TRANSMISI : JURNAL ILMIAH TEKNIK ELEKTRO, 26, (1), JANUARI 2024
p-ISSN 1411-0814 e-ISSN 2407-6422

BIG DATA PIPELINE INFRASTRUCTURE DESIGN IN MSME E-COMMERCE SYSTEMS WITH A FOCUS ON DATA SOURCE PROCESSING USING ORCHESTRATION TOOLS

Isro’ Rizky Wibowo[1], Ramadhan Rakhmat Sani[1], Ika Novita Dewi[1], Farrikh Al Zami[1*)], Ifan Rizqa[1], Abu Salam[1], Candra Irawan[1] dan Diana Aqmala[2]

[1]Faculty of Computer Science, Universitas Dian Nuswantoro, Semarang, Indonesia
[2]Faculty of Economics and Business, Universitas Dian Nuswantoro, Semarang, Indonesia

[*)]*E-mail: alzami@dsn.dinus.ac.id*

Abstract

In the digital era, Micro, Small and Medium Enterprises (MSMEs) need to utilize data to improve their business performance, such as increasing customer targets, product development and pricing strategies. Apache Airflow is a powerful tool for building data scraping pipelines that are scalable, flexible, and easy to monitor. One of them is the Central Java MSME data scraping pipeline, which collects business registration information, business type, location, contacts, products, and financial information from various websites, including the Central Java Provincial Government website, basic goods price comparison tables, and specialized news sites. The captured data is stored in a data warehouse for further analysis by the Central Java souvenir entrepreneurs association (ASPOO) in the region. Apache Airflow is used to manage the scraping pipeline in the Central Java MSME E-Commerce system and ensure it runs smoothly. Apache Airflow also has a built-in dashboard for monitoring pipelines and troubleshooting issues. Overall, scraping pipeline in the Central Java MSME e-commerce system is a valuable tool for collecting and analyzing data on the MSME sector in Central Java. This pipeline is scalable, flexible and easy to use, and can be adapted to different user needs and can be integrated with various systems.

Keywords: Big Data Pipeline, Data Scraping, Pipeline, Apache Airflow, MSMEs

Abstrak

Di era digital, Usaha Mikro, Kecil, dan Menengah (UMKM) perlu memanfaatkan data untuk meningkatkan kinerja bisnisnya, seperti peningkatan target pelanggan, pengembangan produk, dan strategi penetapan harga. Apache Airflow adalah alat yang kompatibel untuk membangun data *scraping pipelines* yang skalabel, fleksibel, dan mudah dipantau. Salah satunya adalah data *scraping pipelines* UMKM Jawa Tengah yang mengumpulkan informasi pendaftaran usaha, jenis usaha, lokasi, kontak, produk, dan informasi keuangan dari berbagai website, antara lain website milik pemerintah yang berisi tabel perbandingan harga barang pokok dan situs berita. Data yang diambil disimpan di data warehouse untuk dianalisis lebih lanjut oleh Asosiasi Pengusaha Oleh-Oleh Jawa Tengah (ASPOO) di wilayah tersebut. Apache Airflow digunakan untuk mengelola *scraping pipeline* didalam sistem *E-Commerce* UMKM Jawa Tengah dan memastikannya berjalan lancar. Apache Airflow juga memiliki dasbor bawaan untuk memantau saluran pipa dan memecahkan masalah. Secara keseluruhan, *scraping pipeline* pada sistem *e-commerce* UMKM Jawa Tengah merupakan alat yang berharga untuk mengumpulkan dan menganalisis data sektor UMKM di Jawa Tengah. Pipeline ini dapat diskalakan, fleksibel, dan mudah digunakan, serta dapat disesuaikan dengan kebutuhan pengguna yang berbeda dan dapat diintegrasikan dengan berbagai sistem.

Kata kunci: Big Data Pipeline, Scraping Data, Pipeline, Apache Airflow, UMKM

1. Introduction

Micro, Small and Medium Enterprises (MSMEs) are considered the backbone of the national economy because they are crucial in economic development and job creation [1]. In this rapidly developing digital era, the use of technology is also increasingly relevant to MSME business processes, especially in aspects of the e-commerce system. One of the key elements in running an e-commerce business is data collection and analysis, which often involves large amounts of data or Big Data. Big Data is described as very large amounts of data (both structured and unstructured) that require sophisticated database management systems to capture, store, manage and analyze that data [2]. The importance of Big Data storage technology in supply chain integration which can influence

SCP (Supply Chain Performance) which leads to sustainable business performance (i.e. environmental, social and economic) [3]. However, storing and managing Big Data is not a simple task. This is where the Big Data Pipeline concept becomes relevant. Big Data Pipeline is a series of steps and tools used to collect, transform and input data from various sources into a system that can be accessed and processed [4]. In the scope of performance on the MSME e-commerce platform, Big Data Pipeline plays a role in overcoming obstacles in organizing large data, converting it into a more structured format, and storing it in an appropriate system [5]. So that it can make more targeted decisions and based on accurate data analysis.

Web scraping is one way to optimize information utilization using the Big Data Pipeline. This strategy can be used to provide information to help make strategic decisions [6]. Web scraping is used because it provides daily information about Indonesia's economic activities, such as business and UMKM statistics, and price data for local goods, providing national price data every day. This program uses two websites to gather external data that can be integrated into the UMKM e-commerce system. Run in an orchestration tools environment so that the scraping program runs on an automated schedule, Selenium is used for browser optimization and dynamic web interaction, specifically BeautifulSoup, which parses and extracts HTML data, enabling efficient and accurate information extraction using Big Data Pipeline [7],[8].

Overall, the scope of this research is to examine the use of the Data Pipelining method in a Big Data environment by using orchestration tools combined with a scraping program to form a Big Data Pipeline concept which is useful for better data processing from various data sources. In an effort to build a data requirements flow system in the MSME e-commerce system, the stages of collecting, storing and analyzing Big Data play an important role. Using the Big Data Pipeline, MSMEs have started to explore different ways to utilize raw data [2]. This can help MSMEs to remain competitive in a business world that continues to transform into a more digital realm. In this way, this research is focused on developing a big data processing architecture based on a big data pipeline for the automated process of streaming data into database storage in the case of MSMEs to facilitate data collecting and data retrieving for MSMEs in a flexible manner that is adjusted to system needs.

2. Related Work

The new approach to big data processing requires a thorough analysis of the current research context and existing literature to identify key trends and issues. Previous studies have attempted various methods, but still face shortcomings and unresolved issues.

Many studies on orchestration and web scraping tools have been conducted. Orchestration tools can be integrated with various data processing programs to meet data needs, including cloud services in research [4]. Where an analysis process was carried out to check the capabilities of Apache Airflow to create data pipeline workflow management for DataCloud projects. Next, on big data processing services in research [9]. Apache Airflow is able to overcome the complexity of big data processing using the ETL (Extract, Transform, Load) method combined with the Apache Hadoop program to move data and convert from XML to Parquet, and PostgreSQL as a database for data transformation in testing using the KPIs dataset (Key Performance Indicators) in railway companies. Lastly, in research [10], when carrying out web scraping using UI Path Studio, limitations were also found in the research, one of which was that manual scraping was prone to errors and time consuming.

One of the gaps identified is the lack of a dynamic and adaptive approach that can be adapted to system needs in collecting data from the web. Several studies have attempted to address this issue, but there is still room for further development. Therefore, this research proposes an approach that can not only handle data collection automatically but also provides high reliability and scalability for integration in various systems.

In order to overcome the gaps that have been identified, this proposed method is designed by integrating the latest technologies in Big Data processing and scraping pipelines. The proposed method is based on a deep understanding of the main challenges faced by current pipeline scraping methods, the results of which can be integrated with the latest technologies such as machine learning to increase adaptability.

2.1. Research Method

This research was conducted in the ETL (Extract, Transform, Load) method using Apache Airflow. Testing the ETL method is an important and vital phase during data warehouse testing, because this phase affects the quality of the data [11]. Starting from creating a batch processing system with the need to collect external data from services, or from APIs and databases. However, this research will focus on retrieving data with the web scraping method that runs on Apache Airflow. Web scraping is the process of pulling data from website pages to collect information from web pages and store it in a format that is easier to use for further analysis [12]. Then the data will be forwarded to the data warehouse so that the data can be output in the form of .csv files so that it can be used in the Machine Learning and MLOps processes. The complete scraping steps in this research are shown in Figure 1.

2.2. Creation of big data pipeline infrastructure

Big Data Pipeline refers to a series of processes that collect, store, process and analyze data on a large scale (Big Data) to support decision making and other applications [13]. This pipeline will process data from various and diverse sources and present it in an easy-to-understand way for efficient decision making in supporting the sustainability of the system for collecting and analyzing structured and unstructured data through analytical dashboards and machine learning programs.

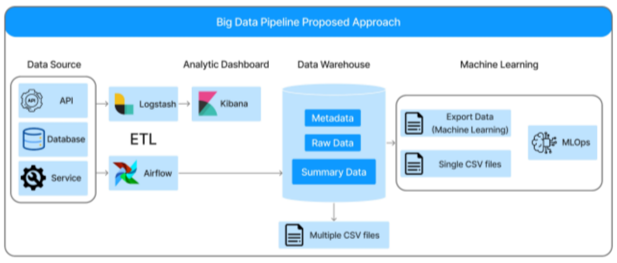


Figure 1. Big Data Pipeline Proposed Approach

2.3. Creating a scraping pipeline program to complete external data

This scraping programme serves to retrieve data from the website of the Market Monitoring System and Basic Needs of the Ministry of Trade of the Republic of Indonesia (https://sp2kp.kemendag.go.id/), in Figure 1 is simulated as Scraping A and Business news portal website (https://bisnis.com/) as Scraping B, automatically. Apache Airflow is an open-source workflow management orchestration tool used to organise and monitor data workflows using a built-in web interface [14]. At this stage, the system is created to optimize the performance of the scraping program so that it carries out scheduled program execution automatically using Apache Airflow.

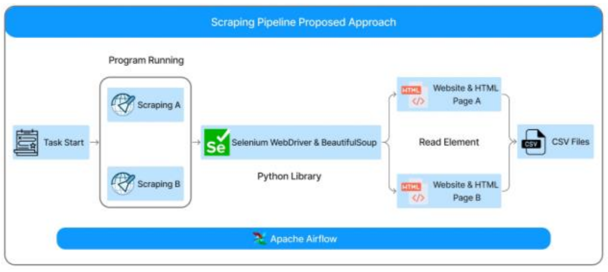


Figure 2. Scraping Pipeline Proposed Approach

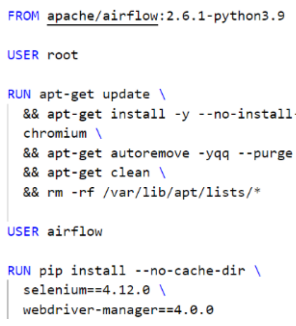
When the scraping pipeline is operated, Scraping A and Scraping B will run a program to read website elements from page A and Page B. Selenium is a tool used to carry out testing on web-based applications [15]. Using a combination of Selenium and BeautifulSoup libraries is the best approach because it can process program execution faster [16]. The creation of a scraping pipeline is not just about creating a smooth flow of data extraction, but also opens up opportunities to gain deep insights from the acquired data. The differences in each function of the scraping tools used in this research can be seen in Table 1.

Table 1. Comparison between Python Web scraping libraries and frameworks[17]

Factors	BeautifulSoup	Selenium
Extensibility	Suitable for low-level complex projects	Best for projects dealing with Core JavaScript
Performance	Pretty slow compared to other libraries while performing a certain task	Can handle up to some level
Ecosystem	It has a lot of dependencies on the ecosystem	It has a good ecosystem for the development

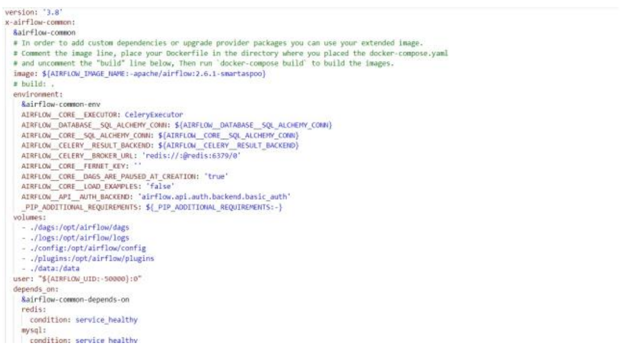
2.4. Docker Compose and Dockerfile Configuration

Docker is an open-source platform that allows developers to package applications and their dependencies into containers that can run anywhere [18]. In order for Apache Airflow and the tools in this research to run on top of the Docker environment for data extraction tasks, it is necessary to configure Docker Compose and Dockerfile. All configurations are made including the environment, service, network, library and volume settings needed, created with Dockerfile in Figure 3 and Docker Compose in Figure 4.



```
FROM apache/airflow:2.6.1-python3.9

USER root

RUN apt-get update \
  && apt-get install -y --no-install-recommends \
  chromium \
  && apt-get autoremove -yqq --purge \
  && apt-get clean \
  && rm -rf /var/lib/apt/lists/*

USER airflow

RUN pip install --no-cache-dir \
  selenium==4.12.0 \
  webdriver-manager==4.0.0
```

Figure 3. Dockerfile Configuration

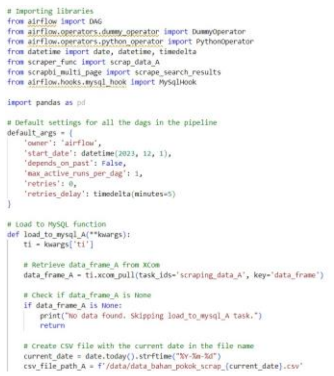


```
version: '3.8'
x-airflow-common:
  &airflow-common
  # In order to add custom dependencies or upgrade provider packages you can use your extended image.
  # Comment the image line, place your Dockerfile in the directory where you placed the docker-compose.yaml
  # and uncomment the "build" line below, Then run `docker-compose build` to build the images.
  image: ${AIRFLOW_IMAGE_NAME:-apache/airflow:2.6.1-smartaspoo}
  # build: .
  environment:
    &airflow-common-env
    AIRFLOW__CORE__EXECUTOR: CeleryExecutor
    AIRFLOW__DATABASE__SQL_ALCHEMY_CONN: ${AIRFLOW__DATABASE__SQL_ALCHEMY_CONN}
    AIRFLOW__CORE__SQL_ALCHEMY_CONN: ${AIRFLOW__CORE__SQL_ALCHEMY_CONN}
    AIRFLOW__CELERY__RESULT_BACKEND: ${AIRFLOW__CELERY__RESULT_BACKEND}
    AIRFLOW__CELERY__BROKER_URL: 'redis://:@redis:6379/0'
    AIRFLOW__CORE__FERNET_KEY: ''
    AIRFLOW__CORE__DAGS_ARE_PAUSED_AT_CREATION: 'true'
    AIRFLOW__CORE__LOAD_EXAMPLES: 'false'
    AIRFLOW__API__AUTH_BACKEND: 'airflow.api.auth.backend.basic_auth'
    _PIP_ADDITIONAL_REQUIREMENTS: ${_PIP_ADDITIONAL_REQUIREMENTS:-}
  volumes:
    - ./dags:/opt/airflow/dags
    - ./logs:/opt/airflow/logs
    - ./config:/opt/airflow/config
    - ./plugins:/opt/airflow/plugins
    - ./data:/data
  user: "${AIRFLOW_UID:-50000}:0"
  depends_on:
    &airflow-common-depends-on
    redis:
      condition: service_healthy
    mysql:
      condition: service_healthy
```

Figure 4. File docker-compose.yml Configuration

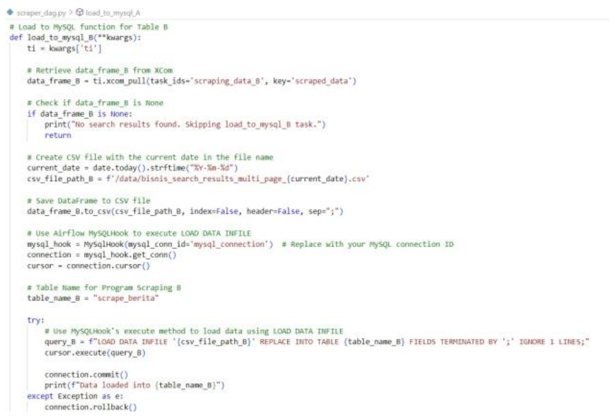
2.5. Creation of a DAG file

Directed Acyclic Graph (DAG) describes the relationship between tasks and defines the order in which they are performed [19]. DAG files are operated from a Python script placed in the specified directory. In addition, the graph is also acyclic, meaning that there are no cycles or tasks that result in the running task returning to the previous task, The Scraping A and Scraping B pipelines in Figure 2 will be imported and assembled into a DAG file that will be run using Apache Airflow according to the schedule declared in Figure 5 in the schedule_interval='@daily' code section. This means that the scraping pipeline program will run automatically every day, and 'retries_delay': timedelta(minutes=5) is also added to give a pause for retrying if there is an interruption in program execution. After building the schedule, the next step is to assemble the scraping A and scraping B programs so that they become a task pipeline that runs according to the dependencies between tasks as in Figure 6. Thus, Apache Airflow and the concept of DAG preparation are an important foundation in managing complex workflows and ensuring the successful implementation of the tasks contained in them. Figure 7 and Figure 8 contain the respective scraping programmes that will be called upon to function in the DAG programme to execute the scraping pipeline proposed approach.



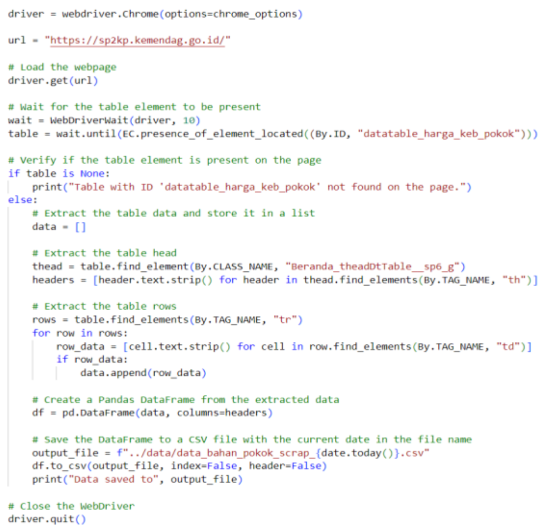
```python
# Importing libraries
from airflow import DAG
from airflow.operators.dummy_operator import DummyOperator
from airflow.operators.python_operator import PythonOperator
from datetime import date, datetime, timedelta
from scraper_func import scrap_data_A
from scrapbi_multi_page import scrape_search_results
from airflow.hooks.mysql_hook import MySqlHook

import pandas as pd

# Default settings for all the dags in the pipeline
default_args = {
    'owner': 'airflow',
    'start_date': datetime(2023, 12, 1),
    'depends_on_past': False,
    'max_active_runs_per_dag': 1,
    'retries': 0,
    'retries_delay': timedelta(minutes=5)
}

# Load to MySQL function
def load_to_mysql_A(**kwargs):
    ti = kwargs['ti']

    # Retrieve data_frame_A from XCom
    data_frame_A = ti.xcom_pull(task_ids='scraping_data_A', key='data_frame')

    # Check if data_frame_A is None
    if data_frame_A is None:
        print("No data found. Skipping load_to_mysql_A task.")
        return

    # Create CSV file with the current date in the file name
    current_date = date.today().strftime("%Y-%m-%d")
    csv_file_path_A = f'/data/data_bahan_pokok_scrap_{current_date}.csv'
```

Figure 5. DAG File Composition



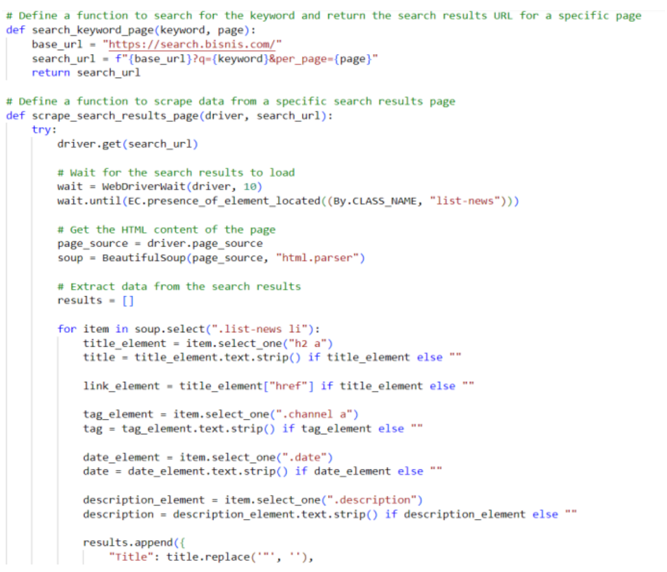
```python
# Load to MySQL function for Table B
def load_to_mysql_B(**kwargs):
    ti = kwargs['ti']

    # Retrieve data_frame_B from XCom
    data_frame_B = ti.xcom_pull(task_ids='scraping_data_B', key='scraped_data')

    # Check if data_frame_B is None
    if data_frame_B is None:
        print("No search results found. Skipping load_to_mysql_B task.")
        return

    # Create CSV file with the current date in the file name
    current_date = date.today().strftime("%Y-%m-%d")
    csv_file_path_B = f'/data/bisnis_search_results_multi_page_{current_date}.csv'

    # Save DataFrame to CSV file
    data_frame_B.to_csv(csv_file_path_B, index=False, header=False, sep=";")

    # Use Airflow MySQLHook to execute LOAD DATA INFILE
    mysql_hook = MySqlHook(mysql_conn_id='mysql_connection')  # Replace with your MySQL connection ID
    connection = mysql_hook.get_conn()
    cursor = connection.cursor()

    # Table Name for Program Scraping B
    table_name_B = "scrape_berita"

    try:
        # Use MySQLHook's execute method to load data using LOAD DATA INFILE
        query_B = f"LOAD DATA INFILE '{csv_file_path_B}' REPLACE INTO TABLE {table_name_B} FIELDS TERMINATED BY ';' IGNORE 1 LINES;"
        cursor.execute(query_B)

        connection.commit()
        print(f"Data loaded into {table_name_B}")
    except Exception as e:
        connection.rollback()
```

Figure 6. Main Function DAG File



```python
driver = webdriver.Chrome(options=chrome_options)

url = "https://sp2kp.kemendag.go.id/"

# Load the webpage
driver.get(url)

# Wait for the table element to be present
wait = WebDriverWait(driver, 10)
table = wait.until(EC.presence_of_element_located((By.ID, "datatable_harga_keb_pokok")))

# Verify if the table element is present on the page
if table is None:
    print("Table with ID 'datatable_harga_keb_pokok' not found on the page.")
else:
    # Extract the table data and store it in a list
    data = []

    # Extract the table head
    thead = table.find_element(By.CLASS_NAME, "Beranda_theadDtTable__sp6_g")
    headers = [header.text.strip() for header in thead.find_elements(By.TAG_NAME, "th")]

    # Extract the table rows
    rows = table.find_elements(By.TAG_NAME, "tr")
    for row in rows:
        row_data = [cell.text.strip() for cell in row.find_elements(By.TAG_NAME, "td")]
        if row_data:
            data.append(row_data)

    # Create a Pandas DataFrame from the extracted data
    df = pd.DataFrame(data, columns=headers)

    # Save the DataFrame to a CSV file with the current date in the file name
    output_file = f"../data/data_bahan_pokok_scrap_{date.today()}.csv"
    df.to_csv(output_file, index=False, header=False)
    print("Data saved to", output_file)

# Close the WebDriver
driver.quit()
```

Figure 7. Scraping A for Market Monitoring System



```python
# Define a function to search for the keyword and return the search results URL for a specific page
def search_keyword_page(keyword, page):
    base_url = "https://search.bisnis.com/"
    search_url = f"{base_url}?q={keyword}&per_page={page}"
    return search_url

# Define a function to scrape data from a specific search results page
def scrape_search_results_page(driver, search_url):
    try:
        driver.get(search_url)

        # Wait for the search results to load
        wait = WebDriverWait(driver, 10)
        wait.until(EC.presence_of_element_located((By.CLASS_NAME, "list-news")))

        # Get the HTML content of the page
        page_source = driver.page_source
        soup = BeautifulSoup(page_source, "html.parser")

        # Extract data from the search results
        results = []

        for item in soup.select(".list-news li"):
            title_element = item.select_one("h2 a")
            title = title_element.text.strip() if title_element else ""

            link_element = title_element["href"] if title_element else ""

            tag_element = item.select_one(".channel a")
            tag = tag_element.text.strip() if tag_element else ""

            date_element = item.select_one(".date")
            date = date_element.text.strip() if date_element else ""

            description_element = item.select_one(".description")
            description = description_element.text.strip() if description_element else ""

            results.append({
                "Title": title.replace('"', ''),
```

Figure 8. Scraping B for News Scraping for Specific Topics

3. Result and Discussion

From creating docker image components, scraping programs, to being assembled into a DAGs scraping pipeline. This research succeeded in collecting data in a structured format and can be analyzed. Scraping programmes capture web page content and store the data in a customised format or on a specialised database [20].

3.1. Scraping Result

Overall, the price comparison table data of basic necessities extracted on the website https://sp2kp.kemendag.go.id/ totalling 5 columns and 16 rows, while on the website https://bisnis.com/ consists of 5 columns and numbers from tens to hundreds of rows of data, following the availability of data from the website.

By using scraping pipeline program, we can extract the entire target table like the results in Table 2, while for news it is also successful in extracting data with the topic "MSMEs" or can be added or even replaced with other topics as needed, such as keywords that have been declared in the DAG program. Apart from that, here we can also limit the amount of news that will be extracted from the keywords entered on the website. So the news scraping results appear as in Table 3.

Table 2. Sample dataset scraping comparison table of prices of basic necessities

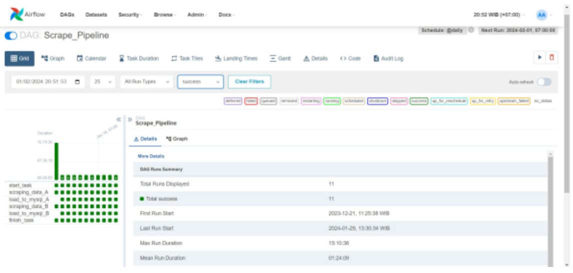
Commodities	Unit	Price Yesterday (26-09-2023)	Price Today (27-09-2023)	Fluctuating Percentage (%)
Medium Rice	Kg	13.300	13.300	0
Premium Rice	Kg	15.000	15.000	0
Sugar	Kg	15.200	15.200	0
Premium Packaged Cooking Oil	Lt	20.700	20.700	0
Bulk Cooking Oil	Lt	14.500	14.500	0
"MINYAKITA" Cooking Oil	Lt	15.100	15.100	0
Beef Hamstrings	Kg	137.400	137.400	0
Purebred Chicken Meat	Kg	35.800	35.800	0
Purebred Chicken Eggs	Kg	30.000	29.900	-0.33
Flour	Kg	13.300	13.300	0
Imported Soybeans	Kg	15.100	15.100	0
Curly Red Chilies	Kg	40.700	41.400	1.72
Red Cayenne Pepper	Kg	40.400	41.300	2.23
Large Red Chili	Kg	40.100	40.100	0
Red Onion	Kg	25.000	25.200	0.8
Honan Garlic	Kg	37.500	37.400	-0.27

Table 3. Sample news website scraping dataset

Title	Link	Tag	Date	Description
Coordinating Minister for SMEs is happy that Persib is collaborating with 2 local fashion brands to develop together	https://bandung.bisnis.com/read/20230926/549/1698840/menkop-ukm-senang-persib-gandeng-2-jenama-fesyen-lokal-berkembang-bersama	West Java News	26 September 2023 22:10	Minister of Cooperatives and Small and Medium Enterprises Teten Masduki welcomed and supported the collaboration program initiated by Persib with MSME players.
PIP Distributes IDR 1.85 Trillion Financing in Bali and Nusa Tenggara	https://ekonomi.bisnis.com/read/20230926/9/1698797/pip-salurkan-pembiayaan-rp185-triliun-di-bali-dan-nusa-tenggara	Economy	26 September 2023 21:03	The Government Investment Center (PIP) has distributed IDR 1.85 trillion in financing to Ultra-micro or Umi business actors in the Bali and Nusa Tenggara regions
MSME Association: TikTok Shop is used more by influencers	https://ekonomi.bisnis.com/read/20230926/12/1698747/asosiasi-umkm-tiktok-shop-lebih-banyak-dipakai-influencer	Trade	26 September 2023 20:56	The MSME Association believes that TikTok Shop is more widely used by artists and influencers
Social Commerce Regulations Are Said to Make MSMEs Breathe More Easily	https://teknologi.bisnis.com/read/20230926/266/1698777/regulasi-social-commerce-disebut-bikin-umkm-bernafas-lebih-lega	Startup	26 September 2023 16:58	It is believed that regulations that separate social media and e-commerce will allow MSMEs to breathe more easily
TikTok Prohibited from Selling, MSME Association: Look for Other E-Commerce	https://ekonomi.bisnis.com/read/20230926/12/1698667/tiktok-dilarang-jualan-asosiasi-umkm-cari-e-commerce-lain	Trade	26 September 2023 15:15	The Indonesian MSME Association assesses that the government's policy of prohibiting TikTok sales is correct

3.2. Apache Airflow Monitoring Interface

This interface allows users to monitor the performance of each DAG and the status of Apache Airflow. More powerful than traditional WMS (Workflow Management System) cron-based implementations, Airflow scheduler allows multiple tasks to run at specific times or intervals, while monitoring them [21]. Figure 9 below is a display of the dashboard in the DAGs scraping pipeline program which contains detailed information on the pipeline design being built to help users in comprehending the program's operations and identifying potential issues during the pipeline program.


DAG: Scrape_Pipeline

Schedule: @daily | Next Run: 2024-02-01, 07:00:00

Scrape_Pipeline

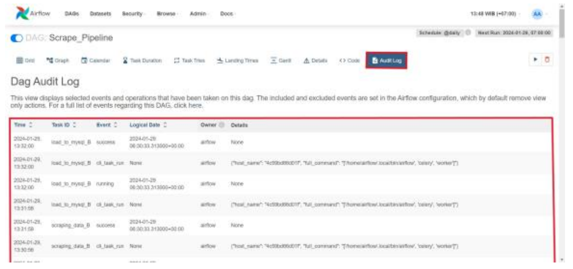
More Details

DAG Runs Summary	
Total Runs Displayed	11
🟩 Total success	11
First Run Start	2023-12-21, 11:25:38 WIB
Last Run Start	2024-01-29, 13:30:34 WIB
Max Run Duration	15:10:36
Mean Run Duration	01:24:09

Figure 9. DAGs Dashboard Monitoring Interface

3.3. Measuring the Scraping Result

To ensure that the program that has been created is running properly, the program implementation process can be monitored using the Audit Log feature, or if you want to know in more detail the process of each program. In the bar on the left, there is a series of tasks, each box representing a program/operator written in the DAGs in red in Figure 9 and Figure 10.


DAG: Scrape_Pipeline

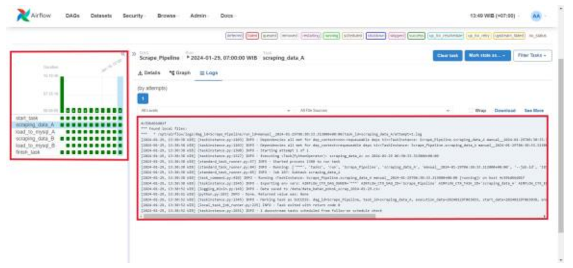
Schedule: @daily | Next Run: 2024-01-29, 07:00:00

Dag Audit Log

This view displays selected events and operations that have been taken on this dag. The included and excluded events are set in the Airflow configuration, which by default remove view only actions. For a full list of events regarding this DAG, click here.

Time	Task ID	Event	Logical Date	Owner	Details
2024-01-29, 13:32:00	load_to_mysql_B	success	2024-01-29 06:30:33.313000+00:00	airflow	None
2024-01-29, 13:32:00	load_to_mysql_B	cli_task_run	None	airflow	{"host_name": "4c59bd66d01f", "full_command": "['/home/airflow/.local/bin/airflow', 'celery', 'worker']"}
2024-01-29, 13:32:00	load_to_mysql_B	running	2024-01-29 06:30:33.313000+00:00	airflow	None
2024-01-29, 13:31:58	load_to_mysql_B	cli_task_run	None	airflow	{"host_name": "4c59bd66d01f", "full_command": "['/home/airflow/.local/bin/airflow', 'celery', 'worker']"}
2024-01-29, 13:31:58	scraping_data_B	success	2024-01-29 06:30:33.313000+00:00	airflow	None
2024-01-29, 13:30:56	scraping_data_B	cli_task_run	None	airflow	{"host_name": "4c59bd66d01f", "full_command": "['/home/airflow/.local/bin/airflow', 'celery', 'worker']"}

Figure 10. Audit Log for DAGs Features


Scrape_Pipeline ▸ 2024-01-29, 07:00:00 WIB scraping_data_A

```
[illegible]
```

Figure 11. Monitoring Task and Checking Logs

If the scraping pipeline program runs successfully, the bar on each task will show green. If it fails, you can check the color on the bar with the following details:

1) Deferred (violet): The task in this program is postponed and has not been scheduled for execution.
2) Failed (red): The task being executed experienced an error or failure.
3) Queued (dark grey): The task has been scheduled for execution and is being processed.
4) Removed (light gray): The task is removed from the queue and is not executed.
5) Restarting (purple): The task is in the process of re-executing.
6) Running (light green): The task is currently executing.
7) Scheduled (brown): The task has been scheduled for execution and is waiting for the specified execution time.
8) Shutdown (dark blue): The task was stopped manually or for some reason.
9) Skipped (pink): Task is skipped during the execution process.
10) Success (dark green): Task completed successfully without errors.
11) Up For Reschedule (aqua): Tasks can be rescheduled for execution.
12) Up For Retry (yellow): Task can be re-executed after experiencing a previous failure.
13) Upstream Failed (orange): The task cannot be executed because one or more previous tasks failed.
14) No_Status (empty): Task has no clear status or cannot be determined.

Apart from that, we can also check the logs of failed tasks to find out further error details. So, if the program in a series of pipelines runs successfully, it will produce data like the sample scraping results from each data source shown in Table 2 Sample dataset scraping comparison table of prices of basic necessities and Table 3 Sample news website scraping dataset. This is different from conventional scraping programs. We cannot monitor the running program process, cannot find details of where the error is in the program, and require manual steps to run it, even though it is assisted by a cron job, it still does not have the features mentioned previously. By building a data scraping program that runs on Apache Airflow, we can combine various existing data sources to build a dataset that runs automatically every day.

4. Conclusion

One of the important findings in this research is to develop a Big Data Pipeline in the data extraction process from various data sources using a combination of Selenium and BeautifulSoup algorithms that complement each other for the data needs in this research. Especially in the scraping process so that the data obtained can later be used for analysis and machine learning process needs so that later the collection of data sources that are run in the pipeline can be processed to produce insights that can help MSMEs in making decisions about their business processes. On this occasion, Apache Airflow, an Orchestration Tool that functions as a WMS (Workflow Management System), is able to create data pipelines using the ETL method in a Big Data Pipeline environment.

5. Acknowledgements

We sincerely thank the Directorate General of Higher Education, Research, and Technology, Ministry of

Education, Culture, Research, and Technology, Republic of Indonesia, for partially funding this project through the Kedaireka Program. This work is also supported by Dian Nuswantoro University (UDINUS) through the Center of Excellence in Science and Technology, UDINUS, and the Asosiasi Pengusaha Oleh-Oleh Jawa Tengah Makanan dan Minuman. Grant contract number: Implementation of Supply Chain Management System for SMEs in the Food and Beverage Sector, Case Study in Central Java No: 4501/F.9.02/UDN-01/IV/2023.

References

[1]. V. T. Ragoobur, B. Seetanah, Z. K. Jaffur, and V. Mooneeram-Chadee, "Building recovery and resilience of Mauritian MSMEs in the midst of the COVID-19 pandemic," *Sci. Afr.*, vol. 20, p. e01651, Jul. 2023, doi: 10.1016/j.sciaf.2023.e01651.

[2]. M. Tuba, S. Akashe, and A. Joshi, Eds., *ICT Systems and Sustainability: Proceedings of ICT4SD 2020, Volume 1*, vol. 1270. in Advances in Intelligent Systems and Computing, vol. 1270. Singapore: Springer Singapore, 2021. doi: 10.1007/978-981-15-8289-9.

[3]. S. Kamboj and S. Rana, "Big data-driven supply chain and performance: a resource-based view," *TQM J.*, vol. 35, no. 1, pp. 5–23, Jan. 2023, doi: 10.1108/TQM-02-2021-0036.

[4]. M. Matskin *et al.*, "A Survey of Big Data Pipeline Orchestration Tools from the Perspective of the DataCloud Project".

[5]. A. Q. Khan *et al.*, "Smart Data Placement Using Storage-as-a-Service Model for Big Data Pipelines," *Sensors*, vol. 23, no. 2, p. 564, Jan. 2023, doi: 10.3390/s23020564.

[6]. R. Hofstetter, "A Step-by-Step Guide for Data Scraping," in *The Machine Age of Customer Insight*, M. Einhorn, M. Löffler, E. De Bellis, A. Herrmann, and P. Burghartz, Eds., Emerald Publishing Limited, 2021, pp. 129–143. doi: 10.1108/978-1-83909-694-520211013.

[7]. Y. Kryvenchuk, M. Burak, and Lviv Polytechnic National University, "COMPARATIVE ANALYSIS OF SELENIUM AND BEAUTIFULSOUP EFFICIENCY," *Her. Khmelnytskyi Natl. Univ.*, vol. 305, no. 1, pp. 50–52, Feb. 2022, doi: 10.31891/2307-5732-2022-305-1-50-52.

[8]. R. Bhargava, R. Lobo, R. Shah, N. Shah, and S. Nair, "Easier Web Navigation Using Intent Classification, Web Scraping and NLP Approaches," in *2022 5th International Conference on Advances in Science and Technology (ICAST)*, Mumbai, India: IEEE, Dec. 2022, pp. 286–290. doi: 10.1109/ICAST55766.2022.10039559.

[9]. A. Suleykin and P. Panfilov, "Implementing Big Data Processing Workflows Using Open Source Technologies," in *DAAAM Proceedings*, 1st ed., vol. 1, B. Katalinic, Ed., DAAAM International Vienna, 2019, pp. 0394–0404. doi: 10.2507/30th.daaam.proceedings.054.

[10]. A. Mishra, S. Mishra, and N. S. Kumar, "Data Analysis using Robot Process Automation Study on Web Scraping using UI Path Studio," in *2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India: IEEE, Dec. 2022, pp. 2221–2225. doi: 10.1109/ICAC3N56670.2022.10074502.

[11]. A. Yudhistira, I. S. Sitanggang, and H. A. Adrianto, "Development of ETL (Extract, Transform and Load) Module in Indonesian Agricultural Commodities OLAP System," vol. 15, no. 2, 2023.

[12]. A. Mistry, "NCRD's Technical Review : e-Journal ISSN: 2455-166X Volume 7, Issue 1 (Jan-Dec 2022)," vol. 7, no. 1, 2022.

[13]. A. Issac, A. Ebrahimi, J. Mohammadpour Velni, and G. Rains, "Development and deployment of a big data pipeline for field-based high-throughput cotton phenotyping data," *Smart Agric. Technol.*, vol. 5, p. 100265, Oct. 2023, doi: 10.1016/j.atech.2023.100265.

[14]. B. Harenslak, "Data Pipelines with Apache Airflow".

[15]. "Design and Visualization of Python Web Scraping Based on Third-Party Libraries and Selenium Tools," *Acad. J. Comput. Inf. Sci.*, vol. 6, no. 9, 2023, doi: 10.25236/AJCIS.2023.060904.

[16]. S. V. Oprea and A. Bâra, "Why Is More Efficient to Combine BeautifulSoup and Selenium in Scraping For Data Under Energy Crisis," no. 2.

[17]. LaboNFC, University of Quebec at Chicoutimi, 555 Boulevard de l'Université, Saguenay (QC), Canada, C. Lotfi, S. Srinivasan, M. Ertz, and I. Latrous, "Web Scraping Techniques and Applications: A Literature Review," in *SCRS CONFERENCE PROCEEDINGS ON INTELLIGENT SYSTEMS*, Soft Computing Research Society, 2021, pp. 381–394. doi: 10.52458/978-93-91842-08-6-38.

[18]. D. Reis, B. Piedade, F. F. Correia, J. P. Dias, and A. Aguiar, "Developing Docker and Docker-Compose Specifications: A Developers' Survey," *IEEE Access*, vol. 10, pp. 2318–2329, 2022, doi: 10.1109/ACCESS.2021.3137671.

[19]. M. Kotliar, A. V. Kartashov, and A. Barski, "CWL-Airflow: a lightweight pipeline manager supporting Common Workflow Language," *GigaScience*, vol. 8, no. 7, p. giz084, Jul. 2019, doi: 10.1093/gigascience/giz084.

[20]. S. Ashouri *et al.*, "Indicators on firm level innovation activities from web scraped data," *Data Brief*, vol. 42, p. 108246, Jun. 2022, doi: 10.1016/j.dib.2022.108246.

[21]. R. Mitchell *et al.*, "Exploration of Workflow Management Systems Emerging Features from Users Perspectives," in *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA: IEEE, Dec. 2019, pp. 4537–4544. doi: 10.1109/BigData47090.2019.9005494.