is the Git compatibility, it shines in this regard. As well as probably the easiest to use. Quick start-up and has almost no upskilling. Might not have enough Data Lakehouse specific traits as it's contemporaries and is a bit pricey too with approx. \$35 AUD per user for the base tier.



## Dremio

### Dremio

**Ease of access:** Offers easy access to data with GitHub integration.

**Data storage:** Requires third-party Cloud or Server

**Pricing:** Free trial into low-cost model

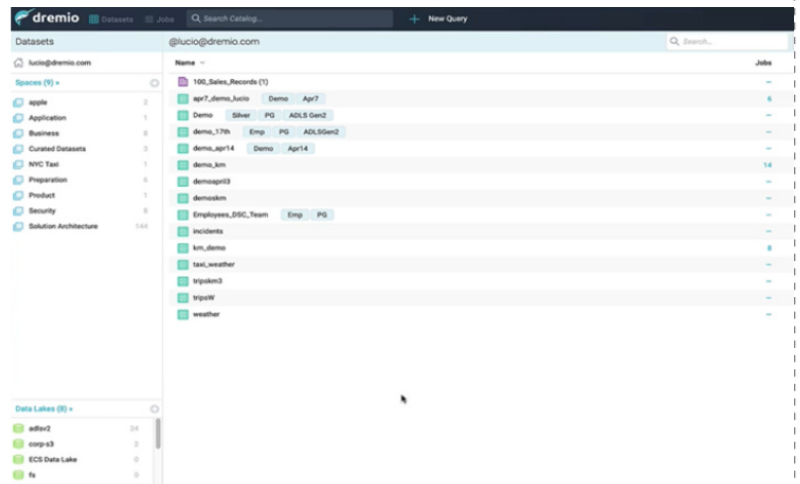
**Learning Curve:** Moderate, SQL based, Apache stack.

**Open Source:** Backend is open source.

**Licensing:** No

**Compression Tool?** Built in, as part of parquet storage option.

**Key features:** Unsure about suitability to store unstructured data.



Dremio has a larger focus on ease of access with self-service analytics. This would make accessing for new students easier. the Dremio application requires self-hosting through Docker and this adds an extra step to set-up for whoever becomes admin in future Trimesters. Unfortunately, Dremio is only available to set-up on a Linux operating system if on-prem/server implementation so a use case would require cloud storage most likely. Performance is comparatively fast to the other options because of its Data Lakehouse architecture with Apache Arrow, Iceberg and spark under the hood and focus on query speed and bandwidth which is a big requirement for Project 4 and any IoT projects that require streaming data instead of batch processing.



## Microsoft Fabric/Azure

### Microsoft (Fabric/Azure)

**Ease of access:** Subscription, Workspace based

**Data storage:** Requires third-party Microsoft Azure

**Pricing:** Tiered, in Redback Ops case: 8,409.60 per month

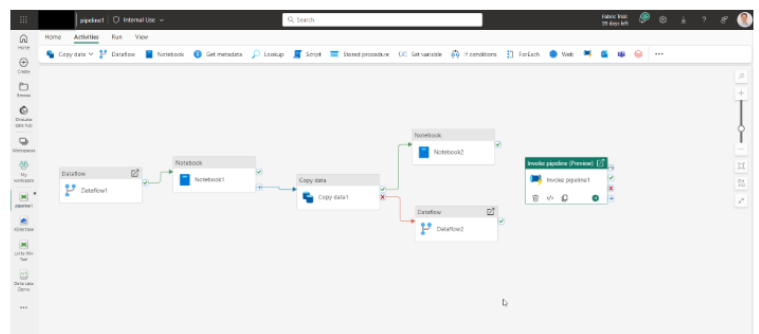
**Learning Curve:** Microsoft platform, low-code options

**Open Source:** No

**Licensing:** No, enterprise-wide subscription

**Compression Tool:** Allows for Parquet files and some in Azure built-in compression.

**Key features:** Microsoft Purview, Microsoft's data security and governance tool



Microsoft's Data Lake offering comes in the form of Microsoft Fabric. It represents an umbrella of the Microsoft end-to-end data stack including a tool for every step of the data pipeline and ETL process, in-built data integration, specific data engineering tools, data science, data warehousing, real-time analytics, complete data platform and dedicated Data Lake as well as full enterprise level Power BI subscription for any and all users on the tenant. Naturally, all this capability comes at an equally large price, for Redbacks case acquiring fabric in a F64 capacity which would comfortably cover all requirements costs \$8,409.60 USD without added One Lake storage, firmly out of the company price range. This option is included because a scenario does exist where the 60-day trial could be used until expiry then shifted over to another leader/student, theoretically in perpetuity or for as long as the current Fabric trial is on offer in the current form. Fabric has low-code functionality and is cloud based.



## MongoDB

### Mongo DB

**Ease of access:** Provides relatively easy access to data, but GitHub integration might require additional setup.

**Data storage:** Requires virtual machine or cloud hosting third-party.

**Pricing:** Free tier with mongo DB Atlas

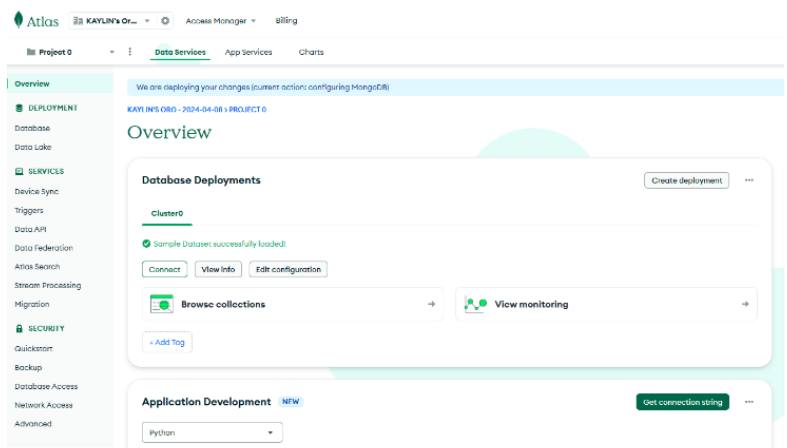
**Learning Curve:** Moderate to high, especially for students new to NoSQL databases.

**Open Source:** Yes

**Licensing:** Sign-up required but easy to get started

**Compression Tool?** Not specified

**Key features:** Designed to unstructured data in the form of JSON, Established name in the database industry, good size community.



MongoDB works well with on-prem data storage and does all the basics well. It is designed to store unstructured data in the form of JSON which is an advantage for Redback operations need for flexible data type requirements. It has long been considered one of the better Database solutions for querying data and has its own MongoDB query language, which is relatively easy to pick-up.

It is a more traditional data warehouse than enterprise-wide data Lakehouse solution. Instead of its own Data Lakehouse, MongoDB suggests integrating with Databricks, a separate Data Lakehouse platform. It has a large community for troubleshooting and setup and the company and product have been around forever, it's not free, but integrates well with on-prem/virtual machine, with some nice added features for data engineering.



## Apache Hudi

### Apache Hudi

**Ease of access:** Difficult, no user interface, completely code based.

**Data storage:** Requires third party, Hudi itself, scalable.

**Pricing:** Open source, typically free to use.

**Learning Curve:** Advanced, no low-code option, uses SQL-like querying.

**Open Source:** Full open source

**Licensing:** Nothing, none.

**Compression Tool?** Supports some compression formats.

**Key features:** Completely open source and raw,

```
# pyspark
columns = ["ts","uid","rider","driver","fare","city"]
data = [(1695159649087, "334e26e9-8355-45cc-97c6-c31daf0df330", "rider-A", "driver-K", 19.10, "san_francisco"),
        (1695091554788, "e96c4396-3fad-413a-a942-4cb36106d721", "rider-C", "driver-M", 27.70, "san_francisco"),
        (1695046462179, "9909a8b1-2d15-4d3d-8ec9-efc48c536a00", "rider-D", "driver-L", 33.90, "san_francisco"),
        (1695516137016, "e3cf430c-889d-4015-bc98-59bdce1e530c", "rider-F", "driver-P", 34.15, "sao_paulo"),
        (1695115999911, "c8abbe79-8d89-47ea-b4ce-4d224bae5bfa", "rider-J", "driver-T", 17.85, "chennai")]

inserts = spark.createDataFrame(data).toDF(*columns)

hudi_options = {
    'hoodie.table.name': tableName,
    'hoodie.datasource.write.partitionpath.field': 'city'
}

inserts.write.format("hudi"). \
    options(**hudi_options). \
    mode("overwrite"). \
    save(basePath)
```

Apache Hudi is a completely free and open-source data Lakehouse framework. It is the newest option in the list, only in version 0.14 and only recently left beta. It was designed with transactional data in mind which isn't relevant to Redback however, Hudi has ample streaming and data Lakehouse use cases or any conceivable use case, if you can engineer it. It allows for Parquet files, excellent for compression and supplies its own out-of-the-box tools for ingesting and ETL. This is an interesting option; the cost effectiveness and access are unmatched and the suite of services still competitive. However, being a community product makes it volatile, buggy, and prone to large changes without notice, this adds to the operational complexity and therefore an advanced learning curve, the interaction is entirely code based and from research not ideal for a windows machine. It doesn't even have a UI. It would require upskilling for an even a capable data engineer. Not to mention the obvious data security concerns.



## **Data Storage**

Data storage for the initial implementation of the Data Lakehouse platform will be the Deakin virtual machine, this will operate as BareMetal on-premises storage. In future Trimesters there is scope to secure funding for a cloud storage solution.