REFERENCE : D04-STATSBDA2.0-PI-SDS-02

D04
SYSTEM DESIGN SPECIFICATION DOCUMENT
(SDS)

PRICE INTELLIGENCE (PI) for STATSBDA 2.0

AGENCY NAME	**:**	**DEPARTMENT OF STATISTICS MALAYSIA (DOSM)**
PARENT AGENCY NAME	**:**	**MINISTRY OF ECONOMY**
DOCUMENT DATE	**:**	**25 APRIL 2025**
DOCUMENT VERSION	**:**	**V2.0**

i. Document Description

This document outlines the module System Design Specification (SDS) for the Price Intelligence (PI) module. It serves as a technical reference for architects, engineers, and infrastructure team from Department of Statistics Malaysia (DOSM) who will be involved in the design, deployment, and management of Price Intelligence (PI) module.

ii. Document Review and Verification

This section is for the officers responsible for reviewing and verifying the information contained in this document.

Document Review

Review by	Position	Signature	Date of Review
Siti Faizah Hanim Md Matar	Ketua Penolong Pengarah, Bahagian Perangkaan Harga, Pendapatan dan Perbelanjaan, Jabatan Perangkaan Malaysia.		
Ts. Liyana Safra Zaabar	Pegawai Teknologi Maklumat,Bahagian Pengurusan Maklumat, Jabatan Perangkaan Malaysia.		
Noradilah Adnan	Penolong Pengarah, Core Team Analitik Data Raya, Jabatan Perangkaan Malaysia.		

Document Verification

Verified By	Position	Signature	Date of Approval
Jamaliah Jaafar	Ketua, Core Team Analitik Data Raya, Jabatan Perangkaan Malaysia.		
Ts. Norhasniah Ab Aziz	Pengarah Kanan, Bahagian Pengurusan Maklumat, Jabatan Perangkaan Malaysia.		

Verified By	Position	Signature	Date of Approval
En Mohd Yazid Kasim	Pengarah Kanan, Bahagian Perangkaan Harga, Pendapatan & Perbelanjaan Jabatan Perangkaan Malaysia.		

iii. Document Control

This section outlines the version number, date, summary of amendments for the document.

No. Version	Date	Description	Revised By
1.0	23 December 2024	Document Creation	Muhammad Naqiuddin Bin Ahmad
2.0	23 April 2025	Document Review & Correction	Muhammad Naqiuddin Bin Ahmad

.

iv. Table of Contents

v. Table of Figures

vi. Table of Tables

vii. Acronym and Definition

a. Acronym

The below table contains the list of acronyms :-

Acronym	Description
DOSM	Department of Statistics Malaysia
MCOICOP	Malaysia Classification of Individual Consumption According to Purpose
PI	Price Intelligence
ICP	International Comparison Program
CPI	Consumer Price Index
AAD	Azure Active Directory
ABS	Azure Blob Storage
MAS	Malaysia Airline System
KTM	Keretapi Tanah Melayu
LLM	Large Language Model

Table 1 - Acronyms

b. Definition

The below table explain the definitions of terms or phrases used in the document.

Term/Phrase	Definition
Data Acquisition	The process of gathering raw data from various sources, such as websites, APIs, or databases, to capture relevant information for analysis.
Data Collection	Collecting specific data points from acquired sources, structuring them to form a unified dataset. This includes steps like web scraping and other methods of data extraction.
Data Cleansing	Identifying and rectifying errors or inconsistencies in the dataset, such as handling missing values, duplicates, or inaccuracies, to ensure data quality.
Data Transformation	Modifying the collected data into a standardized format, including type conversions, formatting, and aggregating fields, to make the data suitable for analysis.
Data Enrichment	Enhancing the dataset by integrating additional context, such as geographic, demographic, or external datasets, to provide a more comprehensive view.
Data Storage	Storing the structured and enriched data in a database or data warehouse, ensuring it is accessible, secure, and scalable for further analysis and retrieval.
Data Analytics	Analyzing the stored data using statistical and computational methods to derive insights, identify patterns, and support data-driven decision-making.
Data Visualization	Representing the analysed data through charts, graphs, and dashboards to communicate insights effectively and support easy interpretation of complex data patterns.
Large Language Model	Advanced artificial intelligence system trained on vast amounts of text data to understand and generate human-like language. These models, such as GPT-4, use deep learning techniques, particularly. transformer architectures, to predict and generate coherent text based on input prompts.

Table 2 - Definitions

viii. References

This section outlines the sources of references, and the terms used in the preparation of this document.

1. MCOICOP

   Data Catalogue :
   MCOICOP 2024/2025 updated version document uploaded by DOSM.

2. ICP

   a. International Comparison Program :
      https://www.worldbank.org/en/programs/icp

   b. International Comparison Program for Asia and the Pacific :
      https://icp.adb.org/overview

   c. Catalogue of Housing Rental

   d. Operational Guide for Housing Rental Survey

   e. Catalogue of Household Product

   f. Catalogue of Machinery and Equipment

   g. Catalogue of Construction

1. Introduction

1.1. STATSBDA 2.0 Overall Architecture

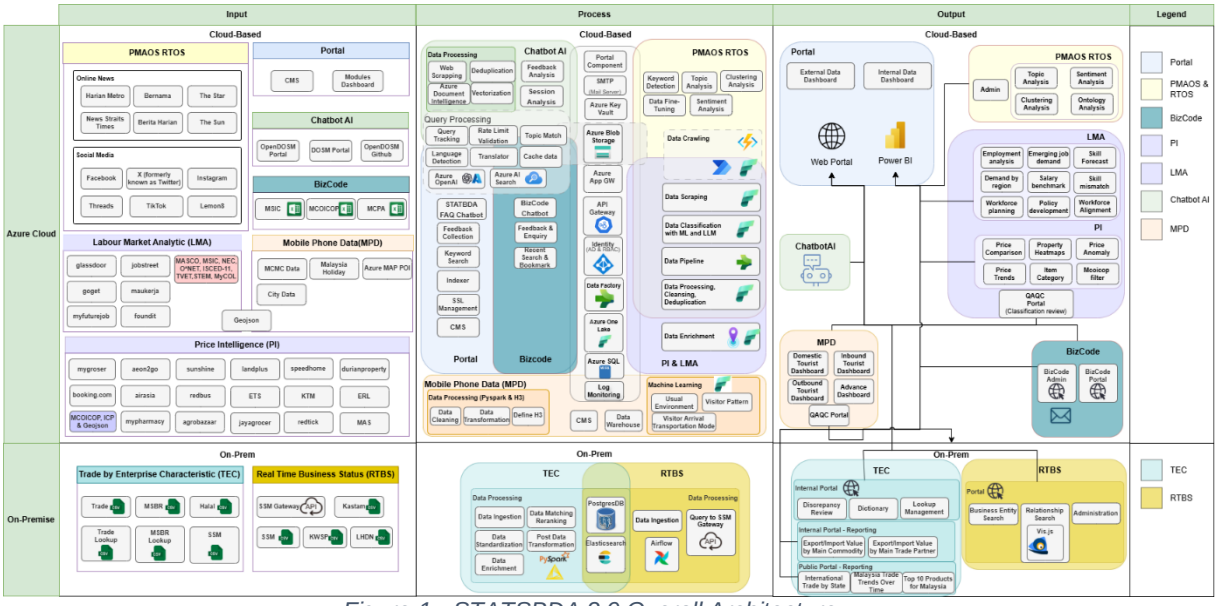


Figure 1 - STATSBDA 2.0 Overall Architecture

The overall STATSBDA architecture comprises 9 modules operating across cloud-based and on-premises environments, enabling seamless data ingestion, processing, and output delivery.

- Input represents various data sources required by each module.
- Processing involves Azure components used within respective modules, with shared services such as Azure Key Vault, Azure App Gateway, Azure AD, Data Factory, Azure OneLake, Log Monitoring and SMTP. On-premises processing relies on PostgreSQL and Elasticsearch, shared by both TEC and RTBS.
- Output consists of insights delivered within each module. Most of the dashboards are built using Power BI, while MPD Dashboard, StatsBDA Portal and BizCode Admin Portal are designed as web-based portals. The chatbot AI provides an embedded code for seamless integration into external portals.

The 9 key modules in this architecture are:

1. BizCode
2. StatsBDA Portal
3. Mobile Phone Data (MPD)
4. Chatbot AI
5. Labor Market Analytic (LMA)
6. Price Intelligence (PI)
7. Trade Enterprise Characteristic (TEC)
8. Public Maturity Assessment on Official Statistics (PMAOS) & Real-Time News on Official Statistics (RTOS)
9. Real-Time Business Status (RTBS)

This document will focus on the Price Intelligence module, detailing its design, components, workflow, and data integration.

1.2. Design Purpose

The purpose of the Price Intelligence Module's design is to provide a robust, scalable, and efficient framework for the automated acquisition, storage, and analysis of pricing data across multiple sectors. This design ensures the delivery of actionable market insights to DOSM, enabling data-driven decision-making and strategic market positioning.

By aligning with the business objectives outlined in the BRS and Software Requirement in the SRS, the module's design emphasizes the creation of a system that is both user-centric and adaptable to evolving market and regulatory landscapes.

1.3. Design Scope

The scope of the Price Intelligence Module design focuses on achieving seamless integration of data collection, processing, analysis, and reporting functionalities. The business scope specified based on the BRS document are:

I. Market Coverage Across Key Sectors
II. Automated Data Acquisition and Analysis
III. Enhanced Decision-Making Through Analytics and Reporting
IV. Regulatory and Compliance Considerations
V. Flexibility and Scalability

The Design Scope adheres to the business scope by addressing the key aspects:

I. Market Coverage Across Key Sectors

a. Aims to ensure comprehensive extraction of price data from E - Commerce, property, and transport websites to enable analysis across diverse market sectors.
b. Design Implementations:
   - Utilize Power Automate selectors for precise extraction of dynamic and static web content from targeted sectors websites in E - Commerce, Property, and Transportation categories.
   - Employ Microsoft Fabric Notebooks to integrate APIs and parse E-Commerce, property, and transportation categories website elements using Beautiful Soup for broader data coverage.
   - Ensure periodic updates to the data extraction logic to align with evolving website structures and market trends.
   - Provide support for additional market sectors through modular enhancements to the data acquisition pipelines by changing the setting in power automate or fabric notebook.
c. Following website are in scope as the new proposed data source to be scrape and developed, including but not limited to:

i. E - Commerce
   - myaeon2go.com
   - agro bazaar.com.my
   - bigpharmacy.com.my
   - hlkonline.my
   - Lotus.com.my

ii. Transportation
- Airpaz.com
- Booking.com
- Agoda.com
- Airasia.com
- redbus.my
- busonlineticket.com
- ktmb.com.my

iii. Property
- Durianproperty.com.my
- iRumah.co (Formerly landplus)
- Speedhome.com
- Iproperty.com.my
- Propertyguru.com.my

d. The following list of website have existing ongoing running crawler which will remain active and to be migrated to new crawler in statsbda 2.0.
- fama.gov.my
- jayagrocer.com
- mygroser.com
- myaeon2go.com
- shop.redtick.com
- lotuss.com
- iproperty.com.my
- sunshineonline.com.my
- mydin.my
- directd.com.my
- sogo.com.my
- lazada.com.my
- lelong.com.my
- mymotor.my
- satugadjet.com.my
- senheng.com.my
- zalora.com.my
- mudah.my
- motortrader.com.my

II. Automated Data Acquisition and Analysis.

a. To implement a fully automated ETL pipeline to acquire, process, and analyse data efficiently.

b. Design Implementations:
- Automate data scraping using Power Automate and Fabric Notebooks for real-time and scheduled data extraction.
- Store scraped data in Azure Blob Storage temporarily in CSV format before ingesting it into Microsoft Fabric Lakehouse/OneLake.
- Clean, format, and standardize data using Python scripts executed within Fabric Notebooks.
- Classify cleansed data into MCOICOP and ICP codes using a machine learning model built in Python.
- Transfer processed data to Fabric Warehouse in Delta table format to support downstream analytics.

III. Enhanced Decision-Making Through Analytics and Reporting

a. Enable stakeholders to derive actionable insights through interactive dashboards and reports.

b. Design Implementations:
   - Integrate cleansed and classified data with Power BI to build semantic models for advanced analytics and visualization.
   - Design Power BI dashboards tailored to display trends, comparisons, and predictive insights across sectors.
   - Ensure reports are dynamic, customizable, and accessible to stakeholders with varying levels of technical expertise.

4. Regulatory and Compliance Considerations

a. Adhere to industry and legal standards for data processing and governance.
b. Design Implementations:
   - Implement Microsoft Purview for end-to-end data governance, including lineage tracking, classification, and access policies.
   - Ensure compliance with data privacy and security standards through data masking and anonymization techniques during processing.
   - Use Azure Entra ID to manage role-based access control (RBAC) and ensure secure data access.
   - Maintain a detailed audit trail of data operations and changes to meet regulatory requirements.

5. Flexibility and Scalability

a. Design a solution capable of adapting to evolving business needs and handling increasing data volumes.
b. Design Implementations:
   - Leverage Microsoft Fabric Lakehouse/OneLake for scalable storage and compute resources.
   - Design modular ETL workflows and tools such as Fabric Datafactory connector and Fabric Notebook to accommodate new data sources and analysis requirements and allowing reconfiguration of ETL workflow.
   - Employ delta table formats in Fabric Warehouse to support incremental updates and high-performance querying.
   - Integrate monitoring and alerting using Power Automate and Fabric Monitoring to identify and address bottlenecks promptly.
   - Build the architecture to be cloud-native, ensuring seamless scaling to support higher data loads and additional processing tasks.

1.4. List of System Actors

The list of system actors of the Price Intelligence system includes:

1. System Administrator

a. The responsibilities include configures the module settings, manages user access, and ensures the system operates efficiently.
b. They may also monitor system performance and address any technical issues.

2. Data Analyst
    a. The responsibilities include utilizes the collected price data for analysis.
    b. They generate reports, identify trends, and derive insights that inform business decisions.
    c. Analysts may interact with the analytics dashboard to visualize data.

3. Web Crawler
    a. The responsibilities include navigating and scraping data from targeted websites.
    b. This actor is not human but is crucial for executing the crawling and data extraction processes.
4. Data Storage System
    a. The responsibilities include securely storing the scraped and normalized data.
    b. It interacts with other actors for data retrieval and management but operates autonomously.
5. Compliance Officer
    a. The responsibilities include ensuring that the system adheres to legal and ethical standards related to web scraping and data privacy.
    b. This actor monitors compliance with regulations and best practices.

6. IT Support
    a. The responsibilities include providing technical support and maintenance for the system, troubleshooting any issues that arise.
    b. They may assist in system upgrades and integrations.

7. Business Stakeholders
    a. The responsibilities include using the insights from the price data to make inform strategy and operations.
    b. They may interact with reports and analytics to guide pricing strategies or market positioning.

2. Price Intelligence Design

Design for Price Intelligence module focuses on solution that allows for gathering price data from webpage, cleansing, and classifying the data according to MCOICOP and ICP classifications.

Based on SRS the stated features and requirement to be guideline for the design architecture is as follows:

i. Crawling mechanism:
    a. Daily crawling of selected online sources to gather real-time pricing data.
    b. Support for crawling various categories, including groceries, electronics, and property listings.
    c. An alerting mechanism to notify users of any crawling issues.

ii. Data Management and ETL Processes
    a. Integration with Microsoft Fabric for data storage and processing.

b. Efficient ETL (Extract, Transform, Load) processes to manage the data pipeline from crawling to analysis.
c. Implement deduplication to ensure unique entries, especially for identical products from same sources and day

iii. Data Cleaning and Transformation

a. Procedures to clean the data, including handling missing values, normalizing price formats, and calculating price ratios.
b. The system will track historical data to identify trends and average pricing over specified periods.

iv. Dynamic Data Storage

a. The ability to store historical data for trend analysis while ensuring old data is preserved for reference.
b. The system shall provide the capability to use the updated version MCOICOP list as needed.

vi. Data Classification

a. Implement a classification system based on the Malaysian Classification of Individual Consumption According to Purpose (MCOICOP), structured into 13 groups (e.g., food, alcohol, meat).
b. Each item will be identified by 8-digit code that includes specific brand information and detailed product specification/condition.
c. Implement a classification system based on ICP codes.

vii. Geocoding

a. Geocoding functionalities primarily for property-related data to enhance location-based analysis.
b. DOSM already have a Postal Address to Lat/Long (Latitude/Longitude) database that can be shared to convert Postal Address to Lat/Long (Latitude/Longitude) subject to the DOSM management approval.

viii. Dashboard and Reporting

a. A user-friendly dashboard for visualizing price trends and inflation metrics for CPI and ICP data.
b. Time-series data updates shall be performed daily before 9 AM, ensuring timely insights.

ix. User Configuration and Notifcations

a. Users can configure crawling parameters and alert thresholds for price discrepancies.
b. Notification system to inform users of significant changes in data or system issues.

3. Design Architecture

Overall architecture of Price Intelligence comprises of data flow process which starts from acquisition, staging of the raw data, processing and classification of data and last staging place for analytics and structured data.

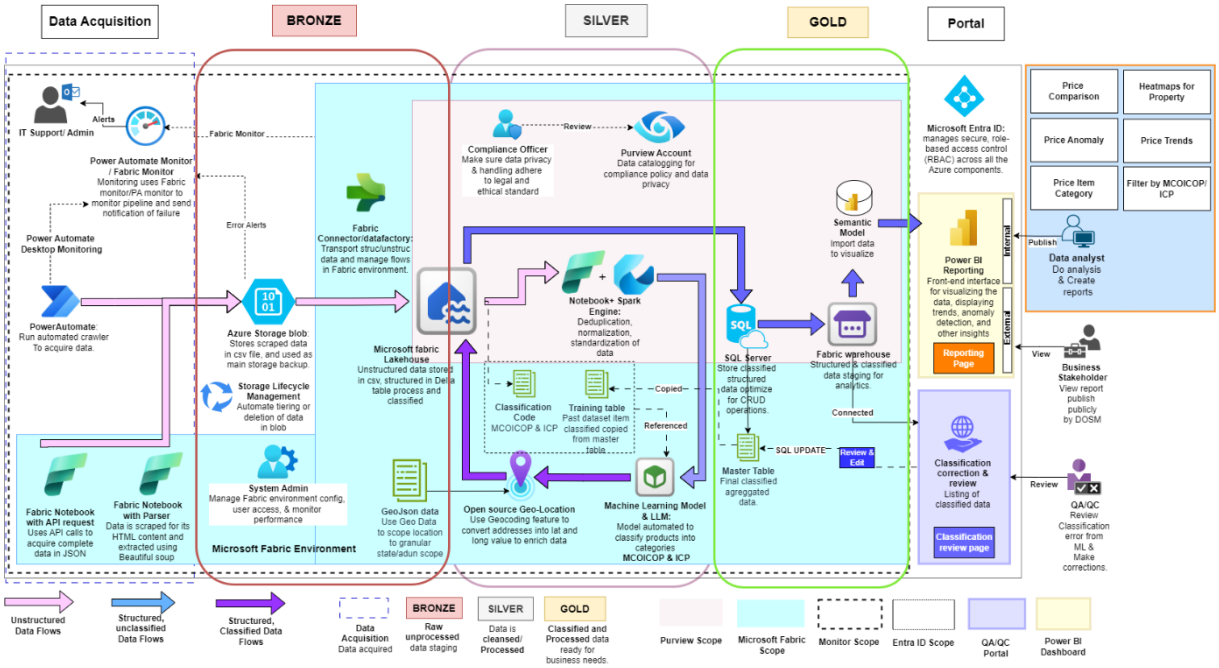


Figure 2 - Overall architecture of Price Intelligence (PI).

Further description of architecture is as follows:

Data Phase	Services involved	Description
Data Acquisition	• Power Automate • Power Automate Desktop monitor • Microsoft Fabric monitor • Fabric Notebook	During this phase data is acquired using 3 method of data acquisition, namely through **Power Automate desktop with selector**, using **API request** and also scraping using **beautiful soup with Fabric notebook.** Data acquired using power automate utilize selector with user interface to gather price data. Data acquisition through API request utilize website underlying API that is used to list price data in website. API can be called to gather complete data. Data acquired using beautiful soup and notebook is done by parsing website elements using Beautiful Soup Python package to gather price data. All fails, complete, past run is monitored using **Power automate desktop flow monitor** and **Fabric monitoring** respectively and fail or complete runs can be configure to sent alerts to users.
Bronze Phase (data Staging and Storage)	• Blob storage • Data Factory/Pipeline	Unstructured and structured data acquired from Crawling and Scraping phase is

	• Fabric Lakehouse	stored in **Blob storage** temporarily as csv file. Logs are also stored in blob in a form of unstructured data. Data is formatted and transported using **Fabric data factory/pipeline**. Data is stored into Fabric Lakehouse as place for data staging, for structured data. To be used further for classification, categorization, and cleaning.
Silver Phase (Data cleaning, processing, and classification)	• Fabric notebook with spark engine • Machine learning model and Fabric LLM	Data is further cleaned, standardized and processed using Python automated run code inside fabric notebook with underlying spark engine. Data is further classified and categorized with **MCOICOP and ICP** reference by using Python and Machine learning model and Large Language Model (LLM) Cleaned and classified data stored as delta table format with T-SQL capability to run SQL query.
Gold Phase (Data aggregation and enrichment)	• SQL Server • Data warehouse • Power BI semantic model • Power BI	Cleaned and classified data in delta table format is imported into SQL server, which will be used by Internal QA/QC portal for classification review and changes. Data is then imported into warehouse as final staging for analytics. Final data inside warehouse is imported to a Power BI semantic model for data analytics and visualization.

Table 3 - Overall Architecture Description

3.1. Data source & Crawling

3.1.1. Crawling and Scraping workflows for websites.

Generally, data is acquired using two (2) services, through **Power Automate** and **Fabric Notebook with Python code** elaborated below with each of its main features:

i. Power Automate
Desktop automated RPA tool that uses selector to interact and gather data from webpage with user interface.

Service	Method	Main Feature
Power Automate	**Method 1 PA:** Use **Power Automate desktop** to extract data from webpage using selector with user interface.	User interface selector to select website elements and parse the website elements for useful data.

Table 4 - Power Automate method and feature

ii. Python with Fabric Notebook

Run scheduled Python code in notebook with underlying spark engine to execute code to gather and scrape data from websites.

Service	Method	Main Feature
Fabric Notebook with Python and request	**Method 1 Python:** Use underlying website **API calls** to extract data	Fast & complete data acquisition through website own underlying API used on the website
Fabric Notebook with Python and beautiful soup	**Method 2 Python:** Use **webpage source** to extract data using **beautiful soup** module	Fast data scraping method to select element available on the website

Table 5 - Fabric Notebook method and feature

Below are few of the considerations taken into consider on which services is better suited to be used on certain use case in data acquisition phase:

a) Python with Fabric Notebook:

- Suitable for calling APIs that return complete webpage data in JSON format, without the need for browser interaction.
- Ideal for scenarios where the page source can be directly retrieved through a web request.

b) Power Automate:

- Suitable in cases where launching a browser is required, such as for websites with content dynamically loaded via JavaScript.
- Recommended for scenarios that involve user interaction, including actions like clicking, scrolling, or entering input.

Below is the further breakdown comparison of using the two services:

	Power Automate	Python with Fabric Notebook
Time to gather data.	Power Automate on average able to scrape **10.8 products/minute.**	Python scraping on average able to scrape **1200 products/minute.**
Proxy setup	Configuring proxy requires modification of config file in VM C:// path.	Configuring proxy can be done through parameter in the code

VM used	An additional VM is needed to run the Power Automate Desktop flow. One website takes 8 hours to process, a single VM can handle a maximum of 3 websites, for daily crawling.	VM is not needed as data acquiring process script is run using underlying fabric notebook engine
Parallel run execution	Would need multiple accounts to log in to the Windows server to enable parallelism.	The Python fabric notebook approach allows parallelism easily using concurrent notebook libraries.
Scheduling	Support scheduled runs	Support Scheduled runs

Table 6 - Power automate and Fabric notebook comparison

Figure below further elaborates the flow on decision of what service and method to be used for different use case and consideration.

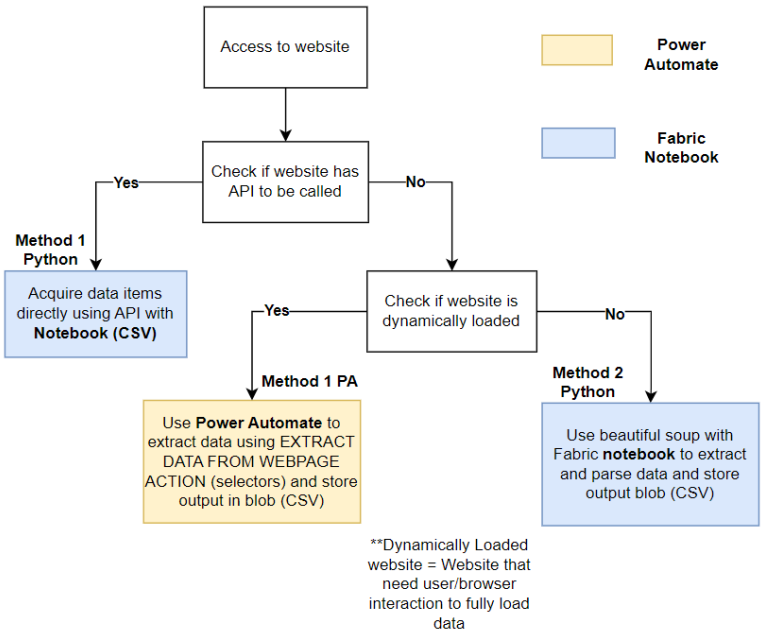


Figure 3 - Data acquisition method flow

Decision flow	Description	Decision
Check if websites have API to be called	First decision flow is when website is accessed and checked whether the website have an underlying API to be called. It is done through searching the websites network header tab of developer's tool	If the website has underlying API to be called it will proceed to **acquire data items directly using API through Python in Fabric notebook environment. (Method 1 Python)**

		If the website does not have underlying API to be called it will move to the next decision flow.
Check if websites is dynamically loaded	Next decision flow happens if the website doesn't have an underlying API to be called. It first check if the websites is dynamically loaded which basically means the websites requires browser interaction on its page to fully load data before it can be acquired. It is done through checking the website structure whether it requires scrolling action to fully load website, requires prompt/login to access or if the website is blocked by javascript.	If the website is not dynamically loaded it will utilize method of data acquisition through **parsing the website element using Python beautiful soup. (Method 2 Python)** If it is dynamically loaded it will just use **Power Automate to extract data using "Extract Data From Webpage Action" (Method 1 PA)**

Table 7 - Data acquisition method flow description

Dynamic and non-dynamic website identification

During the scraping solution design and implementation phase, it is important to **classify websites** as either **dynamic** or **non-dynamic (static)**, as this directly influences the scraping approach, tools used, and potential challenges.

1. Non-Dynamic (Static) Websites

Non-dynamic websites are pages where the **content is rendered on the server-side** and sent directly to the client (browser). This means that:

- **All necessary data is present in the initial HTML source code**.
- The data can usually be accessed by standard parsing tools like BeautifulSoup or by directly calling **public or hidden APIs**.
- These websites **do not require user interaction** (like clicking or scrolling) to reveal their data.
- These pages are typically **lightweight, faster**, and more straightforward to scrape.

Indicators of a non-dynamic website:

- Viewing the page source (Right-click > View Page Source) reveals the data you are looking for.
- No JavaScript rendering is required to access the content.
- A network tab (browser dev tools) shows direct API responses in XHR calls, which can be reused in the scraper.

Example: A government website listing tenders in table format or a product list where data loads immediately.

2. Dynamic Websites

Dynamic websites generate content **on the client-side using JavaScript**. The data is not present in the initial HTML but is rendered after scripts are executed in the browser. These sites may also require user interactions such as:

- Clicking buttons or tabs
- Scrolling to load more content (infinite scroll)
- Completing a login process
- Waiting for AJAX or XHR calls to complete

Scraping dynamic websites typically requires **headless browsers or automation tools** like Selenium or Playwright that can simulate human interaction and wait for elements to load.

Indicators of a dynamic website:

- The data is **not visible** in the page source but appears in the browser after page load.
- Dev Tools (Network tab) show AJAX/XHR requests fetching the data separately after page load.
- Website heavily relies on JavaScript frameworks like React, Angular, or Vue.

Example: An e-commerce site like Agrobazaar.com.my, where prices and products load only after scrolling or JavaScript execution.

Below are list examples of dynamic and non-dynamic website that is tested:

Dynamic/Non-dynamic	Website name
Dynamic (Power Automate/ Python with Selenium and Fabric notebook)	malaysiaairlines.com
	Agrobazaar.com
	Airasia.com
	Redbus.com
	Busonlineticket.com
	Ktmb.com.my
	Airpaz.com
Non-Dynamic	bigpharmacy.com.my
	hlkonline.my
	aeon2go.com
	Lotuss.com.my
	Senheng.com
	Durianproperty.com.my
	Landplus.com
	Iproperty.com.my
	Propertyguru.com
	Speedhome.com
	Booking.com

	Agoda.com

Table 8 - List of website with dynamic or non-dynamic structure

Further elaboration of each method:

3.1.1.1. Method 1 Python: Acquire Data Items Directly Using API request with Python Fabric Notebook

3.1.1.1.1. Initial step for detecting API in websites

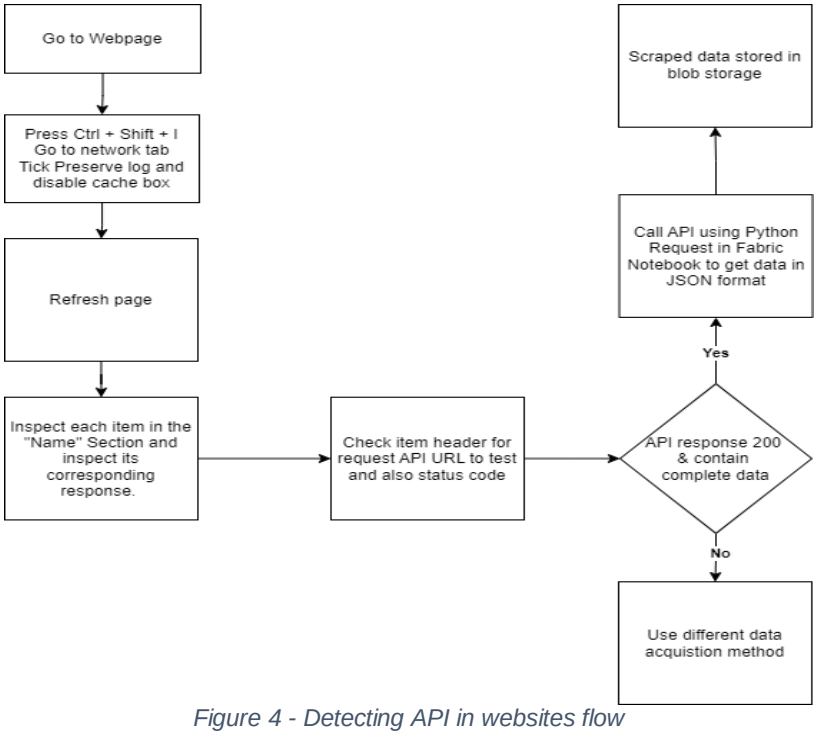


Figure 4 - Detecting API in websites flow

Before gathering data using API is done, it is required to check first if the underlying website have available API which contains the complete data displayed in the website, some website doesn't provide API to be called publicly and instead uses different method to display the data in the website portal
Below Further elaborates the step taken to detect API in website.

Flow	Description
Go to webpage	Website is accessed and analysed for its structure. Whether it is dynamic and require user interaction to load webpage.
Press Ctrl + Shift + I, go to network tab, tick preserve log and disable cache box.	Keyboard shortcut to access the developers console of the webpage and checking the Preserve log and Disable Cache to allow detailed analysis of webpage.

Refresh the page & inspect each item in “Name” section, inspect its corresponding response.	Refreshing the page allows for updated list item in the developer’s console “Name” list.
Check item header for request API URL to be tested and also check the status code	Navigate to header section to check for the underlying API used by the website in request URL, also checked for status code 200 meaning success and test the API either using Python or Postman to inspect the result if retrieved result is complete.
Use different data acquisition method.	If the API response shows other than 200 result code, or the API result shows incomplete/irrelevant data, different data acquisition method is used.
Call API using Fabric notebook request to get data in JSON format.	If the API is complete and callable with response code 200, call the API using Python request module through Fabric notebook.
Data processed and stored in blob storage.	The result of the API call is in JSON format, the next standardize step to take is to convert to CSV format and store into blob storage as unstructured data.

Table 9 - API Method flow

Below visual shows result for a working API structure of a website.

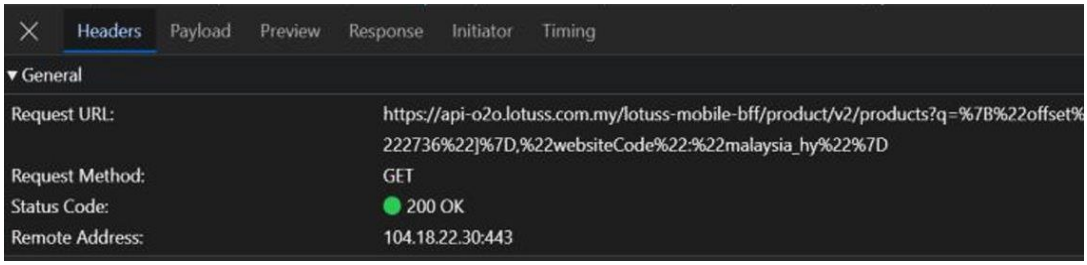


Figure 5 - Example of working API

Below visual shows result for a non-working API structure of a website, the structure shows API just calls the Html visual elements instead of the actual price data.

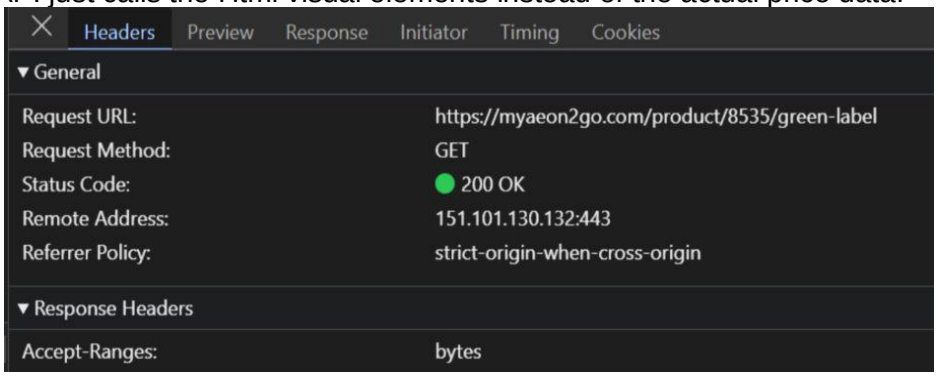


Figure 6 - Example of non-working API

3.1.1.1.2. Using API request to gather data with scheduler

Once API is acquired from website that includes a complete data, the API is used together with Python request code to call the API and obtain the complete data and stored it inside storage.

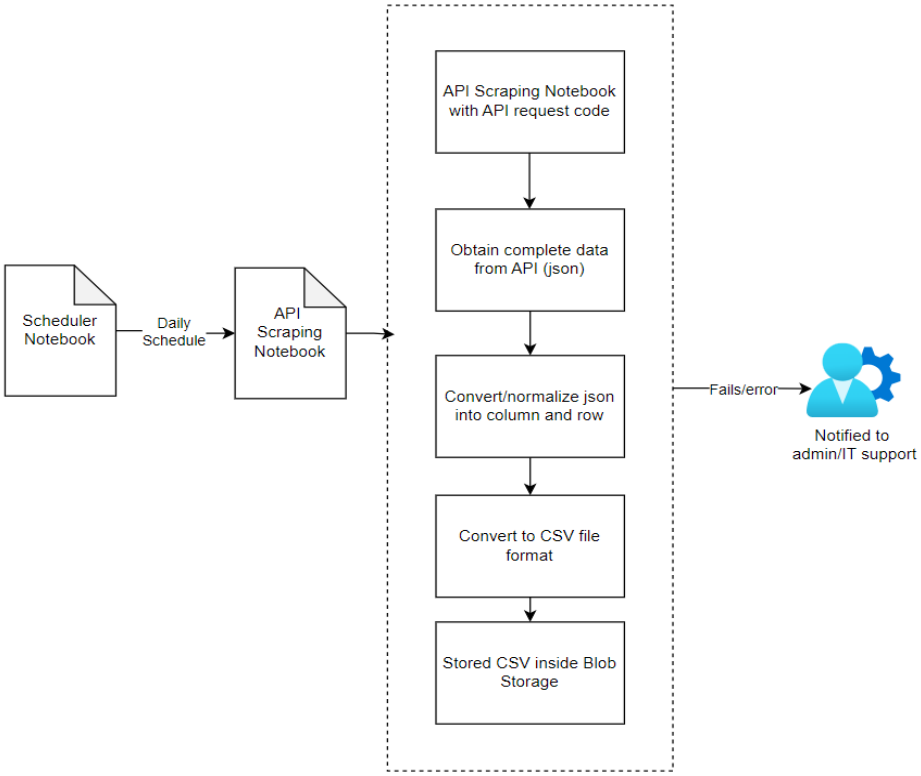


Figure 7 - Using API scraping code with scheduler

Notebook	Flow
Scheduler Notebook	Scheduler notebook contain schedules and frequency in which when the API scraping notebook containing request calling code is run at what interval. (e.g daily, monthly, hourly)
API scraping notebook	i. API scraping notebook with the working API will call the API obtained in previous step ii. Complete data will be obtained from the API iii. Data obtained from API will be converted into structured column and row format iv. Data will be converted into CSV file and will be stored inside a blob storage with proper “ddmmyy_websitename” format

Table 10 - Using API with scheduler and scraping code

3.1.1.2. Method 2 Python : Scraping Data with Python and Fabric Notebook

3.1.1.2.1. Initial step for Extracting website element for parsing

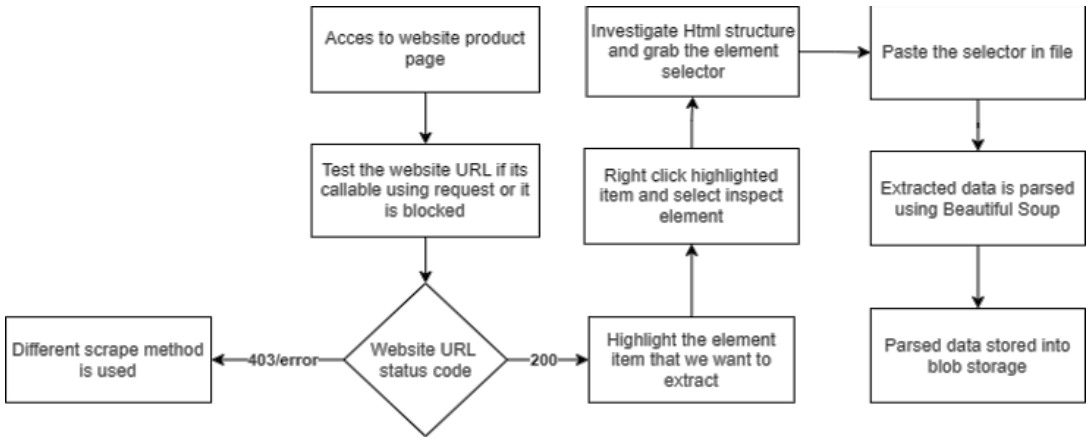


Figure 8 - Scraping with Beautiful Soup method

Flow	Description
Access website product page	Website product page is accessed.
Test website URL if its callable using request or if it is blocked.	Test the website URL request using tool such as Postman or python request
Check status code of website URL.	Beautiful soup is part of Python module to parse inspected elements from website to extract data to be processed.
Highlight element item we want to extract.	The element item that we want to extract such as title, price, description, discount, is highlighted.
Right click highlighted item and select inspect element.	Right click and select the inspect element in list of options
Investigate Html structure and grab the elements selector.	Select the element structure on the list of html structures
Paste the selector in file.	Paste the element in a file
Extracted data is parsed using Beautiful Soup.	Parsed the data that has been extracted using beautiful soup code in Fabric notebook.
Parsed data stored into blob storage.	The parsed data is then converted into .csv format and stored into blob storage.

Table 11 - Beautiful Soup Parsing Flow

3.1.1.2.1. Using Beautiful Soup to gather data with scheduler

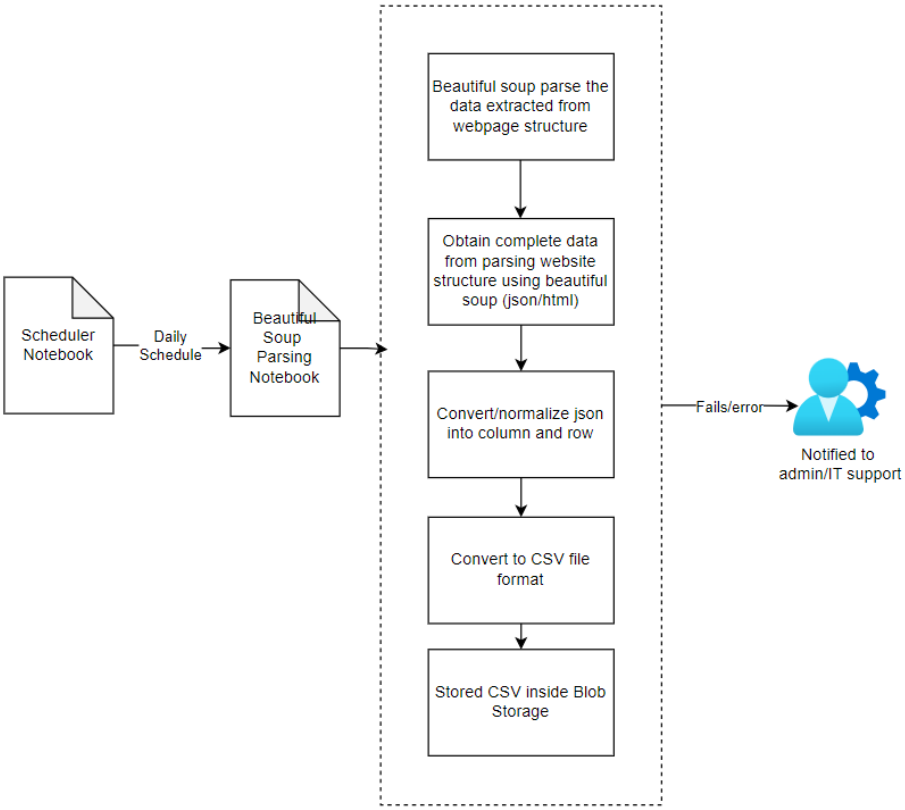


Figure 9 - Using Beautiful Soup scraping code with scheduler

Notebook	Flow
Scheduler Notebook	Scheduler notebook contain schedules and frequency in which when the API scraping notebook containing request calling code is run at what interval. (e.g Daily, monthly, hourly)
API scraping notebook	i. Beautiful Soup parsing code will parse website structure detected from previous step. ii. Complete data will be obtained from the parser code. iii. Data obtained from parser will be converted into structured column and row format iv. Data will be converted into CSV file and will be stored inside a blob storage with proper “ddmmyy_websitename” format

Table 12 - Using Beautiful Soup Parsing code notebook with scheduler

3.1.1.3. Method 1 PA: Power Automate Crawling Flow

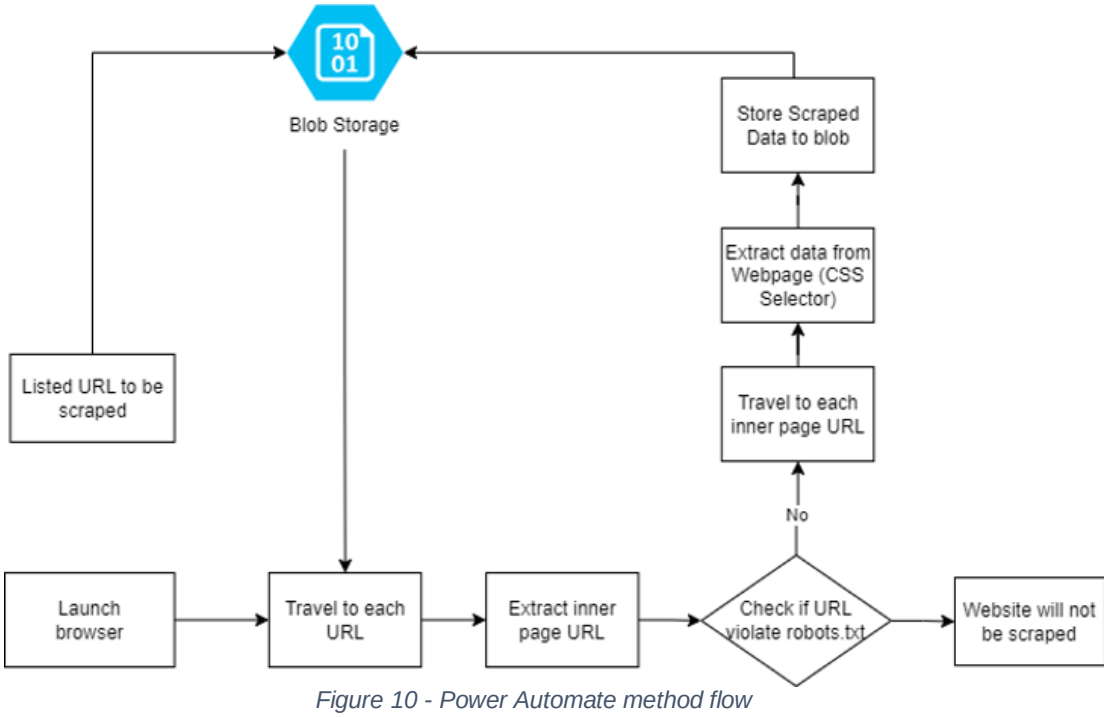


Figure 10 - Power Automate method flow

Flow	Description
Travel to each URL	The Power Automate crawler will explore to each website URL
Extract Inner page URL	It will extract inner page URL
Check if URL violate robots.txt	It will first check if it violates the robots.txt to scrape the website or if there is any other restriction
Travel to each inner page URL	Inner page URL to begin extraction
Extract data from webpage using CSS selector.	Extract data using CSS selector through selection done manually.
Store scraped data into blob storage	Data is first converted from raw crawled data into a CSV file format to be stored inside blob storage for backup.

Table 13 - Power Automate Crawling Flow

Below shows parameter set in power automate UI

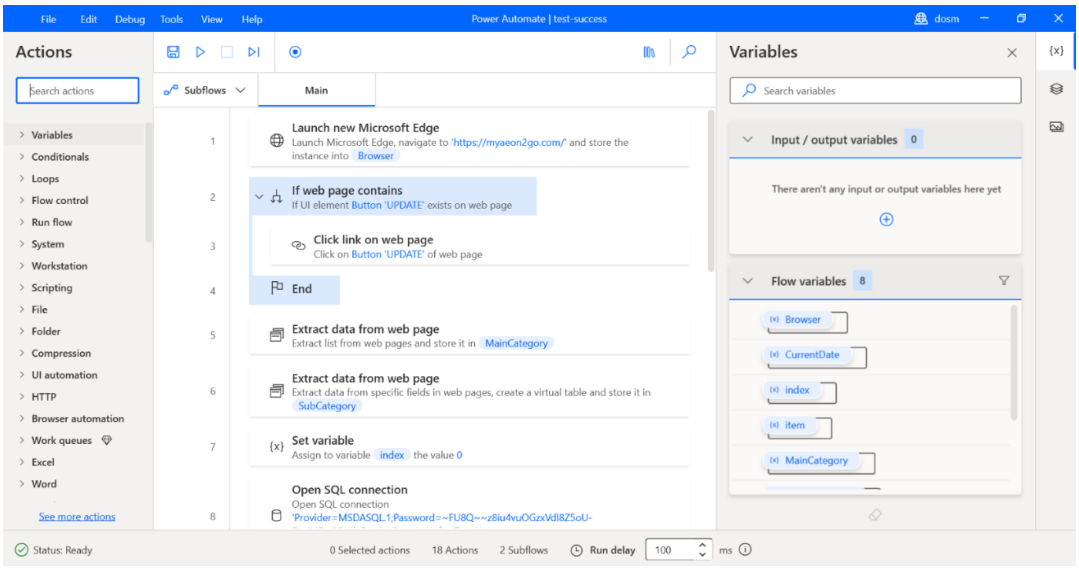


Figure 11 - Power Automate user interface visual

3.1.2. Technical Strategies for Captcha/Anti-Bot Handling

Websites with Captcha are approached with strict adherence to ethical web scraping practices and compliance with the website's rules.

Measures to minimize Captcha prompts include **implementing rate limiting** to avoid overloading the website and **utilizing API access where available**. As a last resort, dynamically changing IP addresses by using **Proxy** is suited, provided it aligns with ethical considerations and the website's terms of service.

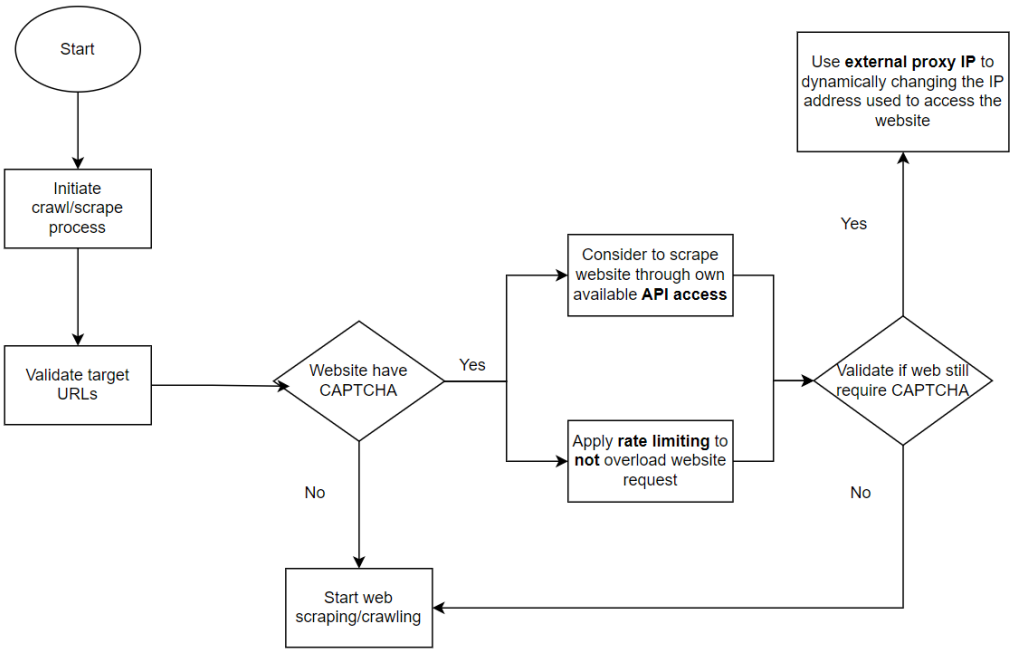


Figure 12 - Captcha handling flow visual

No.	Process Flow	Description
1.	Initiate Crawl Request	Initiated by the scheduler to start the data web crawling process.
2.	Validate Target URLs	The system checks the provided URLs to ensure they are reachable and comply with crawling rules (e.g., robots.txt). If website have a Captcha prompt when accessing the website, the process logs an error and notifies the administrator.
3.	Start Web Crawling	If the website doesn't have a Captcha the Web Crawler begins the process by accessing the homepage of the specified website and begin scraping process.
4.	Implement Rate Limiting if CAPTCHA encountered	Implement Rate limiting through timeout request in accessing website referring to the website Robots.txt timeout set, so that to avoid encountering Captcha, some website have different rate limit which need to be considered,
5.	Scrape data using website own available API	Considering prioritizing using available website API access by searching the website elements if scraping using API is available in the website.
6.	Change IP through external proxy	If selected website URL still requires Captcha to crawl/access the website , external Proxy IP is used which will change the IP address used to crawl the website. If it doesn't have Captcha it will just continue to crawl the website.

Table 14 - Captcha Handling flow description

3.1.3. Proxy setup

In the design of a web scraping solution, incorporating a **proxy** is essential for the following reasons:

1. **Bypassing Geo-Restrictions**

   Some websites restrict access based on IP location. Using a **Malaysian proxy** allows access to country-specific content that is otherwise blocked for foreign IPs.

2. **Avoiding IP Blocking & Rate Limits**

   Websites may block repeated requests from the same IP. Proxies help **rotate IP addresses**, reducing the risk of being detected and banned.

3. **Anonymity & Security**

   Scraping directly from a server exposes its IP. Using a proxy **masks the original IP**, protecting the infrastructure from tracking or restrictions.

4. **Load Distribution & Performance Optimization**

   Distributing requests across multiple proxies helps **prevent overload** on a single IP and improves scraping efficiency.

5. **Accessing Dynamic Content**

   Some sites serve different content based on the user's location. A proxy ensures that the scraper **sees the same data** as a real user in the targeted region.

This ensures a **reliable, scalable, and undetectable** scraping process while complying with access restrictions.

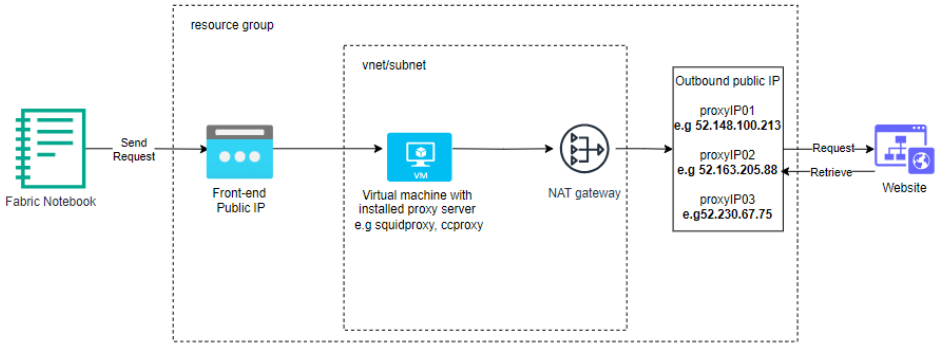


Figure 13 - Example proxy setup for changing outbound iP

Proxy setup in Azure allow masking of internal IP coming from DOSM with an external IP which will outbound to the website, this if in some case the website would block the IP, it would not block DOSM IP and would only block the outbound IP coming from NAT Gateway hosted in Azure.

3.2. Processing and storage

3.2.1. Deduplication, Cleansing, Classification Objectives

Process	Services Involve	Key Objectives
Data Deduplication	Fabric Notebook	• Reduce redundancy from same sources and day. • Improve storage efficiency. Ensure one accurate version of each record. (URL)
Data cleansing	Fabric Notebook	• Remove errors (e.g., typos, incorrect entries) through Python word correction code (fuzzyWuzzy, textDistance, symSpell) • Standardize formats (e.g., dates as "YYYY-MM-DD", weight of item as “g”, “kg”). • Fill in missing data and tag the imputed price (e.g., using imputation or estimation). • Eliminate irrelevant or outdated data through manual identification or filtering with main master dictionary table (e.g irrelevant data not needed is not cleansed, outdated data previously obtained is filtered out) • Ensure consistency (e.g., harmonizing flight or destination code like "KUL" and "CHN").
Data Classification	Fabric Notebook with Fabric Machine learning/ experiments	• Classifies data into MCOICOP at 8-Digit to allow for analysis. • Classifies data into ICP to allow for analysis.

Table 15 - Cleansing, Deduplication and classification objectives

3.2.2. Naming Convention for Azure & Fabric Services

Category	Type	Naming Convention	Example
Azure	Resource group	region-project-clientname - module-RG	SEA-STATSBDA2-DOSM-PI-RG
	Service name	region-project-clientName-module-serviceName	SEA- STATSBDA2 - DOSM-PI-BLOB
Azure Blob Storage	Container Name	region-project-clientName-module-CON	SEA- STATSBDA2 - DOSM-PI-CON
	Data File Path	project/data/year/month/date/ project/logs/year/month/date/	statsbda2/data/<yyyy>/<mm>/<dd>/ statsbda2/logs/<yyyy>/<mm>/<dd>/
Azure DB	DB Name	[Region]_[Project]_[ClientName]_[Module]_DB	SEA_STATSBDA2_DOSM_PI_DB
	DB Table Name	[Schema].[TableName] *Schema = ModuleName	PI.ECOMMERCE
Fabric Lakehouse	Lakehouse Name	[REGION]_[Project]_[ClientName]_[Module]_LH	SEA_STATSBDA2_DOSM_PI_LH
	Lakehouse Table Name	[Schema].[TableName] *Schema = ModuleName	PI.ECOMMERCE
Fabric Warehouse	Warehouse Name	[REGION]_[Project]_[ClientName]_[Module]_WH	SEA_STATSBDA2_DOSM_PI_WH
	Warehouse Table Name	[Schema].[TableName] *Schema = ModuleName	PI.ECOMMERCE
Fabric Pipeline	Pipeline Name	[Team]_[Module]_[ProcessType]_[TableName]	ETL_PI_CRAWL_PI_ECOMMERCE
	Dataflow	PipelineName.Dataflow	ETL_PI_CRAWL_PI_ECOMMERCE_DF
	Copy Job	Copy [DataName] From [Source]	Copy PI_ECOMMERCE from SEA_STATSBDA2_DOSM_PI_DB
Fabric ML model	Model Name	[Module]_[UseCase]_[Function]_[ModelType]	PI_ML_MCOICOP_CLASSIFICATION_SVM
Fabric Notebook	Notebook Name	[Module]_[UseCase]_[Function]	PI_CRAWLER_AEON_SCRAPER PI_CLEANING_ECOMMERCE_PREPROCESS

Table 16 - Naming Convention for PI Service

3.2.3. Storage Design Architecture (Fabric Lakehouse/ Data Warehouse)

Data Storage Architecture

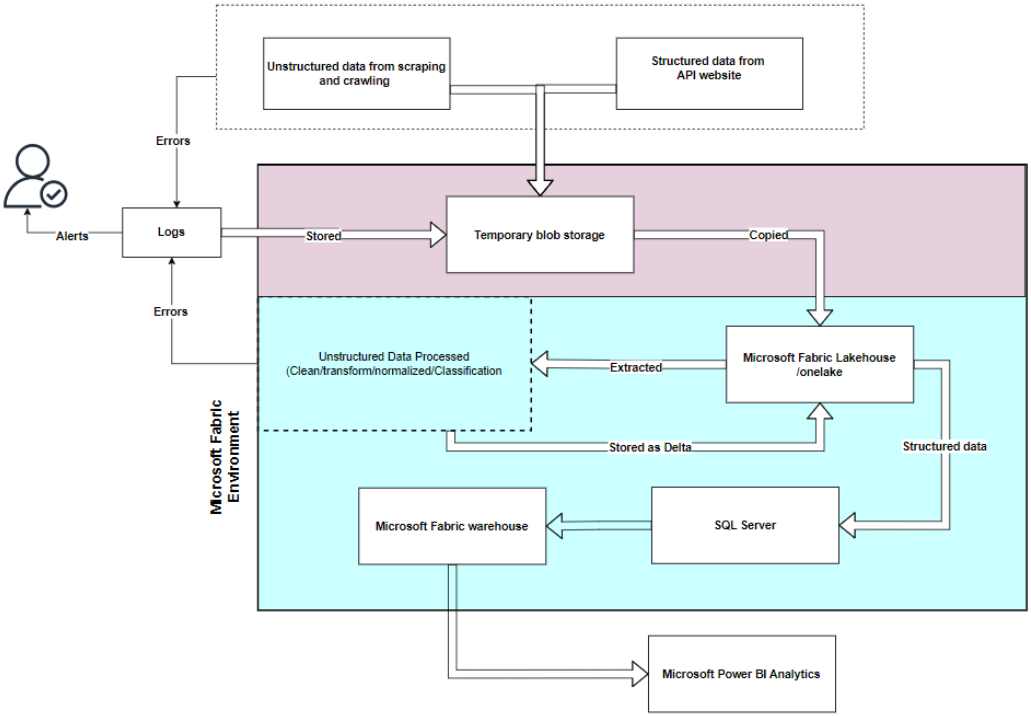


Figure 14 - Data storage flow visual

There **Three** storage category solution used in the design architecture of Price Intelligence in throughout the overall ETL process:

1. Temporary storage / backup storage
   - Azure blob storage
2. Structured storage
   - Microsoft fabric Lakehouse
3. Aggregated storage
   - SQL Server
   - Microsoft fabric data warehouse

Data Stored	Storage Type	Process
Crawled Data (.csv) & Log File (.log)	Azure Blob storage	Raw crawled data is stored here temporarily in CSV format with column and row value through "ddmmyy_websitename" naming convention
Delta table	Azure Fabric lakehouse/onelake.	Data extracted from blob storage CSV will be stored as delta table format in Fabric Lakehouse with defined schema. Data in Fabric lakehouse is used for processing, cleaning , standardization classification and enrichment process before

		being staged as aggregated data for useful analytics.
SQL table and Delta table	▪ SQL Server ▪ Microsoft Fabric Data Warehouse	After data is processed and classified, data is transported to SQL server where it will be used in CRUD operation where it is optimized for editing and update in PI internal portal It is then imported into Microsoft Fabric data warehouse for analytics and reporting using Power BI

Table 17 - Data storage flow description

Below shows the hierarchy convention for data stored inside blob storage, it is separated into Crawled data and Log data for audit logs.

Data	Hierarchy
Crawled Data	crawled_data/ yyyymmdd/ website_name/ **yyyymmdd_website_name.csv**
Audit Log Data:	crawled_data/ yyyymmdd/ website_name/ **yyyymmdd_website_name.log**

Table 18 - storage data naming convention hierarchy

When performing web scraping, the retrieved data is typically in **unstructured HTML or JSON format**. To ensure the data can be analyzed, stored, or queried effectively, it must first be **converted into a structured format**.

The following explains how **HTML content is processed into a structured CSV file**, below further describe the process:

No.	Step	Description
1.	**Retrieve Raw HTML**	Web scraping tools (e.g., Python-based scripts) access and download the webpage content, which includes all elements: text, tags, scripts, styles, etc.

2.	**Extract Relevant Data**	Only specific elements are targeted for extraction (e.g., product names, prices, descriptions). These are identified by their HTML structure (tags, classes, etc.).
3.	**Remove Unwanted HTML Markup**	Tags such as , , <script>, and other formatting elements are removed. Only the clean text content is retained.
4.	**Normalize the Data**	The extracted data is cleaned, aligned by rows (e.g., one row per product), and formatted into a consistent structure for tabulation.
5.	**Convert to CSV Format**	The normalized data is saved into a **Comma-Separated Values (CSV)** file, creating a table-like format that can be read by databases, spreadsheets, or analytics tools.
6.	**Store in Blob Storage**	The resulting CSV file is uploaded to **Azure Blob Storage** for safe backup and centralized access by downstream processes.

Table 19 - Raw HTML to structure data conversion

Purpose of This Process

This transformation process is essential because **HTML is not inherently structured for analysis**. By extracting only the required data and converting it into a CSV format, the information becomes:

- Easier to **search and retrieved.**
- Compatible with **data visualization** and **analytics tools**
- Ready for **data cleansing**, **classification**, and **archiving**

3.3. Dashboard and monitoring

3.3.1. Price Intelligence Internal Portal dashboard components

Main portal dashboard for price intelligence consist of 3 component pages where internal user can navigate and make changes to PI data item, below further describe the page component of Price Intelligence internal portal dashboard.

a) Internal Reporting Page
b) Classification Review Page
c) Administration page

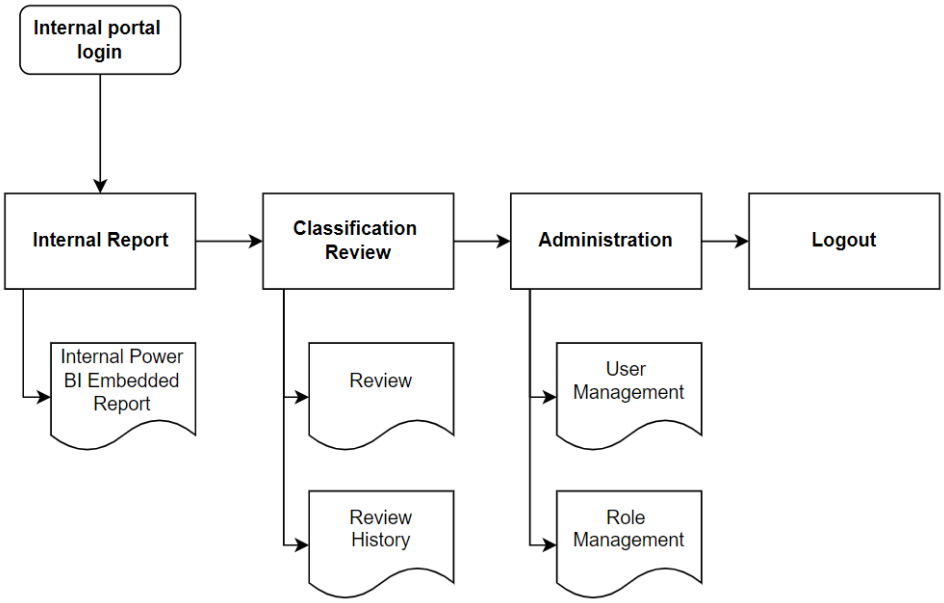


Figure 15 - Internal Portal page navigation

a) Internal Reporting Page

Price Intelligence internal reporting page staged the internal power BI report where **DOSM Data Analyst** publish reporting dashboard analytics and visualization for internal user to analyze price intelligence data.

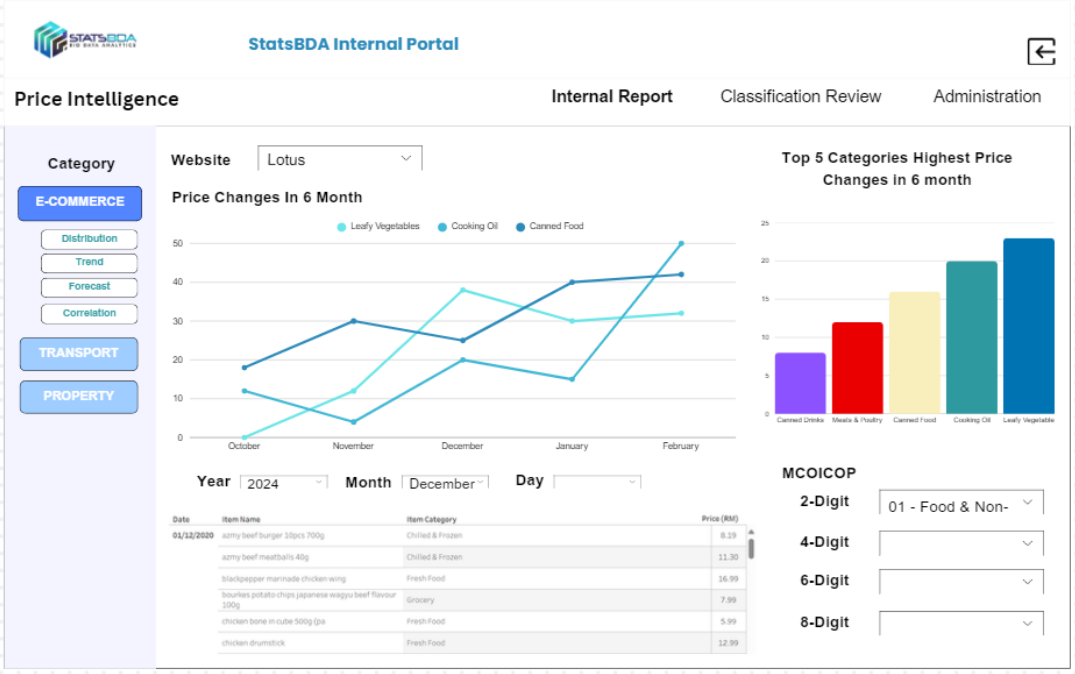


Figure 16 - Internal Power BI Dashboard report

Feature for PI **Internal Report page:**

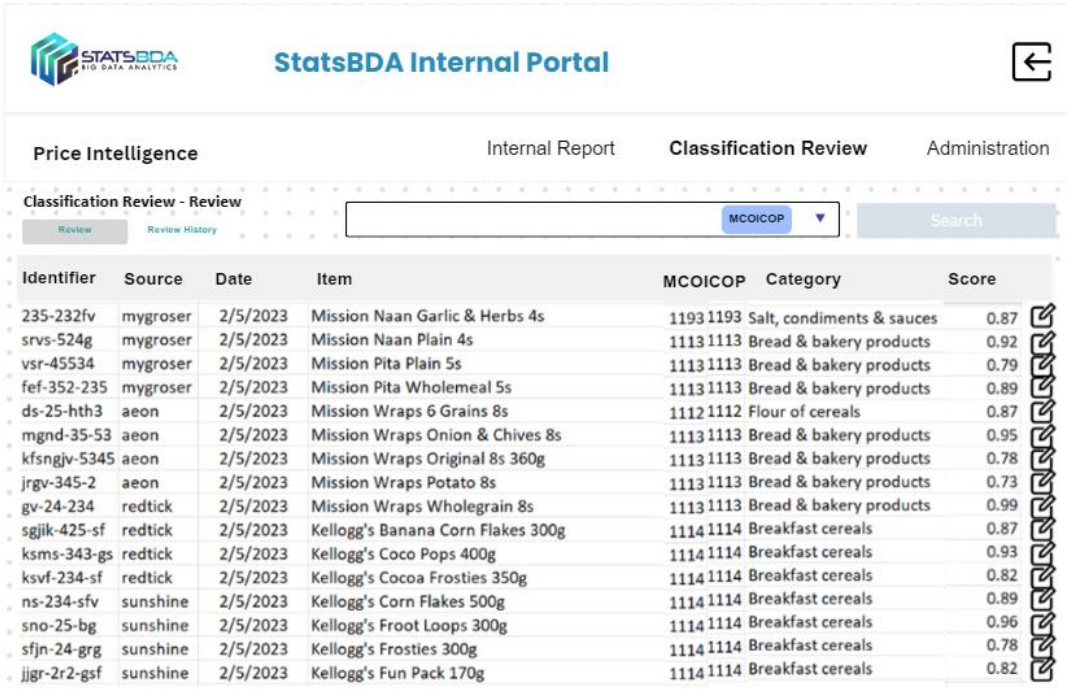
Internal Report	▪ Embedded Power BI report with visualization and analytics reporting publish by Data analyst for internal and external view ▪ Download data from charts/tables.

Table 20 - Proposed Power BI dashboard elements

b) Classification review page

Allow for internal user to review classification done by machine learning, and allow for correction of the classification which will be reflected to the master dictionary table and training dictionary table.

i. Review Tab



Identifier	Source	Date	Item	MCOICOP		Category	Score
235-232fv	mygroser	2/5/2023	Mission Naan Garlic & Herbs 4s	1193	1193	Salt, condiments & sauces	0.87
srvs-524g	mygroser	2/5/2023	Mission Naan Plain 4s	1113	1113	Bread & bakery products	0.92
vsr-45534	mygroser	2/5/2023	Mission Pita Plain 5s	1113	1113	Bread & bakery products	0.79
fef-352-235	mygroser	2/5/2023	Mission Pita Wholemeal 5s	1113	1113	Bread & bakery products	0.89
ds-25-hth3	aeon	2/5/2023	Mission Wraps 6 Grains 8s	1112	1112	Flour of cereals	0.87
mgnd-35-53	aeon	2/5/2023	Mission Wraps Onion & Chives 8s	1113	1113	Bread & bakery products	0.95
kfsngjv-5345	aeon	2/5/2023	Mission Wraps Original 8s 360g	1113	1113	Bread & bakery products	0.78
jrgv-345-2	aeon	2/5/2023	Mission Wraps Potato 8s	1113	1113	Bread & bakery products	0.73
gv-24-234	redtick	2/5/2023	Mission Wraps Wholegrain 8s	1113	1113	Bread & bakery products	0.99
sgjik-425-sf	redtick	2/5/2023	Kellogg's Banana Corn Flakes 300g	1114	1114	Breakfast cereals	0.87
ksms-343-gs	redtick	2/5/2023	Kellogg's Coco Pops 400g	1114	1114	Breakfast cereals	0.93
ksvf-234-sf	redtick	2/5/2023	Kellogg's Cocoa Frosties 350g	1114	1114	Breakfast cereals	0.82
ns-234-sfv	sunshine	2/5/2023	Kellogg's Corn Flakes 500g	1114	1114	Breakfast cereals	0.89
sno-25-bg	sunshine	2/5/2023	Kellogg's Froot Loops 300g	1114	1114	Breakfast cereals	0.96
sfjn-24-grg	sunshine	2/5/2023	Kellogg's Frosties 300g	1114	1114	Breakfast cereals	0.78
jjgr-2r2-gsf	sunshine	2/5/2023	Kellogg's Fun Pack 170g	1114	1114	Breakfast cereals	0.82

Figure 17 - Classification Review in internal portal for QA/QC

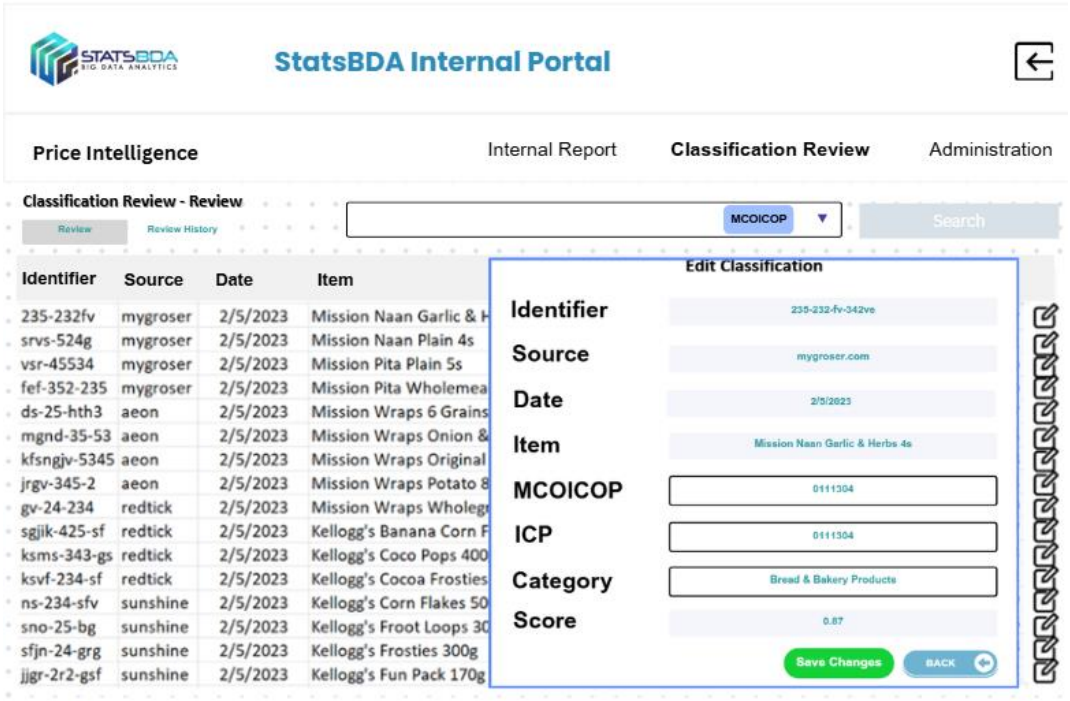


Figure 18 – Edit classification in review

Feature for PI **classification review – Review tab:**

Review Feature	▪ Review & Edit classification code (Mcoicop, icp, etc.) ▪ Search for job listing or code ▪ Filter by classification category (mcoicop, icp, etc.)

Table 21 - Classification Review tab feature

ii. Review History tab

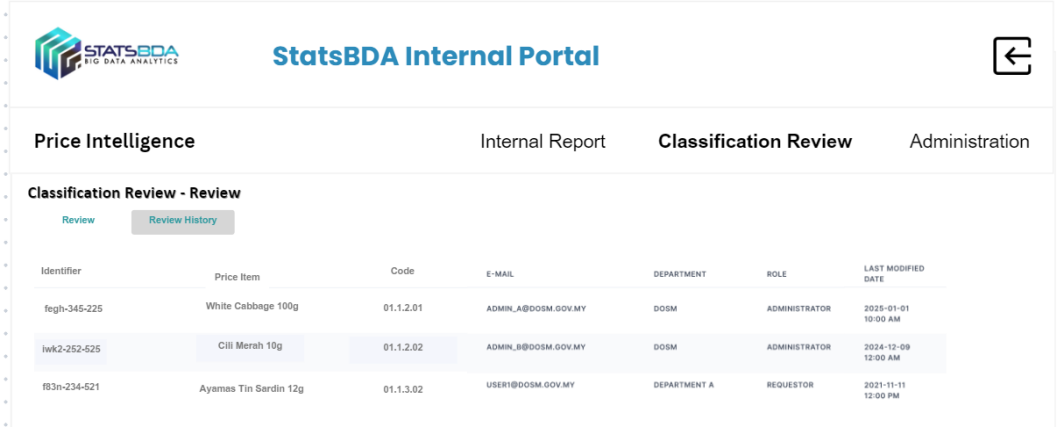


Figure 19 - Classification review history tab page

Review History Feature	▪ View history of classification corrections or changes done by each user role ▪ Show what changes have been made by user (eg: previous code vs latest code, verified code)

Table 22 - Classification review history tab page feature

c) Administration page

General page for user administration and access control to certain part of the internal portal.

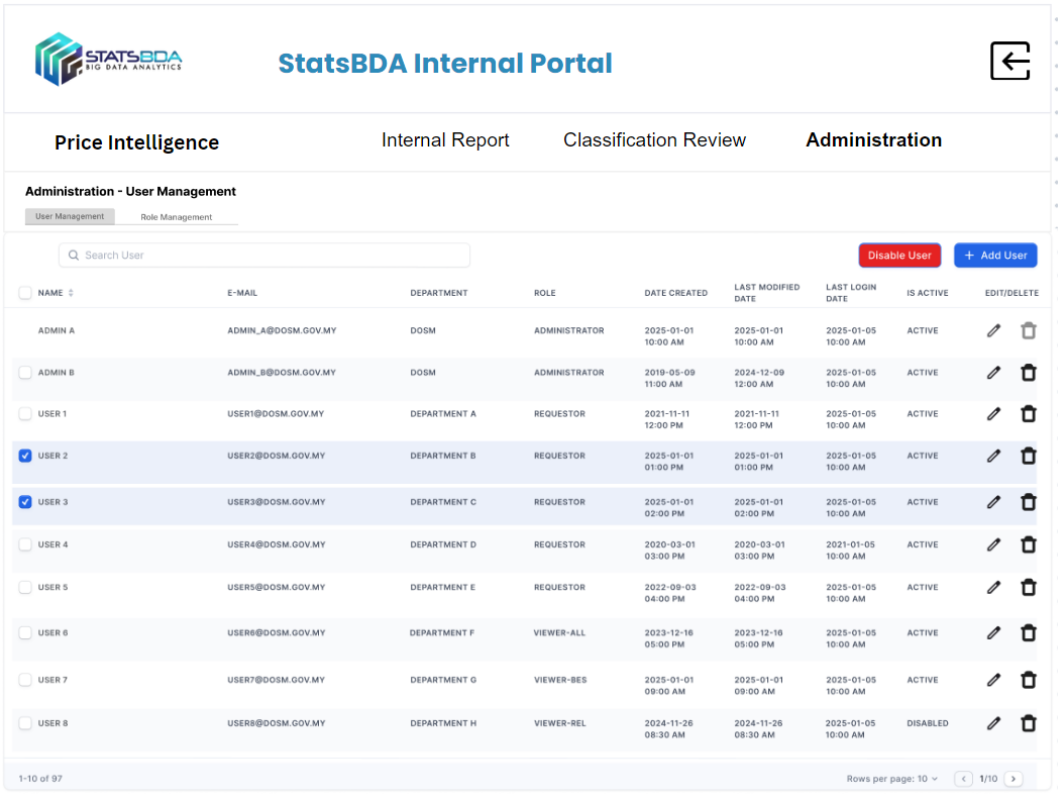


Figure 20 - User management of internal administration page

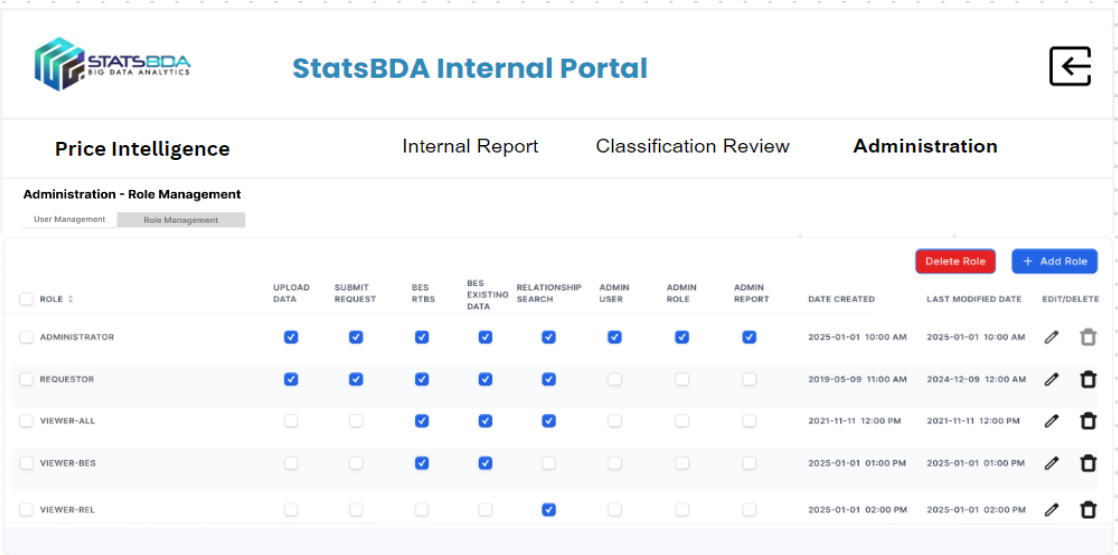


Figure 21 - Role management of internal administration page

3.3.2. Real-time Monitoring and Alerts Planning

Monitoring using azure monitor set up on intervals for each process in ETL process in ingestion to monitor status of ingestion whether it is successful or not successful in

ingestion using crawlers monitor the success of data manipulation, transform or deduplication.

Below further elaborate on the required component for monitoring and alerts setup

Alert Feature	Description	Implementation
Threshold Definition	Define thresholds for alerting based on price fluctuation percentages, missing data limits, or category-specific triggers for each item.	Threshold for price fluctuation or anomaly changes in price can be done through setting the threshold in Z-Score mechanism notebook.
Alert Types	Configure alerts for different scenarios, such as price spikes, missing data, or failure in data ingestion, with custom severity levels.	Types of alerts for different scenario will be configured inside the Fabric notebook code logic for price spike it will be handled using z-score mechanism. For any missing data or failure in data ingestion it will be handled by Fabric monitoring and power automate desktop monitoring
Monitor Job Status	Use Power Automate to monitor the status of each crawling job in real-time, tracking completion, failure rates, and data quality metrics.	Monitoring Job Status is done using the Fabric monitoring solution and Power Automate desktop flow monitoring which will monitor every job fail or success.
Setup Alerts	Set up alerts for issues such as inaccessible sites, excessive missing data, or unresponsive web pages.	Every monitored event such as inaccessible sites, excessive missing data or unresponsive web page will be handled by Fabric monitoring alert or Power Automate desktop monitoring, the subsequent logic will be implemented inside scheduled notebook code.
Trigger Notification	Trigger notifications to relevant teams or system administrators for issues that require attention, such as changes in website structure or unexpected blocks.	Every issue, error or notification can trigger notification message to user through Fabric flow or power automate flow which will trigger a sending of email report regarding the event

Table 23 - Monitoring and alerts component description

Monitoring service	Subject monitored	Alerts	Output
Microsoft power automate Desktop Flow monitoring	▪ Execute of Every Steps of Desktop Flow (durations, success status, log message) ▪ Desktop flow completion status, Error Rate, Flow run details, Error and so on.	Alerts are sent to user email using power automate flow	Logging stored in blob storage
Microsoft fabric monitoring	▪ Successful/failed data pipeline execution ▪ Execution time	Alerts are sent to user email using fabric pipeline connector.	Logging stored in blob storage
Notebook code with z-score analysis mechanism	▪ Price spike, missing price item,	Alerts are sent to user email using fabric flow	Anomaly reporting is visualize in Power BI dashboard

Table 24 - Monitoring and alerts decription

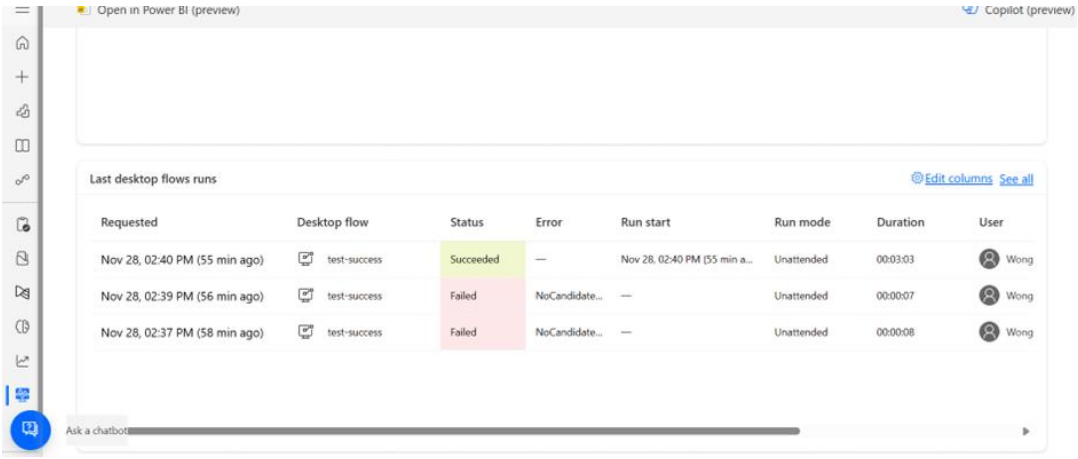


Figure 22 - Power Automate Monitor Succeeded and failed runs

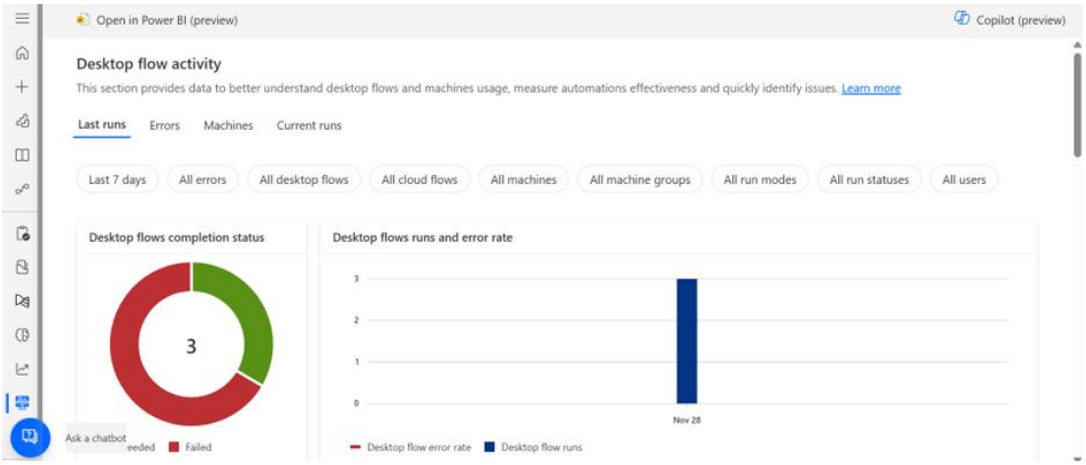


Figure 23 - Power Automate Monitor Metrics of flow runs

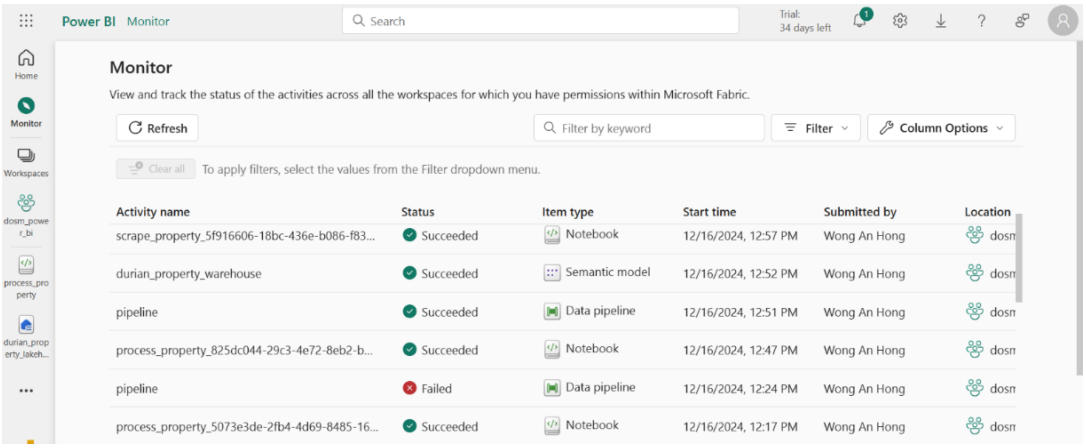


Figure 24 - Fabric Monitor succeed and fail runs

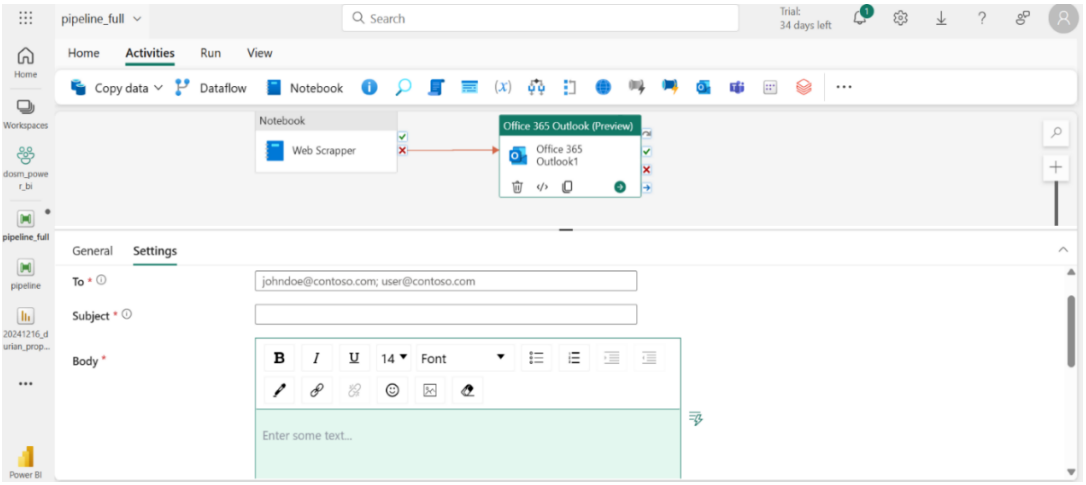


Figure 25 - Fabric Flow to send email of fail runs

4. System Component Design

Overall Price Intelligence (PI) component design can be further described in the following diagram:

No color differentiation for all websites

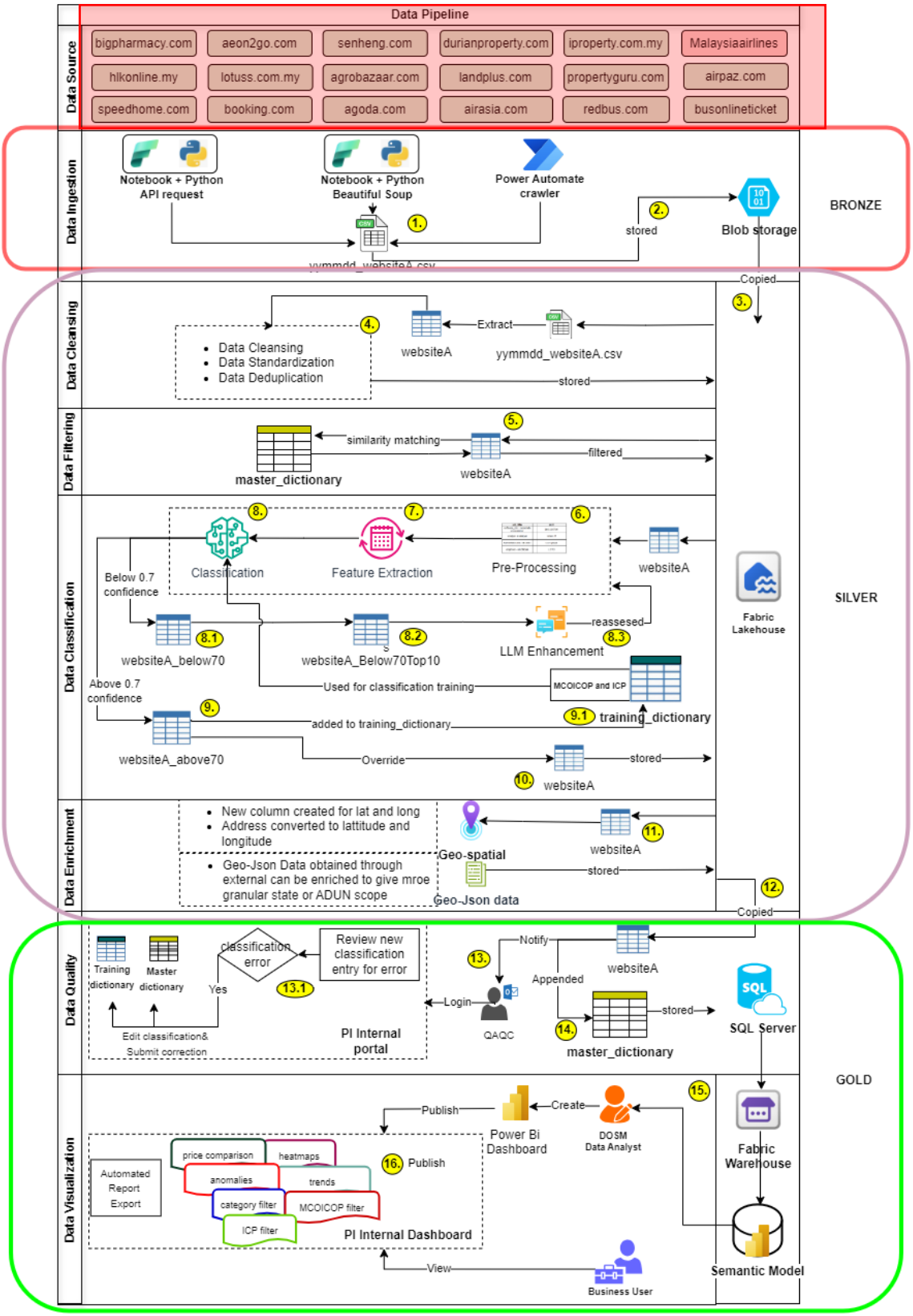


Figure 26 - Overall system component design and flow

4.1. Web Crawling and Scraping Component (Process 1 and 2)

Detailed web crawling and scraping component can be explained further below:

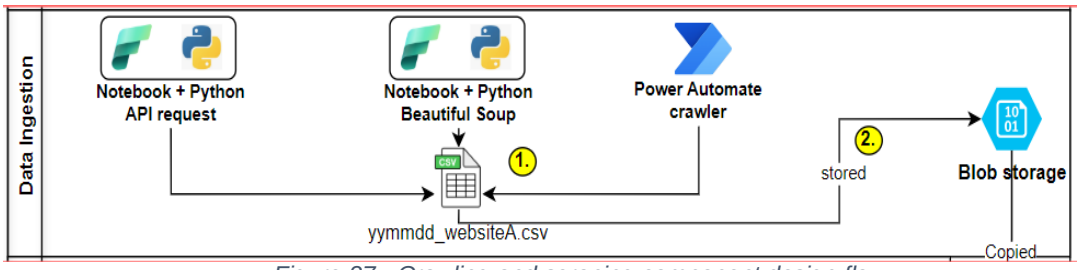


Figure 27 - Crawling and scraping component design flow

Flow	Description
Process 1	Data is gather using 3 methods as stated previously in: (3.1 Data Source Crawling) Initial format can be in Json and HTML
Process 2	Standardized and stored in blob as CSV file with name hierarchy "yymmdd_website_name".

Table 25 - Crawling and scraping component process description

4.1.1. Crawling Rules and Sites Definition

Below are some general ethical considerations when scraping a website URL:

- Crawler only collects and processes data that are made publicly available on millions of webpages in different countries.
- Crawlers collect minimal personal identifiable information or other personal data.
- Crawler observes the robot exclusion protocol. When a robots.txt file on a website requests web crawler to ignore files or directories, we will comply with the requests.
- We do not access data that is protected by a log-in, security code, other technical measures.
- The company cannot challenge Captcha as the company's crawler is not capable of solving the Captcha challenge.
- The company executes a 'right-to-be-forgotten' policy. Websites can be taken out of the database, and PII fields can be disabled upon request.

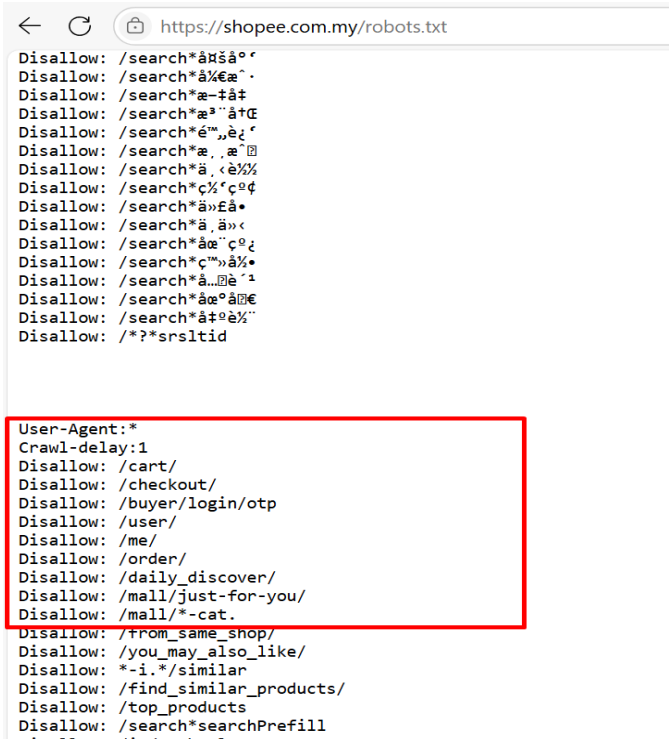


Figure 28 - Robots.txt example restriction

Append /robots.txt to the end of the website's URL (e.g., https://shopee.com.my/robots.txt).

- Review the file for message indicating that crawling is restricted.
- Find the User-Agent: * section to identify directories or files that are blocked for all crawlers.
- Check for Crawl-delay values, which specify how long a crawler should wait between requests.

4.1.2. Data Extraction Fields Specification

Below are the fields extracted when data scraping and crawling is done for each category, subject to change based on new dataset website:

Category	Data extraction fields
E-commerce	product_name,product_title,product_price_ori,product_price_disc,product_Transportation
Property	property_name,property_description,property_price,property_type,property_location,property_langtitude,property_longtitude.
Transport	Operator_name, price, departure_location,destination_location, depature_time, arrival_time.

Table 26 - Scraped Data extraction fields

4.2. Data processing component

4.2.1. Detailed Data transformation, and classification workflows

4.2.1.1. Data Cleansing, deduplication & standardization (Process 3 and 4)

Referring to objectives stated previously in (3.2.1 Deduplication, cleansing and classification objectives), this section further describe the process it takes to deduplicate, cleansed and standardize data.

Further elaboration of the process can be described below:

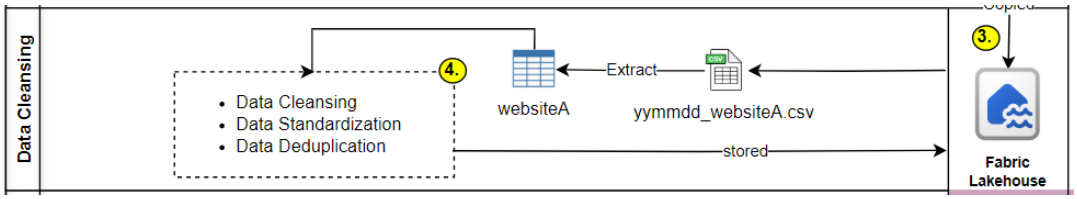


Figure 29 - Data cleansing and transformation component flow

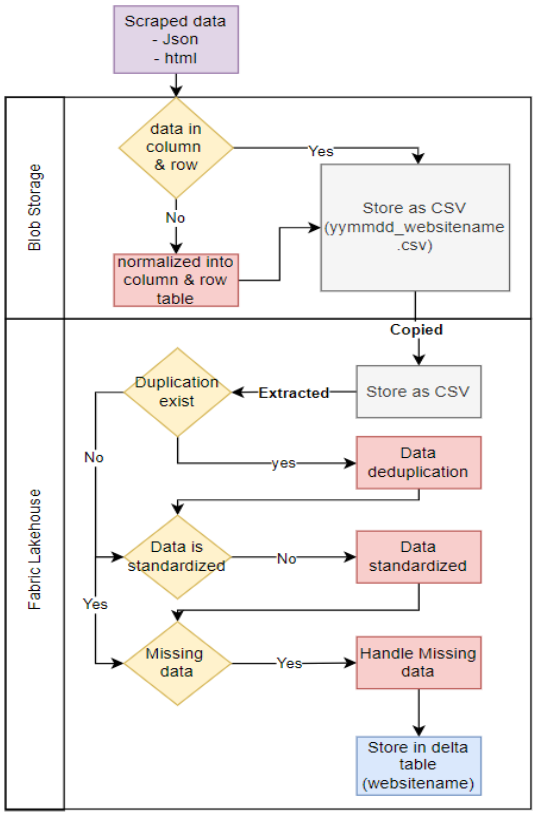


Figure 30 - Cleansing, Deduplication, Processing flow

Process	Description
Data extracted stored in column and row format	Data originating from crawler or scraper is obtained in either json or html format. Data obtained is first converted into column and row through code which is then stored as CSV file with yymmdd format name in the blob storage (e.g yymmdd_websitename.csv)
Extraction of csv data into delta table	Data from crawler in csv file with column and row value is then extracted into delta table for cleaning and standardization process using python cleaning notebook A Delta table in Fabric Notebook is a high-performance, ACID-compliant table format built on top of Apache Parquet. It is designed for reliable and scalable data storage and is used as stored format by Fabric Lakehouse and warehouse.
Data deduplication	Extracted data in delta table column-row format is first check if there is duplicated data that existed in the row this to ensure only single unique version of data available for processing.
Data Standardization	Extracted data is checked whether the unit used is standardized. Units are converted and standardized for the dataset dictionary using Python code run inside notebook (e.g changing g to kg)
Missing data handling	Data extracted is checked if there is missing data through python code that checked each row if there is null value, the null value is either replaced with an integer or removed entirely for the column, this is to avoid error in data processing in later stage
Stored in structured delta table.	Data deduplicated, standardized and handled for its missing value is stored inside a final structured delta table which is used further in data classification, enrichment and analytics.

Table 27 - Cleansing, Deduplication and process flow description

4.2.1.2. **Data Filtration** (Process 5)

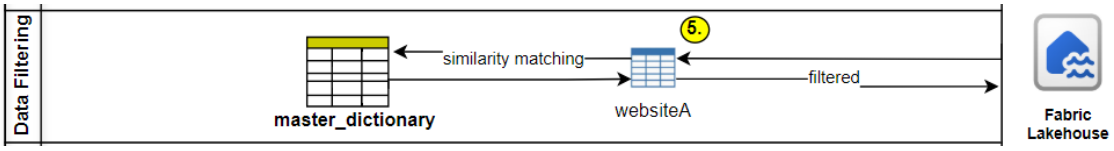


Figure 31 - Data Filtration component flow

Flow	Description
Process 5	Initially data item is filtered to check whether it is the same data being classified previously. The newly scraped website listing is compared with the master dictionary table, which is the final table database where previously classified item is staged. The newly scraped dataset table item namely websiteA filtered based on the unique classifier (url identifier, price item url) If the item previously classified and processed in the master dictionary table, it won't be classified again and not go through the classification process.

Table 28 - Data filtration component description

4.2.1.3. **Data classification** (Process 6-10)

Next data processing component is classification of data for MCOICOP and ICP classification, in the design approach is to create and train a machine learning model to automate classification of data to be classified for MCOICOP and ICP for new dataset.

System will ingest product descriptions and price information, preprocess the data, extract meaningful embeddings, classify the data using an machine learning model (SVM, Random Forest, or logistic regression) and enhance classification accuracy with a cost-efficient LLM-based approach.

Assumptions & Constraints

- Data sources are structured and contain relevant product detail.
- The system operates within computational constraints, optimizing for performance and cost.
- LLM API usage is to be minimized to reduce token costs.

System follows a hybrid classification approach integrating **Sentence Transformers, machine learning model** such as Support Vector Machines (SVM), Random Forest (RF) or logistic regression, and **Large Language Models (LLMs)** for high-confidence predictions.

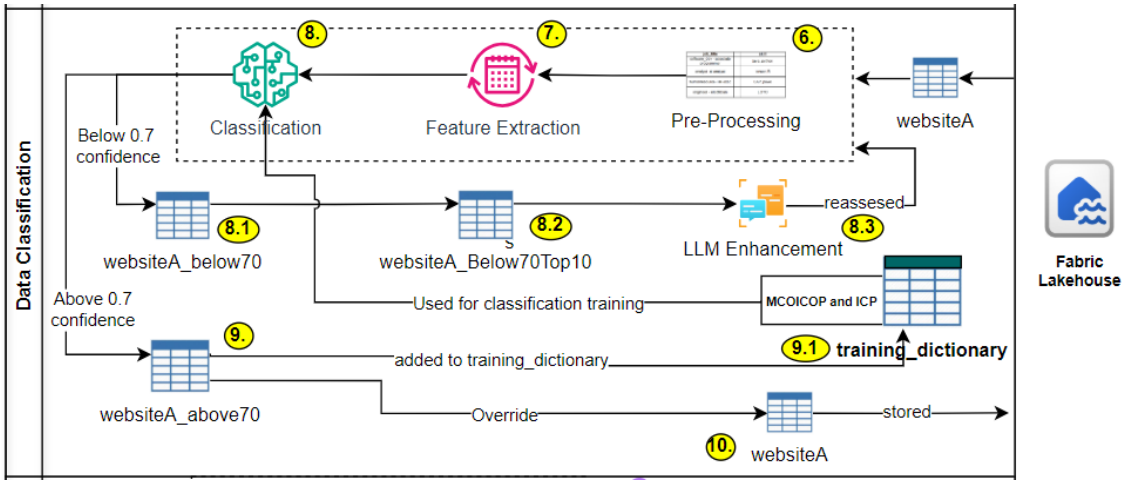


Figure 32 - Detailed data classification component design flow

Flow	Description
Process 6	**Pre-Processing** New scraped website dataset namely **websiteA,** will undergo **pre-processing ,** which is a process of making the title of item to be classified clear and distinguishable for machine learning model to process, for this case the **title** and **category from website** is combine to give better context of item being classified, e.g canned foods – ayamas tin sardin (category –title)

Table 29 - Pre-processing classification component description

combined_text
grocery-snacks-crisps-chips-LAYS ROCK EXTRA BBQ 50G
household-air-freshener-gel-liquid-SAWADAY MOS-BYE LAVENDER 275ML
chilled-frozen-dairy-butter-margarine-LURPAK SPREADABLE SALTED BUTTER 250G
beverages-carbonated-drinks-fruity-others-7UP BOTTLE 1.5L

Figure 33 - Pre-Processing example result

Flow	Description
Process 7	**Feature extraction** Process done by **Sentence transformer** to convert the **category-title** and the **MCOICOP & ICP** category into text and label numerical embedding, which would allow for the machine learning model to quantify and make calculative decision in assigning the data item with its highest likely MCOICOP category.

Table 30 - Feature exration classification component description

Flow	Description
Process 8	**Machine learning classification and scoring** The numerical embeddings of text and label obtained through sentence transformer go through a process of training using machine learning model (**Support Vector Machine (SVM), Random Forest or Logistic Regression)** which would classify and give probability matching score based on its likeliness that data item belongs to correct classification, in this case with help of **training dictionary table**, that contains past data item correctly classified to its MCOICOP categories. The result of classification matching & probability scoring will produce listed data item with its probability score with set threshold of **0.7 or 70%** as initial baseline to distinguished high probability matching or low. The two result are separated into each table, **below70** and **above70** This **threshold can be adjusted** to other value than 0.7 by analysing if the model performed well referring to matrix score and manually checking the classification.

Table 31 - Machine learning classification component description

combined_text	Sub Category	Confidence Score
meat-poultry-poultry-parts-weighted-WHOLE CHICKEN WITH HEAD AND FEET (7730)	Meat, fresh, chilled or frozen	0.982688
fresh-produce-vegetables-shoots-roots-HALIA MUDA /YOUNG GINGER-KG (3541)	Other vegetables, fresh or chilled	0.962054
beverages-coffee-tea-instant-coffee-NESCAFE CLASSIC REFILL PACK 200G	Coffee & coffee substitutes	0.99255
grocery-snacks-other-snacks-MAMEE MONSTER BBQ 8X25G	Other sugar confectionery & desserts n.e	0.988366
grocery-snacks-other-snacks-MAMEE MONSTER BLACK PEPPER 8X25G	Other sugar confectionery & desserts n.e	0.990336
fresh-produce-vegetables-vagetables-mushrooms-CF WHITE BUTTON MUSHROOM SLICED 125G	Other vegetables, fresh or chilled	0.996138
fresh-produce-vegetables-shoots-roots-UBI GARUT CHINA/CHINA ARROWROOT-KG(3515)	Vegetables, tubers, plantains, cooking ba	0.915335
beverages-coffee-tea-instant-coffee-NESCAFE CLASSIC REFILL PACK 200G FOC 20G	Coffee & coffee substitutes	0.985375

Figure 34 - Machine learning model classification above 0.7 confidence score result

combined_text	Sub Category	Confidence Score
fresh-produce-eggs-chicken-eggs-LOTUSS FRESH CHICKEN EGGS 30S (B)	Eggs	0.339437
fresh-produce-noodles-tofu-condiments-condiments-LOTUSS CILI KERING BERTANGKAI 500G	Salt, condiments & sauc	0.199785
fresh-produce-vegetables-shoots-roots-LOTUSS ASPARAGUS 250G	Leafy or stem vegetable	0.489799
fresh-produce-eggs-chicken-eggs-LOTUSS FRESH CHICKEN EGGS 30S	Eggs	0.285135
fresh-produce-vegetables-shoots-roots-DAUN SADERI AUST/AUST CELERY - BDL	Leafy or stem vegetable	0.600052
fresh-produce-noodles-tofu-condiments-condiments-LOTUSS CILI KERING KERINTING 200G	Salt, condiments & sauc	0.385615

Figure 35 - Machine learning model classification below 0.7 confidence score result

Comparison between machine learning model and justification:

Machine learning model	Justification	Advantage	Disadvantage
Support Vector Machine (SVM)	SVM is effective in high-dimensional spaces and is computationally efficient.	Works well for text classification, handles large feature spaces, and provides a good balance between bias and variance.	Does not handle non-linear relationships well, and performance depends on proper hyperparameter tuning.
Random Forest (RF)	RF is an ensemble method that performs well on structured data and	Robust to overfitting, interpretable, and handles high-	Can be computationally expensive for large datasets.

	can capture non-linear relationships.	dimensional spaces well.	
Logistic Regression	A simple, interpretable model that performs well when categories are linearly separable.	Fast, interpretable, and provides probabilistic outputs.	Assumes linearity in feature space, which may limit performance.

Table 32 - Machine learning model comparison description

Reason on choosing Support Vector Machine (SVM) as classification model:

- It performs well in high-dimensional text classification tasks.
- It is computationally efficient with large feature spaces.
- It generalizes well on unseen data, given proper tuning.
- It is less prone to overfitting compared to non-regularized models.

Model Evaluation

Models will be evaluated based on:

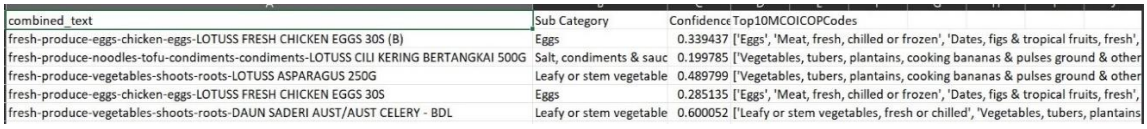
- Accuracy: Measures the proportion of correctly classified products.
- Precision, Recall, F1-score: Determines classification effectiveness across categories.

	precision	recall	f1-score	support
Leafy or stem vegetables, fresh or chilled	1.00	0.00	0.00	1
Baby food	1.00	1.00	1.00	2
Bread & bakery products	0.95	0.97	0.96	126
Breakfast cereals	0.97	1.00	0.98	56
Butter & other fats & oils derived from milk	0.91	1.00	0.95	10
Cane & beet sugar	1.00	1.00	1.00	5
Cereals	0.99	0.96	0.98	109
Cheese	1.00	1.00	1.00	65
Chocolate, cocoa, & cocoa-based food products	0.90	0.92	0.91	76
Citrus fruits, fresh	1.00	1.00	1.00	3
Cocoa drinks	1.00	0.67	0.80	6
Coffee & coffee substitutes	0.99	0.97	0.98	93
Dates, figs & tropical fruits, fresh	0.92	0.92	0.92	13
Eggs	1.00	1.00	1.00	7
Fish preparations	0.67	0.73	0.70	22
Fish, dried and salted	1.00	0.50	0.67	2
Fish, live, fresh, chilled or frozen	0.92	0.85	0.88	40
Flour of cereals	0.92	1.00	0.96	12

Figure 36 - Model precision scoring

Flow	Description
Process 8.1	Data item with probability scoring of 70% and below will be separated into a table
Process 8.2	The low probability dataset item is assigned **top 10 categories** based on its **highest similarity** with the help of sentence transformer.
Process 8.3	The data item with its respective **top 10 categories,** will undergo **Large Language Model (LLM)** processing where it will help to choose the best out of the top 10 categories based on the Data item.

Table 33 - Below 0.7 probability confidence score process description

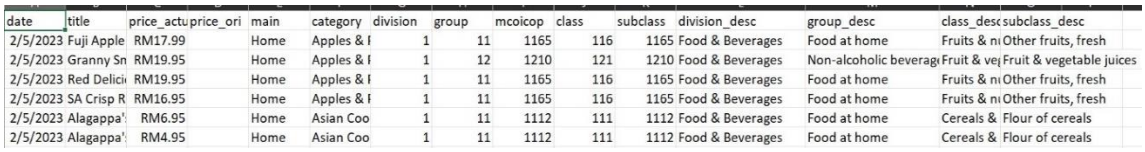


combined_text	Sub Category	Confidence	Top10MCOICOPCodes
fresh-produce-eggs-chicken-eggs-LOTUSS FRESH CHICKEN EGGS 30S (B)	Eggs	0.339437	['Eggs', 'Meat, fresh, chilled or frozen', 'Dates, figs & tropical fruits, fresh',
fresh-produce-noodles-tofu-condiments-condiments-LOTUSS CILI KERING BERTANGKAI 500G	Salt, condiments & sauc	0.199785	['Vegetables, tubers, plantains, cooking bananas & pulses ground & other
fresh-produce-vegetables-shoots-roots-LOTUSS ASPARAGUS 250G	Leafy or stem vegetable	0.489799	['Vegetables, tubers, plantains, cooking bananas & pulses ground & other
fresh-produce-eggs-chicken-eggs-LOTUSS FRESH CHICKEN EGGS 30S	Eggs	0.285135	['Eggs', 'Meat, fresh, chilled or frozen', 'Dates, figs & tropical fruits, fresh',
fresh-produce-vegetables-shoots-roots-DAUN SADERI AUST/AUST CELERY - BDL	Leafy or stem vegetable	0.600052	['Leafy or stem vegetables, fresh or chilled', 'Vegetables, tubers, plantains

Figure 37 - Top 10 classification for price item

Flow	Description
Process 9	Data item with probability score of **higher than 70%** will be separated into a table.
Process 9.1	The data item table with the high score is appended into the **training dictionary table** for future training & classification of data.
Process 10	The table item also overridden the initial **website name table (websiteA)** to be taken further in the pipeline process based on the Data item.

Table 34 - Above 0.7 probability confidence score process description



date	title	price_actu	price_ori	main	category	division	group	mcoicop	class	subclass	division_desc	group_desc	class_desc	subclass_desc
2/5/2023	Fuji Apple	RM17.99		Home	Apples & I	1	11	1165	116	1165	Food & Beverages	Food at home	Fruits & ni	Other fruits, fresh
2/5/2023	Granny Sn	RM19.95		Home	Apples & I	1	12	1210	121	1210	Food & Beverages	Non-alcoholic beverag	Fruit & ve	Fruit & vegetable juices
2/5/2023	Red Delici	RM19.95		Home	Apples & I	1	11	1165	116	1165	Food & Beverages	Food at home	Fruits & ni	Other fruits, fresh
2/5/2023	SA Crisp R	RM16.95		Home	Apples & I	1	11	1165	116	1165	Food & Beverages	Food at home	Fruits & ni	Other fruits, fresh
2/5/2023	Alagappa'	RM6.95		Home	Asian Coo	1	11	1112	111	1112	Food & Beverages	Food at home	Cereals &	Flour of cereals
2/5/2023	Alagappa'	RM4.95		Home	Asian Coo	1	11	1112	111	1112	Food & Beverages	Food at home	Cereals &	Flour of cereals

Figure 38 - Example final output data classified with MCOICOP (The output should also include website name)

4.2.2. Data enrichment Process Component (Process 11 and 12)

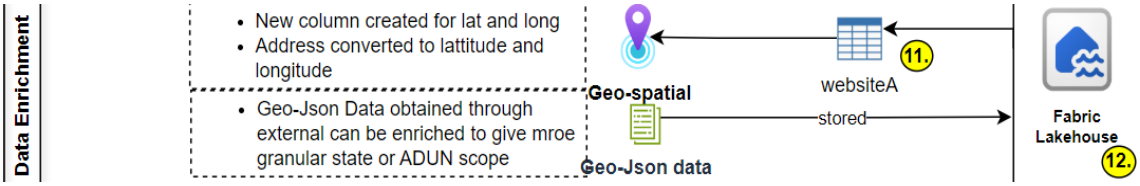


Figure 39 - Data enrichment component flow

Flow	Description
Process 11	Data is enriched with latitude and longitude value with reference from its address data availability. Through using Azure Geo-spatial feature, the address obtained is converted into latitude and longitude value for enrichment.
Process 12	Data converted into latitude and longitude is then cross-referenced with Geo-Json Data available which will give further ADUN, Parliament and State scope of the area for the property listing price area.

Table 35 - Data enrichment component description

Data is first check if it needed to be enriched, in this case, data enriching is only applicable for location as the location data is an additional useful data criterion to bring context and completes the data analysis overview.

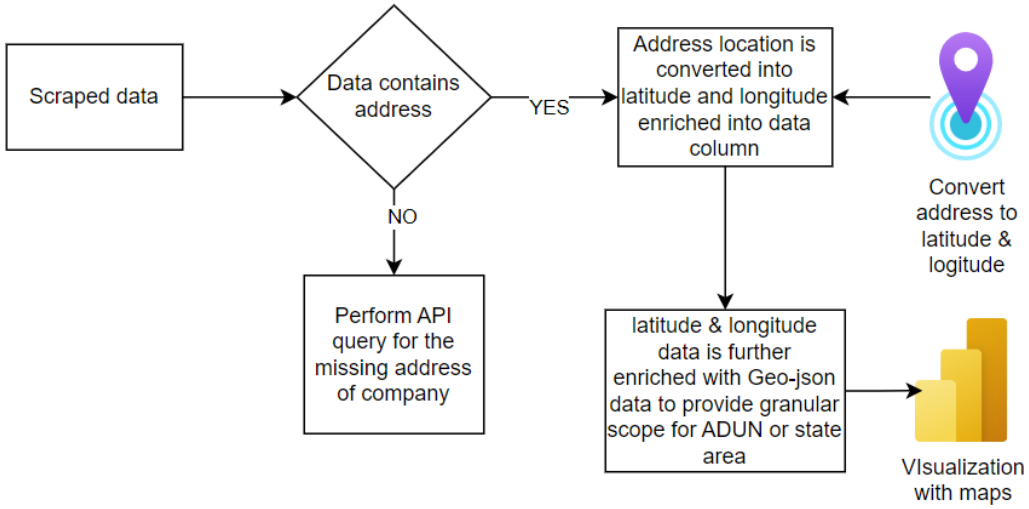


Figure 40 - Data Enrichment decision flow

Below are some of the data that requires location enrichment data:

Data category	Reason	Method
Property	To show the property heatmap or density of the property price data in a particular area or state.	Through address provided on the website for the property listed, latitude and longitude can be obtained through geo-spatial mapping API to convert address to latitude and longitude into heatmap.
E - Commerce or transport	Allowing external address gathered manually to be enriched in current dataset.	Through unique identifier mapping, if there is correlation to the existing dataset, it can be mapped and enriched to provide meaningful location analysis.

Table 36 - Data enrichment flow description

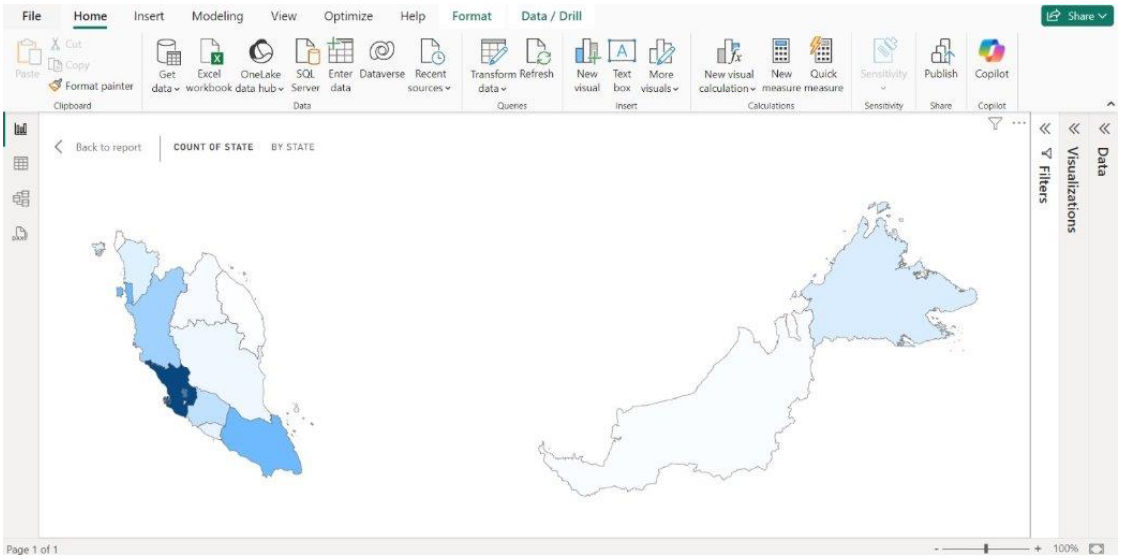


Figure 41 - Example Data location enrichment report visualization

4.2.3. **Data Quality Component** (Process 13-14)

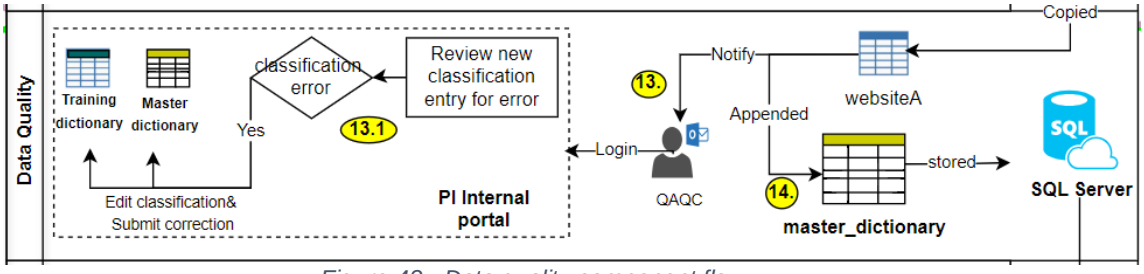


Figure 42 - Data quality component flow

Flow	Description
Process 13	Data item classified will be imported to **SQL server database as structured data**, the copying will trigger a notification email sent to user that a new set of classified dataset is coming from **WebsiteA** The SQL server is optimized for Create, Update and Delete operation which will be done in Internal portal
Process 13.1	**User or QA/QC team** can login to **PI Internal portal** where they can review and make changes if there is error in classification is detected from the machine learning classification. The corrected classification from data item **row** will trigger SQL UPDATE to the **training dictionary table and master dictionary table** corresponding row.
Process 14	Final dataset item from **website A** is appended to the **master dictionary table** where the staging of completed data classified according to **MCOICOP** resides and ready to be analysed.

Table 37 - Data quality flow description

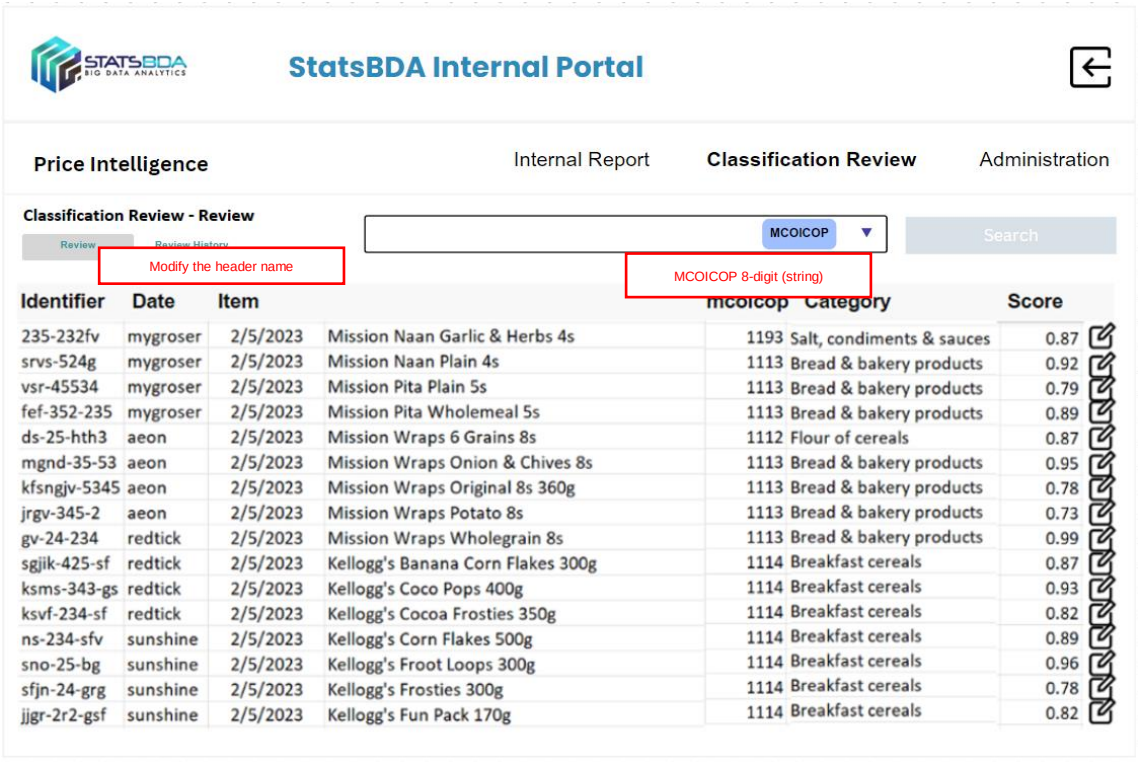


Figure 43 - Example PI QA/QC Internal portal for classication verification

4.3. Analytics and Visualization Component (Process 15 and 16)

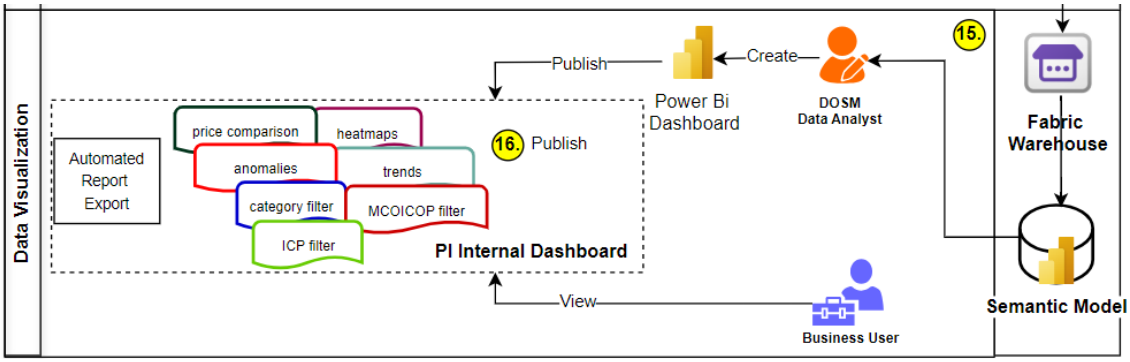


Figure 44 - Data visualization component flow

Flow	Description
Process 15	Master data from the warehouse will be connected to the Price Intelligence Power BI semantic model, and the refresh setting is set at 9 AM daily for refresh of new data entries from the warehouse any update to rows or column of master dictionary table can be done in the warehouse, and will be reflected in the power BI semantic model dataset is coming from WebsiteA
Process 16	Semantic model data can be used to create visualization and analysis as the main objective of Price Intelligence Module.The corrected classification from data item row will trigger SQL UPDATE to the training dictionary table and master dictionary table corresponding row. Analysis such as Time-series, anomaly/outliers detection, correlation analysis, comparative analysis, trend analysis and heatmaps can be done in Power BI. Power BI allows generation of reports and control of row-level security for access to certain report whether public or private. The report generation can be automated through Fabric data factory flow to send report to user email.

Table 38 - Data analytic and visualization component description

4.3.1. Reporting and Visualization Specifications

Below are few specification required for required reporting elements to have inside Power BI reporting and visualization

Process Name	Description
Trend Analysis	Use statistical techniques to calculate trends in pricing data, highlighting seasonal variations, spikes, or declines across product categories.
Anomaly and Outliers Detection	Identify outliers and unusual data points that may indicate significant price changes or potential data quality issues.
Comparative Analysis	Conduct a comparative analysis of prices across regions, vendors, and time periods, benchmarking these against past data or other indicators (e.g., CPI).
Correlation Analysis	Examine relationships between different variables (e.g., inflation vs. price changes in specific categories) to identify influencing factors.
Mapping MCOICOP and ICP Codes	Map relevant items to both MCOICOP and ICP standards for cross-national price comparisons.
Time Series Visualization	Create line graphs to display historical trends and monthly changes in pricing across different categories.
Comparative Visuals	Use bar charts or stacked column charts to compare price changes between multiple vendors, or product categories.
Indicator cards and KPI	Add KPI cards to highlight key indicators like average monthly inflation rates, category-specific price growth, or CPI-related insights.
Alert and Anomaly Indicator	Design alert symbols or color-coded indicators to flag significant price changes, allowing quick identification of areas that require attention.
Automated Report Scheduling	Schedule monthly reports to automatically compile and export summary data on key metrics, trends, and anomalies.

Export Formats	Provide export options in various formats (e.g., PDF, Excel, CSV) to enable sharing.

Table 39 - Power BI component specification description

4.3.1.1. E-Commerce category reporting and visualization

Feature	Visualization	Description
Trend Analysis	▪ Line Chart / Area Chart ▪ X-axis: Date ▪ Y-axis: Average Price or Price Index ▪ Legend: Category / Brand / SKU	To monitor price movement trends over time and detect seasonal or promotional fluctuations.
Anomaly & Outlier Detection	▪ Box Plot or Scatter Plot with Reference Lines ▪ Highlight data beyond ±2 SD	Identifies unusual price behavior due to system errors, promotions, or suspicious pricing.
Comparative Analysis	▪ Clustered Bar/Column Chart Compare by seller/platform/region	Enables side-by-side price comparisons across sources, platforms, or SKUs.
Correlation Analysis	▪ Scatter Plot Matrix or Correlation Heatmap (R/Python visual)	Helps identify relationships like price vs. rating or discount vs. availability.
Mapping MCOICOP & ICP Codes	▪ Matrix Table / Hierarchical Slicer / Tree Map with Drill-down	Aligns eCommerce product data with official classification codes for structured reporting.
Time Series Visualization	▪ Line Chart with Forecast and Decomposition	To analyze seasonality, trends, and forecast future price movements.
Comparative Visuals (Extended)	▪ Butterfly Chart / Multi-row Card Table ▪ Conditional formatting for highlighting differences	To show price deltas between platforms, brands, or sellers for the same product.
Indicator Cards & KPI	▪ KPI Card / Multi-row Card / Gauge Chart Examples: Avg MoM Price, Highest SKU, Total SKUs	Offers quick-glance metrics for key performance indicators in the pricing dataset.
Alert & Anomaly Indicator	▪ Table with Conditional Icons or Color Formatting ▪ Custom DAX rules for anomalies (e.g., flag when price change >20%)	Automatically flags SKUs with significant changes for further investigation.

Table 40 - E-commerce category Power BI visualization description

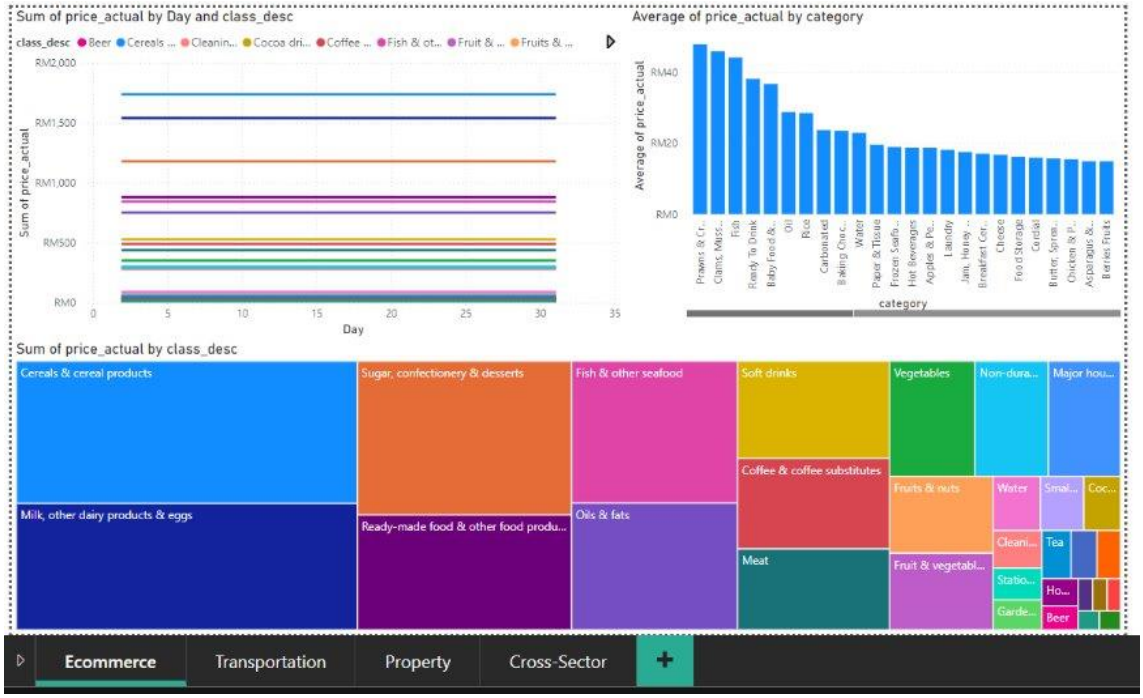


Figure 45 - E-commerce Power BI visualization

4.3.1.2. Transport category reporting and visualization

Feature	Visualization	Description
Trend Analysis	▪ Line Chart / Area Chart ▪ X-axis: Date ▪ Y-axis: Ticket Price or Fare Index ▪ Legend: Transport Type (Bus, Train, Flight)	Tracks how fares change over time due to fuel price shifts, seasonal surges, or new regulations.
Anomaly & Outlier Detection	▪ Box Plot / Scatter Plot with Upper/Lower Bound Indicators	Highlights sudden price spikes (e.g., during holidays) or system glitches in fare listings.
Comparative Analysis	▪ Clustered Column Chart or Bar Chart Compare across modes (Bus vs Train), platforms (RedBus vs EasyBook), or routes	Shows how different transport services price the same route, aiding in fare benchmarking.
Correlation Analysis	▪ Scatter Plot Matrix / Correlation Heatmap (using R/Python visual) ▪ Variables: Price vs Distance, Booking Time, Load Factor	Analyzes relationships like “distance vs fare,” “early booking vs discount,” etc.
Time Series Visualization	▪ Line Chart with Forecasting/Decomposition Tree	Allows modeling of future price trends (e.g., airfare projections during peak seasons).

Comparative Visuals (Extended)	▪ Butterfly Chart / Delta Table / Dual-Axis Bar Chart ▪ Compare ticket prices today vs last month or between platforms	Helps spot pricing differences between service providers or before/after promotions/events.
Indicator Cards & KPI	▪ KPI Cards (e.g., Average Fare, Highest Daily Fare, % MoM Change) ▪ Multi-row Cards for route-specific stats	Quickly display transport metrics and trends to stakeholders for regular monitoring.
Alert & Anomaly Indicator	▪ Table with Conditional Formatting or Custom Alert Icon ▪ E.g., Show ⚠ if fare jumped >30% vs previous day	Detects sudden fare inflation—could indicate errors, peak demand, or operational issues.

Table 41 - Transportation category Power BI visualization description

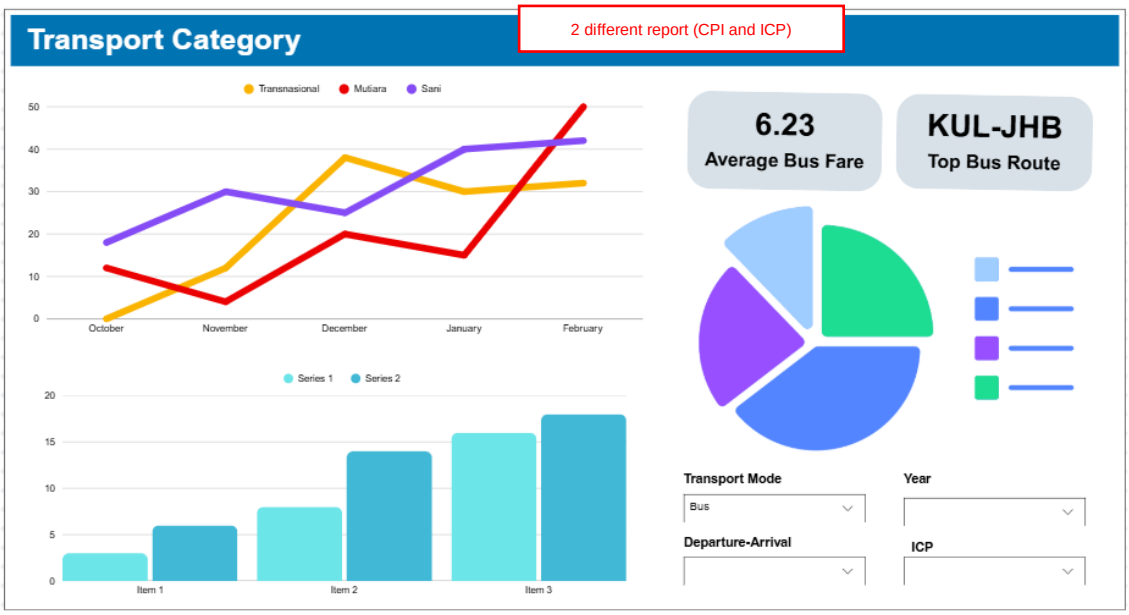


Figure 46 - Trasnportation Power BI visualization

4.3.1.3. Property category reporting and visualization

Feature	Visualization	Description
Trend Analysis	▪ Line Chart / Area Chart ▪ X-axis: Date ▪ Y-axis: Average Price (or PSF - price per sq ft) ▪ Legend: Property Type (Condo, Landed, etc.)	Tracks market trends like property appreciation/depreciation, especially useful in high-demand cities or segments.
Anomaly & Outlier Detection	▪ Box Plot / Scatter Plot with Outlier Highlighting	Flags listings with unusually high/low prices, which could be input errors, fire sales, or luxury properties.

Comparative Analysis	▪ Clustered Column/Bar Chart ▪ Compare by Property Type, District, Developer, or Platform	Helps compare prices between property categories or listing platforms (e.g., Speedhome vs PropertyGuru).
Correlation Analysis	▪ Scatter Plot Matrix / Correlation Heatmap ▪ Variables: Price vs Size, Location, Age, Facilities, Floor Level	Analyzes what factors drive property prices—e.g., do newer or larger units command a premium?
Time Series Visualization	▪ Line Chart with Forecast Line or Decomposition Tree	Enables forecasting of future pricing trends based on seasonality, market cycles, or macroeconomic signals.
Comparative Visuals (Extended)	▪ Butterfly Chart or Dual-Axis Bar ▪ Example: Asking Price vs Transaction Price ▪ Listing Price Last Month vs This Month	Shows deltas between expectations and market reality or month-over-month listing behavior.
Heatmaps & Geo Distribution	▪ Map Visual (ArcGIS or Filled Map) with Heat Layer ▪ Use coordinates or postal codes	Visualizes price concentrations across neighborhoods (e.g., KLCC vs Cheras), showing hotspots and cold zones. Visualizes geographical price variations across regions includes Parliament, ADUN and state.
Indicator Cards & KPI	▪ KPI Cards: Avg Asking Price, Median PSF, % Increase MoM ▪ Multi-row Card: Most Expensive Area, Fastest Growing Area	Quickly communicates key market indicators for decision-makers, realtors, or investors.
Alert & Anomaly Indicator	▪ Table with Conditional Icons (e.g., red/yellow for pricing jump or drop thresholds)	Highlights abnormal price changes that may require re-evaluation or manual check (e.g., data scraping anomaly)..

Table 42 - Property category Power BI visualization description

4.3.1.4. Reporting features

Feature	Service	Description
Automated Report Scheduling	▪ Power BI Service: Scheduled Refresh + Email Subscription	Delivers updated reports to stakeholders on a defined schedule, ensuring timely insights

Export Formats	▪ Export as PDF, Excel, or PowerPoint via Power BI Service	Allows sharing insights externally or archiving for regulatory and review purposes.

Table 43 – Reporting features

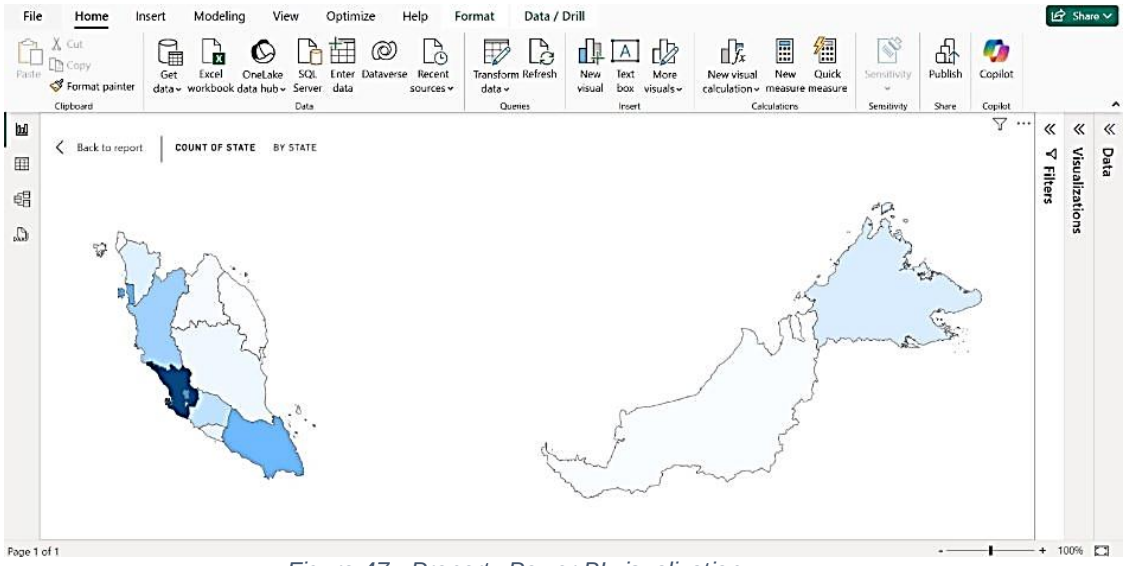


Figure 47 - Property Power BI visualization

4.3.2. Anomaly Detection Algorithms and Alert Mechanisms Designs.

Price changes anomaly detection design

Price changes anomaly detection is done through outlier detection through Z-Score analysis. The Z-score approach is a statistical method used to identify anomalies in the dataset by measuring how far a data point deviates from the mean in terms of standard deviations.

The threshold is set based on

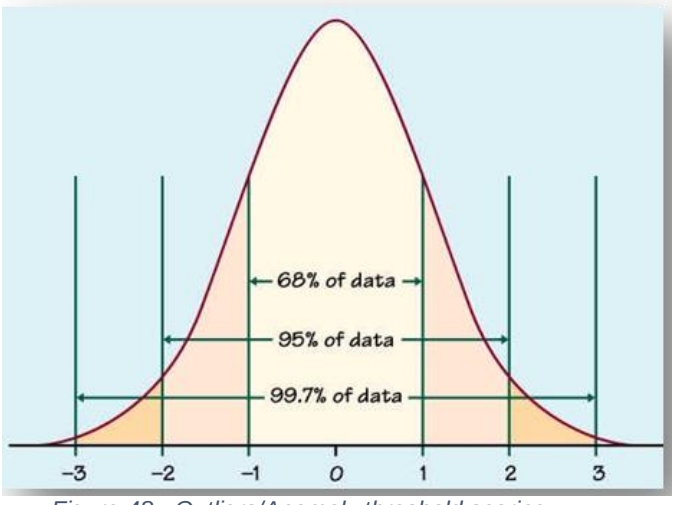


Figure 48 - Outliers/Anomaly threshold scoring

- Positive Z-scores (**+1, +2, +3**) mean the value is above the mean.
- Negative Z-scores (**-1, -2, -3**) mean the value is below the mean.

Threshold	Confidence Interval	Meaning	Use case
±1	68%	Minor deviations; typical variation	Ignore unless very sensitive.
±2	95%	Moderate anomalies; unusual	Investigate for potential issues.
±3	99.7%	Extreme anomalies; rare	Immediate action required. Alert sent to respective group.

Table 44 - Anomaly/outliers threshold interval description

Calculation formula for price change using Z-Score analysis:

The Z-score calculates how far a specific data point (e.g., the current price of an item) deviates from the mean of historical prices, in terms of standard deviations. The general formula is:

Formula	**Formula description**
Z = (X – μ) / σ	**Z**: The Z-score, which represents the number of standard deviations the current price deviates from the mean. **X**: The current price of the item. **M**: The mean (average) of historical prices for the item. **Σ**: The standard deviation of historical prices for the item.

Table 45 - Anomaly/outliers formula description

Integration with the Solution

- The Z-score-based outlier detection will be implemented as part of the data cleansing and standardization pipeline.
- The calculation will be performed using Python within the **Microsoft Fabric Notebook** environment.
- Outliers will be flagged, listed, and send through email to the user requested email or shared email group.

Price changes anomaly flow

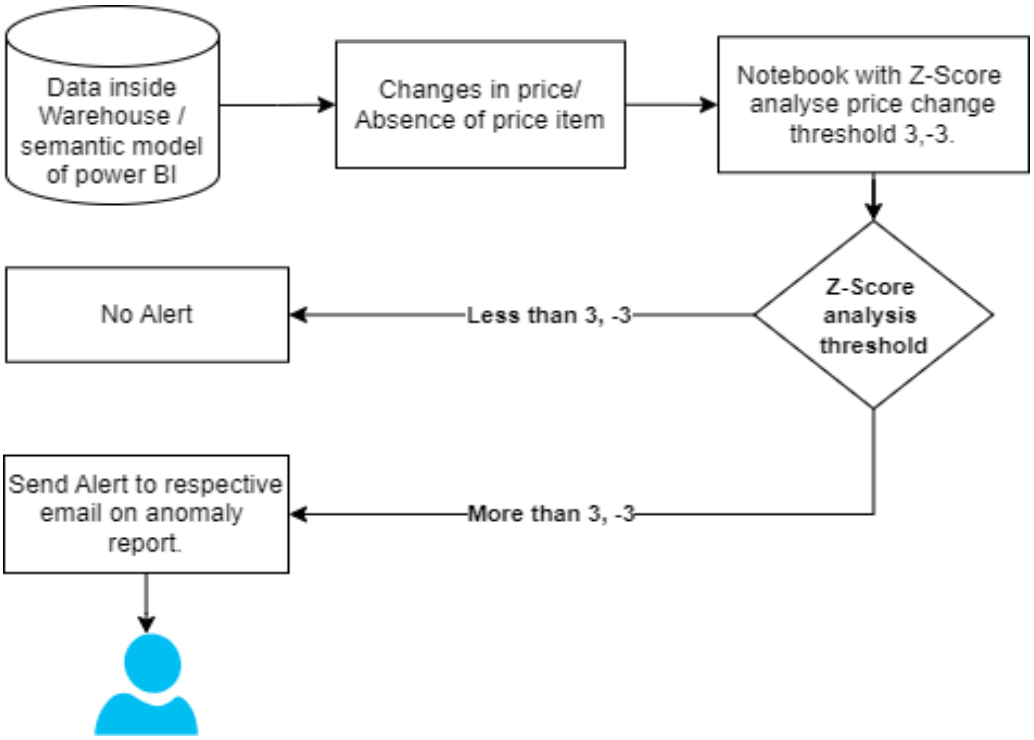


Figure 49 - Anomaly decision flow

Flow	Description
Changes in price/Absence of price item	Any changes in price or absence of price item are analysed automatically using z-score analysis notebook.
Notebook with z-score analysis detect price changes threshold of 3, -3	Price changes is first analysed whether the price changes is above a certain threshold, in this case the z-score threshold is set at 3, -3
Z-Score analysis threshold	Price item changes that is above 3, -3 it will be detected as anomaly and an email report of the anomaly price item will be sent to the respective user or shared email group. Price item changes that are below 3, -3 will not be detected as anomaly and no email alerts is sent.

Table 46 - Anomaly/outliers detection flow description

5. Database and Storage Design

5.1. Data Schema Design

This schema defines the structure of a centralized database designed to store **aggregated data across e-commerce, Transport and property categories** in Price Intelligence module. It consolidates product information, pricing, availability, and metadata from multiple sources (e.g., retail websites, online marketplaces) into a unified format.

This schema apply for storage include:

i. **Fabric Lakehouse**
Initial storage for processing data in delta table format

ii. **SQL Table**
Storage of data dictionary for classification review and update through portal interface

iii. **Fabric Warehouse**
Storage of data dictionary for analytics and visualizations report

The schema is subject to change depends on future requirements.

5.1.1. E- Commerce category aggregated database schema

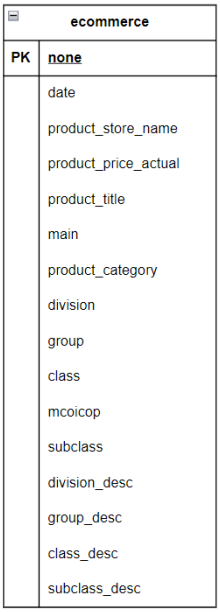


Figure 50 - E-commerce category database schema

Category	Column	Description
E-Commerce	date	Date data is gathered
	product_store_name	The name of store webpage (myaeon2go.com, mygroser.com)
	product_price_actual	Actual price before discount
	product_title	The name of the product item
	main	Default page where data was scraped
	product_category	Category of product where it is scraped
	main_group	Main group digit category of MCOICOP (e.g 01 – Food & Beverages)
	sub_group	Sub Group digit category of MCOICOP (e.g 011 – food at home)
	expenditure_class	Expenditure class digit category of MCOICOP (e.g 0116 – Fruits & Nuts)
	sub_expenditure_class	Sub expenditure class digit category of MCOICOP (e.g 01165 – Other Fruits, fresh)
	expenditure_item	Expenditure Item digit category of MCOICOP (e.g 011651 – Grapes, Fresh)
	Item	Item digit category of MCOICOP (e.g 01165101 – Red Grapes)
	mcoicop	Final MCOICOP code classification for the item (e.g 01165101)
	icp	Final ICP code classification for the item (e.g 011111001 - Rice)
	main_group_desc	Description of the main group digit code
	sub_group_desc	Description of the sub group digit code
	expenditure_class_desc	Description of the expenditure class digit code
	sub_expenditure_class_desc	Description of the sub expenditure class digit code
	expenditure_item_desc	Description of the Expenditure Item digit code
	item_desc	Description of the Item digit code

Table 47 - E-commerce category database schema

5.1.2. Transport category aggregated database schema

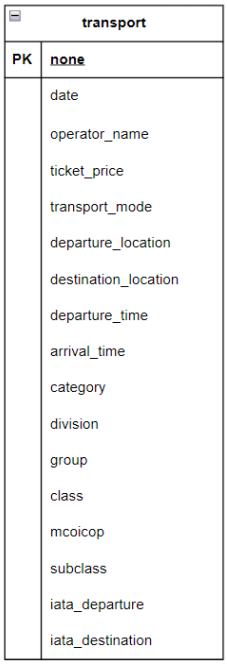


Figure 51 - Transport category database schema

Category	Column	Description
transport	date	Date data is gathered
	operator_name	The name of operator (AirAsia, Plusliner ,KTMB)
	ticket_price	Actual price before discount
	transport_mode	The mode of transport (bus,flight,train)
	departure_location	Departure location of item
	destination_location	Destination location of item
	departure_time	The time of departure
	arrival_time	The time of arrival
	category	Class digit category of MCOICOP (e.g 07 – transport)
	subcategory	subcategory digit category of MCOICOP (e.g 07.3 – transport services)
	class	Class digit category of MCOICOP (e.g 07.3.1 – passenger transport by railway)
	category_desc	Description of the category digit code
	subcategory	Description of the subcategory digit code
	mcoicop	Final MCOICOP classification digit code
	icp	Final ICP code classification for the item
	iata_departure	IATA code used for airline and airport (e.g KUL, PEN, KCH)
	iata_destination	IATA code used for airline and airport (e.g KUL, PEN, KCH)

Table 48 - Transport category database schema

5.1.3. Property category aggregated database schema

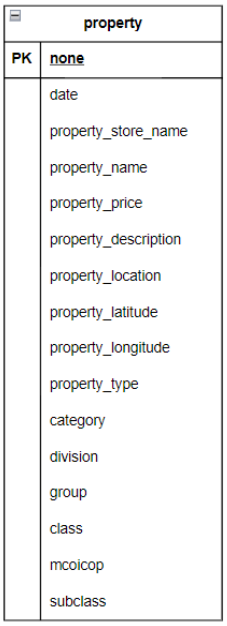


Figure 52 - Property category database schema

Category	Column	Description
Property	date	Date data is gathered
	property_owner_name	The name of property owner (HF properties, az properties)
	property_name	Name of property listings
	property_price	Price of properties
	property_description	Description of properties
	property_location	Location address of properties
	property_latitude	Latitude coordinate of property
	property_longitude	Longitude coordinate of property
	property_type	The type of property (two storey, bungalow, apartment)
	listing_type	The type of property listed as (rent, for sale)
	division	Division digit category of MCOICOP (e.g 04 – Housing)
	group	Division digit category of MCOICOP (e.g 04.1 – Actual rentals)
	mcoicop	Final MCOICOP digit classifcation
	icp	Final ICP code classification for the item

Table 49 - Property category database schema

5.1.4. Historical data storage and retrieval processes

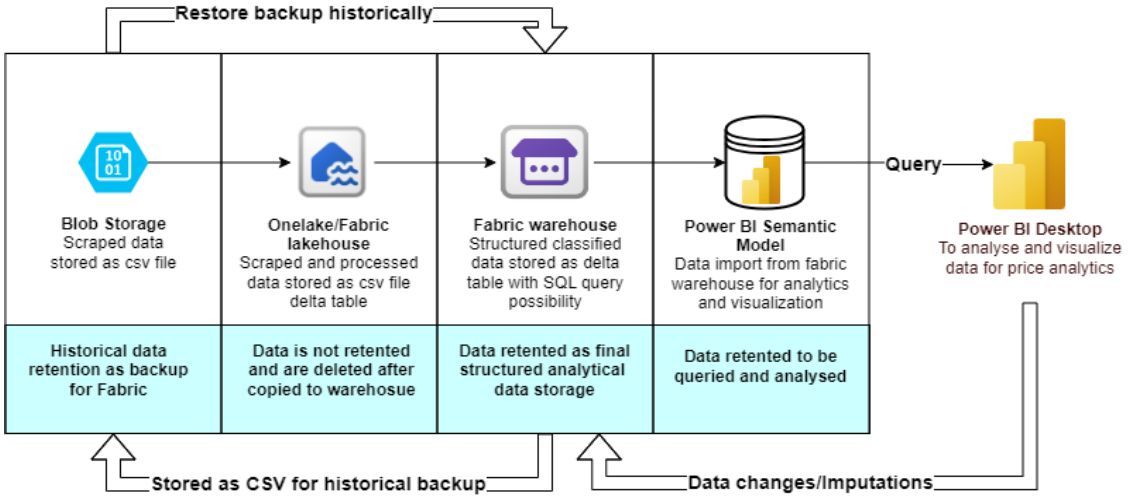


Figure 53 - Historical data storage & Retrieval flow

Service role	Description
Blob Storage	**Historical Backup Storage** Blob storage is used as main place to store initial scraped data and as backup data for structured and classified historical data from Fabric environment in .csv format. Role of blob storage as backup storage is due to ability to save cost through data lifecycle retention policy to allow for deletion and tiering of data storage type based on hot, cold or archive to save retention cost over time.
Onelake/Fabric lakehouse	Data inside Fabric Onelake/lakehouse is not stored for long term and is usually deleted after structured data in delta table is copied to warehouse.
Fabric warehouse	Data stored inside warehouse is a final data staging area for structured and classified data, data in warehouse is used for querying and visualization using power BI through semantic model. Data that is needed to fix, imputations and changes is done in the warehouse.
Power BI semantic model	Data inside semantic model is the imported data from Fabric warehouse and is further modified to fit visualization context.

Table 50 - Historical data storage & retrieval flow description

5.2. Data Retention and Security Planning

5.2.1. Data retention policies for historical records.

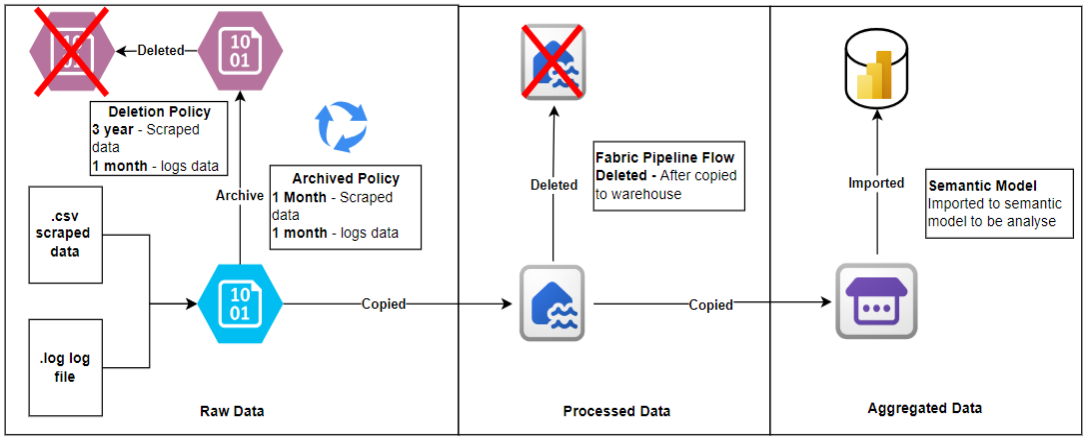


Figure 54 - Data retention policy flow

Data Type	Data Stored	Storage Location	Destination	Retention Period	Description
Raw Data	.csv file of scraped data	Azure Blob Storage	Blob **Archive tier**	1 Month to archive	Scraped data is stored for 1 months in blob **hot tier** for reference, and then moved to Archive for historical backup store.
	.log file of log file from Power automate and Fabric monitor	Azure Blob Storage	Blob **Archive tier**	1 Month	Log data from power automate and Fabric environment stored in blob for 1 month for reference until it is moved to archive tier mode.
Processed Data	- .csv of scraped data - Delta table of processed data	Microsoft Fabric Lakehouse	**Deleted**	Until data copied to warehouse	Scraped, cleansed and classified data stored as temporary staging for processing data. Deleted after structured data

					transferred to warehouse.
Aggregated Data	Delta table of processed and classified data	Microsoft Fabric Warehouse	Warehouse and Power BI Semantic model	Indefinite	Data stored optimized for analysis, querying and imputations of data in semantic model

Table 51 - Data retention policy flow description

Retention policy tool

For different storage environment uses two different lifecycle tools to handle data retention lifecycle.

Storage service	Lifecycle Policy tool	Process
Blob storage	Azure Storage Lifecycle Policy	Setting in Azure Storage Lifecycle policy allow change of tiering from **hot to archive mode**
Fabric Lakehouse Fabric Warehouse	Data factory Pipeline flow	Pipeline allow execution of **deletion** job and **copying** job.

Table 52 - Data retention policy tool description

5.2.2. Encryption and access control mechanisms design for sensitive data.

Microsoft Purview plays a crucial role in enforcing encryption, access control, and governance for sensitive data collected through the data crawling process. The following security mechanisms ensure **data confidentiality, integrity, and controlled access**:

Encryption for **data at rest** and **data in transit** in **Azure Blob Storage**, **Microsoft Fabric Lakehouse (OneLake)**, and **Fabric Warehouse** is designed to ensure the confidentiality, integrity, and security of data. Here's an overview of the encryption mechanisms used for these services:

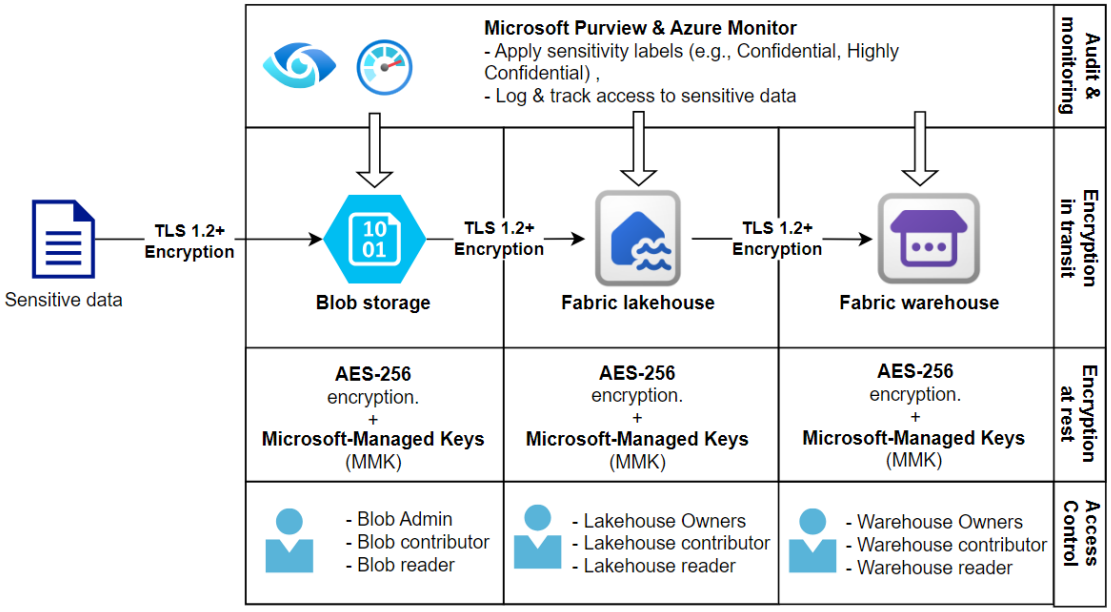


Figure 55 - Encryption mechanism flow

5.2.2.1. Encryption

Encryption type	Encryption	Service
Data at Rest	▪ Use **Azure Storage Encryption (AES-256) Microsoft-managed keys** by default for data in Blob Storage. ▪ Enable encryption for data stored in **Microsoft Fabric Lakehouse** and **Warehouse**.	▪ Blob Storage ▪ Fabric Lakehouse ▪ Fabric Warehouse
Data in Transit	▪ Secure all data transfers with **TLS 1.2+** to prevent interception during processing.	▪ All data transfer in Azure and Fabric environment

Table 53 - Encryption mechanism flow description

5.2.2.2. Access Control

Access control type	Access Control	Service
Role-Based Access Control (RBAC)	▪ Manage access permissions using **Microsoft Entra ID** roles. ▪ Define specific roles for administrators, analysts, and ETL developers, ensuring the principle of least privilege.	Microsoft Entra ID

Microsoft Purview Policies	▪ Enforce access control and data sensitivity labels using **Microsoft Purview**. ▪ Classify and protect sensitive data based on its type (e.g., MCOICOP or ICP classifications).	Microsoft Purview

Table 54 - Access control mechanism design

5.2.2.3. Auditing and Monitoring

Type	Implementation	Service
Monitoring	▪ Use **Azure Monitor** to log and track all access to sensitive data.	Azure Monitor
Audit	▪ Implement **Microsoft Purview Data Insights** to detect and report anomalies in data access patterns.	Microsoft Purview

Table 55 - Audit and Monitoring mechanism design

Microsoft Purview features:

- **AI-powered scanning** detects sensitive data (e.g., **credit card numbers, personal information**).
- Applies **sensitivity labels** (e.g., **Confidential, Restricted**) to enforce encryption and access policies automatically.
- Works across multiple data stores, ensuring **uniform security compliance**.

6. Security and Access Control

6.1. Authentication and Role-Based Access

6.1.1. Setting admin/analyst roles & Microsoft Entra ID.

To ensure secure and controlled access to sensitive data and services, the solution will implement **Role-Based Access Control (RBAC)** policies across **Azure Blob Storage**, **Fabric Lakehouse (OneLake)**, **Fabric Notebook**, **Fabric Warehouse and Power BI**. The RBAC policy will assign roles and permission for accessing certain services and specific scope of security align with the principles of **least privilege access** and ensure compliance with organizational and regulatory requirements.

Access Scope

Assign RBAC permissions at the following levels to ensure granular control:

a) **Azure Blob Storage**:
   - **Account Level**: Manage encryption, networking, and shared access policies.
   - **Container Level**: Assign permissions for specific datasets (e.g., read, write, or delete).

b) **Fabric Lakehouse**:
   - **Workspace Level**: Manage access to datasets, notebooks, and reports.
   - **Data Pipeline Level**: Restrict access to ingestion and transformation pipelines.

c) **Fabric Warehouse**:
   - **Database Level**: Manage access to the entire database.
   - **Table Level**: Enforce row-level security (RLS) or column masking for sensitive fields.

d) **Power BI:**
   - **Sematic Level:** Manage access to semantic model to allow changes to imported data.
   - **Report Level:** Manage access to report visualization and modifications.

Below is a more detailed scope of RBAC permissions for each service:

6.1.1.1. Blob Storage Security:

Role	Permission	Example users	DOSM PIC
Storage Admin	Full Control	Cloud Administrator	
Storage Blob Contributor	Upload/modify/delete data	ETL Process/Notebooks	
Storage Blob Data Reader	Read-Only Access	Data Analyst, Power BI	

Table 56 - Blob storage security role

6.1.1.2. Fabric Onelake Storage Security

Role	Permission	Example users	DOSM PIC
Lakehouse Owner	Full Control	Data Engineers	
Lakehouse Contributor	Add/Update data	ETL Process/Notebooks	
Lakehouse Reader	Read-Only Access	Data Analyst, Power BI	

Table 57 - Fabric Onelake security role

6.1.1.3. Fabric Notebook Security

Role	Permission	Example users	DOSM PIC
Notebook Owner	Full Control	Data Engineers	
Notebook Editor	Modify notebooks	ETL Developers	
Notebook Viewer	View-Only Access	Data Analyst	

Table 58 - Fabric Notebook security role

6.1.1.4. Warehouse Storage Security

Role	Permission	Example users	DOSM PIC
Warehouse Owner	Full Control	Database administrator	
Warehosue Editor	Insert/Update data	ETL Process	
Warehouse Reader	Read-Only Access	Data Analyst, Power BI	

Table 59 - Fabric Warehouse security role

6.1.1.5. Power BI Security

Role	Permission	Example users	DOSM PIC
Admin	Full Control	Power BI admins	
Member	Create/Edit Reports	Data Analyst	
Contributor	Publish Content	Report creator	
Viewer	View-Only Access	Business User	

Table 60 - Power BI security role

6.1.2. Configuring secure access.

To ensure a robust and secure access mechanism for the ETL infrastructure and associated storage solutions, the following authentication and access control configurations will be implemented:

6.1.2.1. OAuth 2.0 and OpenID Connect:

Overview	Implementation
OAuth ▪ Used for authorization, allowing applications to request access tokens on behalf of users or service accounts. ▪ Tokens grant time-limited access to resources such as Azure Blob Storage, Fabric Lakehouse, and Fabric Warehouse.	▪ Applications accessing storage services (e.g., ETL pipelines) will use Azure AD's app registration to acquire access tokens. ▪ User roles and permissions will be assigned based on **Role-Based Access Control (RBAC)** policies.

OpenID Connect ▪ Extends OAuth 2.0 by adding an ID token, enabling authentication of users. ▪ The ID token includes user information (e.g., email, roles) securely signed by Azure AD. ▪ Enables Single Sign-On (SSO) across the ETL system and related services	

Table 61 - Oauth and OpenID authentication

6.1.2.2. Multi-Factor Authentication (MFA)

Overview	Implementation	Use Case
Add an additional layer of security to the authentication process by requiring users to verify their identity using multiple factors (e.g., password + mobile app notification or hardware token).	Enforce MFA for all users accessing the ETL infrastructure, including developers, data engineers, and administrators. Supported MFA methods include: ▪ Microsoft Authenticator app ▪ SMS or phone call verification	▪ Protect sensitive data in **Azure Blob Storage**, **Fabric Lakehouse**, and **Fabric Warehouse** from unauthorized access, even in case of compromised credentials. ▪ Ensure MFA is applied consistently across web applications, API endpoints, and management portals.

Table 62 - MFA Authentication

6.1.2.3. Conditional Access Policy

Overview	Implementation	Key Policies
Conditional Access Policies in Azure AD will enforce security rules based on user context, device, location, and application.	▪ Configure policies in Azure AD under **Security > Conditional Access**. ▪ Apply specific conditions to protect sensitive data resources and enforce secure access to ETL tools.	▪ **Location-Based Access**: Allow access only from trusted IP ranges (e.g., corporate VPN or specific office locations). ▪ **Device Compliance**: Require users to access resources only from compliant and managed devices. ▪ **Risk-Based Access**: Trigger additional authentication steps if the user's login

		behavior is deemed risky (e.g., logins from unusual locations). ▪ **Access Based on Role**: Restrict high-privileged roles (e.g., Database Admins) to use MFA and access only during working hours

Table 63 - Conditional Access Policy

6.1.2.4. Managed Identity for Applications

Overview	Implementation	Key Benefits
Eliminate the need for hardcoding credentials or secrets in applications accessing storage services and ETL tools. These identities allow applications to authenticate securely and retrieve access tokens for Azure services.	Assign **Managed Identities** to all ETL components (e.g., pipelines, external resources) to securely interact with: • **Azure Blob Storage** for data ingestion and processing. • **Fabric Lakehouse (OneLake)** for storing and transforming structured data. • **Fabric Warehouse** for querying and reporting. Grant appropriate RBAC roles to the managed identities for each service.	▪ Improved security: No secrets or credentials are exposed in code or configuration files. ▪ Simplified access control: Permissions are managed directly in Azure AD.

Table 64 - Managed Identity for application

6.2. Data Protection Planning

6.2.1. Encryption protocols and data protection regulations compliance.

6.2.1.1. Data privacy laws and standards

Below is the description of each compliance framework, that in comply with design solution:

Compliance Standard	Description
GDPR (General Data Protection Regulation)	A European Union regulation that governs data protection and privacy for individuals in the EU and EEA.
HIPAA (Health Insurance Portability and Accountability Act)	A US law that protects sensitive health information (PHI) from being disclosed without consent.
ISO 27001	An international standard for information security management systems (ISMS), ensuring organizations follow best practices for data protection.
SOC (System and Organization Controls)	A framework (SOC 1, SOC 2, SOC 3) for auditing an organization's controls over financial reporting (SOC 1) and data security (SOC 2 & 3).

Table 65 - Description of each compliance framwork

- **AES-256 encryption** (Advanced Encryption Standard with 256-bit keys).
- **Microsoft-Managed Keys (MMK)**: Encryption keys managed by Microsoft.
- **Customer-Managed Keys (CMK)**: Stored in **Azure Key Vault** or **Azure Key Vault Managed**

Feature	Azure Blob Storage	Fabric Lakehouse (OneLake)	Fabric Warehouse
Data at Rest Encryption	AES-256 (SSE)	AES-256 (SSE)	AES-256 (SSE)
Key Management Options	MMK, CMK, CPK	MMK, CMK (future support)	MMK, CMK (future support)
Data in Transit Encryption	TLS 1.2+	TLS 1.2+	TLS 1.2+
Compliance	GDPR, HIPAA, ISO 27001, SOC	GDPR, HIPAA, ISO 27001, SOC	GDPR, HIPAA, ISO 27001, SOC

Table 66 - Encryption protocol

a) **Data Classification and Governance**:
   - Leverage **Microsoft Purview** to classify and label sensitive data.
   - Apply **sensitivity labels** (e.g., MCOICOP, ICP classifications) for compliance with regulations like **GDPR**, **CCPA**, or industry-specific standards.

b) **Access Monitoring and Auditing**:
   - Enable **Purview Data Insights** and **Azure Monitor** to track data access and identify anomalies.
   - Maintain detailed audit logs for sensitive data, including encryption key usage and modifications.

c) **Regulatory Alignment**:
   - Map retention and protection policies to regulatory requirements such as **data minimization**, **right to be forgotten**, and **data transfer limitations**.
   - Regularly review and update data protection measures to stay compliant with evolving regulations.

- End of Document -