

MODEL PEMBELAJARAN DAN LAPORAN AKHIR
PROJECT-BASED LEARNING
MATA KULIAH BIG DATA
KELAS C



**“Membangun ETL Pipeline Toko Elektronik Samsung pada
Marketplace Shopee Menggunakan Python dan Airflow”**

DISUSUN OLEH KELOMPOK “V” :

- | | |
|----------------------------|---------------------------|
| 1. ALVIN RYAN DANA | (21083010035) - KETUA |
| 2. ELLEXIA LEONIE GUNAWAN | (21083010027) - ANGGOTA |
| 3. DIANA SINTHYA PUTRI | (21083010090) - ANGGOTA |
| 4. YAYANG DIMAS SAPUTRA | (21083010102) - ANGGOTA |
| 5. MUIZZADIN | (21083010116) - ANGGOTA |
| 6. DINA MAGDALENA MANURUNG | (21083010117) - ANGGOTA |

DOSEN PENGAMPU:

TRESNA MAULANA FAHRUDIN, S.ST., MT (199305012022031007)

PROGRAM STUDI SAINS DATA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”
JAWA TIMUR
2023

DAFTAR ISI

MODEL PEMBELAJARAN.....	II
DAFTAR ISI	IV
DAFTAR GAMBAR.....	V
DAFTAR TABEL	VI
1. 1	
1.1 LATAR BELAKANG	1
1.2 PERMASALAHAN	3
1.3 TUJUAN	3
1.4 MANFAAT	3
2. 44	
2.1 TEORI PENUNJANG	4
2.2 PENELITIAN TERKAIT	7
3. 9	
3.1 TOKO SAMSUNG DI SHOPEE	9
3.2 WEB SCRAPING	9
3.3 PROSES EKSTRAKSI DAN TRANSFORMASI	9
3.4 DATA PIPELINE USING APACHE AIRFLOW	10
3.5 LOAD TO POSTGRESQL	10
4. 1111	
4.1 WEB SCRAPING	11
4.2 DATA PIPELINE USING APACHE AIRFLOW	15
4.3 LOAD TO POSTGRESQL	23
5. 2626	
6. 2727	
7. 2929	

DAFTAR GAMBAR

Gambar 1. Logo Shopee.....	4
Gambar 2. Logo Python.....	4
Gambar 3. Otomatisasi.....	5
Gambar 4. Logo Airflow.....	6
Gambar 5. Logo Postgresql.....	7
Gambar 6. Desain sistem yang diusulkan dalam proyek.....	8
Gambar 7. Instalasi docker.....	15
Gambar 8. Membuat direktori untuk program.....	16
Gambar 9. Inisialisasi lingkungan Airflow.....	21
Gambar 10. Membuat dan menjalankan kontainer.....	21
Gambar 11. Tampilan docker	22
Gambar 12. Koneksi di Airflow.....	22
Gambar 13. Tampilan DAG.....	23
Gambar 14. Menjalankan tugas di DAG.....	23
Gambar 15. Masuk Ke Airflow	23
Gambar 16. Membuat Database.....	24
Gambar 17. Melihat isi file database.....	24
Gambar 18. Menampilkan isi database Shopee.....	25

1. BAB I: PENDAHULUAN

1.1 Latar Belakang

Samsung adalah perusahaan besar asal Korea Selatan yang bergerak di industri elektronik. Perusahaan ini mampu bersaing dengan perusahaan global lainnya seperti Apple yang juga berfokus pada bidang elektronik. Menurut Interbrand, sebuah konsultan merek global, Samsung masuk ke dalam lima besar merek terbaik di Best Global Brands 2022 dengan nilai sekitar USD 87,7 miliar. Nilai ini mengalami peningkatan sebesar 17% dibandingkan dengan tahun 2021 yang sebesar USD 74,6 miliar. Dari banyak produk yang ditawarkan, *smartphone* menjadi produk utama Samsung yang memiliki penjualan tertinggi dibandingkan dengan produk lainnya. Menurut situs International Data Corporation (IDC), Samsung menduduki posisi nomor 1 sebagai merek ponsel terbaik di dunia. Pada kuartal pertama tahun 2023, Samsung berhasil meraih pangsa pasar sebesar 22,5%. Posisi Samsung berhasil naik ke puncak jika dibandingkan dengan kuartal keempat tahun 2022.

Dalam era digital saat ini, toko online menjadi salah satu metode yang paling populer bagi perusahaan untuk menjual produk mereka. Berdasarkan penelitian yang dilakukan (Alvin Edgar, dkk., 2021) tercatat adanya lonjakan yang signifikan dalam pembelian produk melalui platform *e-commerce* pada kuartal kedua tahun 2020, yakni sebesar 38%. Saat ini sudah banyak platform *e-commerce* di Indonesia. Shopee, sebagai salah satu platform jual beli online terbesar di Asia Tenggara, menawarkan layanan yang luas bagi penjual untuk memasarkan produk mereka kepada jutaan pengguna. Berdasarkan data dari Similarweb pada 1 Juni 2023, Shopee merupakan *marketplace* dengan peringkat pertama untuk kunjungan *website* terbanyak. Oleh karena itu, kami akan mendapat data penjualan produk Samsung dari *marketplace* Shopee. Namun, untuk mengelola dan memanipulasi data yang terkait dengan toko di Shopee, diperlukan proses ETL (*Extract, Transform, Load*) yang efisien.

ETL merupakan singkatan dari Ekstraksi, Transformasi, dan Load yang melibatkan pengambilan data dari berbagai sistem sumber, kemudian dilanjutkan dengan transformasi data sebelum akhirnya data tersebut dimuat ke dalam sistem gudang data yaitu *database*. Memilih *database* merupakan bagian

yang sangat penting karena *database* harus mampu menangani data dalam jumlah yang sangat besar agar database tetap berjalan. PostgreSQL merupakan salah satu sistem manajemen basis data relasional yang populer dan kuat karena efisiensi waktu yang cepat dalam memproses query dibanding DBMS yang lain. Data yang sebelumnya tercampur diolah menjadi satu set data yang konsisten yang disimpan dalam PostgreSQL, sehingga memudahkan analisis di masa mendatang saat diperlukan. Dalam konteks toko online Shopee, ETL Pipeline dapat digunakan untuk mengumpulkan data tentang berapa produk yang terjual dan informasi lainnya, serta melakukan pemrosesan dan pengubahan data agar dapat dimanfaatkan untuk analisis lebih lanjut, pemantauan, dan pengambilan keputusan.

Otomatisasi proses ETL merupakan langkah penting dalam mengelola dan menganalisis data karena membantu dalam menghemat waktu dan usaha dengan menjalankan tugas-tugas ETL secara otomatis, menggantikan proses manual yang memakan waktu dan berpotensi menyebabkan kesalahan. Bahasa pemrograman yang digunakan dalam membangun ETL adalah python. Python dapat digunakan untuk mengambil data dari Shopee dengan *scraping* menggunakan BeautifulSoup, melakukan transformasi dan manipulasi data, serta memuatnya ke dalam penyimpanan data yang sesuai. Apache Airflow merupakan platform pengaturan alur kerja yang open-source. Dalam konteks ini, Airflow dapat digunakan untuk mengotomatisasi proses ETL dari hasil *scraping* ke PostgreSQL. Airflow digunakan untuk mendefinisikan alur kerja atau *workflow* yang kompleks dengan tugas-tugas yang saling bergantung, menjadikannya alat yang kuat untuk menjalankan ETL secara otomatis.

Oleh karena itu, pada proyek ini kami ingin membangun ETL ke PostgreSQL dengan Python dan Airflow untuk membantu menghemat waktu dan usaha dalam mengelola data penjualan toko Samsung. Hal ini juga memungkinkan pengulangan proses ETL secara terjadwal dan konsisten, sehingga memperkuat pengolahan data dan analisis data secara efisien.

1.2 Permasalahan

- Bagaimana membangun suatu ETL pipeline toko elektronik Samsung pada *marketplace* Shopee secara terintegrasi?

1.3 Tujuan

- Membangun suatu ETL pipeline toko elektronik Samsung pada *marketplace* Shopee secara terintegrasi menggunakan Python dan Airflow

1.4 Manfaat

1.4.1 Bagi Industri

Proses ETL (Ekstraksi, Transformasi, dan Load) menjadi lebih efisien dan otomatis bagi industri sehingga dapat menghemat waktu dan sumber daya yang berharga, serta mengurangi risiko kesalahan manusia. Selain itu, Python dan Airflow memberikan skalabilitas, dan kemampuan untuk melakukan transformasi data yang kompleks. Maka melalui proyek ini, industri dapat lebih mudah dalam mengelola data penjualan toko Samsung dan melakukan penjadwalan sinkronisasi data yang otomatis agar proses pengolahan data lebih akurat.

1.4.2 Bagi Pengembangan Ilmu

Proyek ini dapat memberikan manfaat yang signifikan bagi pengembangan ilmu seperti memberikan kesempatan untuk belajar dan mengembangkan pemahaman tentang proses ETL, dan penggunaan *tools* seperti Python dan Airflow. Dengan membangun ETL juga membuka peluang eksplorasi dalam pengembangan ilmu dan teknologi informasi secara keseluruhan. Melalui proyek ini, peneliti dapat mengakses data penjualan pada toko Samsung di shopee yang memungkinkan peneliti untuk menganalisis lebih jauh mengenai penjualan toko tersebut.

2. BAB II: TINJAUAN PUSTAKA

2.1 Teori Penunjang

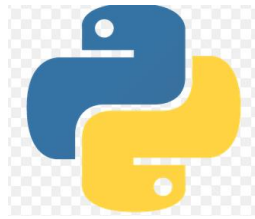
2.1.1 Shopee



Gambar 1. Logo Shopee

Selama pandemi COVID-19, pertumbuhan *e-commerce* di Indonesia terus mengalami peningkatan sebanyak 26%. Di Indonesia sendiri terdapat banyak situs *e-commerce* untuk melakukan pembelian online, dan salah satunya adalah Shopee. Shopee adalah satu dari 10 situs online di Indonesia yang memiliki banyak populasi digital hingga Juni 2017. Pertumbuhan Shopee sebesar 767% dalam semester pertama tahun 2017 menjadi kontribusi terbesar dalam pertumbuhan rata-rata pertumbuhan marketplace. Saat ini, *smartphone* telah menjadi kebutuhan utama bagi banyak orang dari berbagai usia. Alasan utamanya adalah karena *smartphone* memungkinkan pengguna untuk melakukan berbagai aktivitas digital, seperti mengirim pesan, menjadwalkan kegiatan, dan bahkan belajar atau bekerja secara daring. Terutama bagi mereka yang aktif dalam belajar dan bekerja, *smartphone* menjadi alat yang sangat penting karena dapat mendukung kinerja dan kegiatan sehari-hari mereka. Dengan demikian, *smartphone* telah menjadi produk yang sangat diminati dan dicari oleh para pembeli, mengingat pentingnya peran dan fungsi yang dimiliki oleh piranti elektronik ini dalam kehidupan sehari-hari. Oleh karena itu, kami tertarik untuk melakukan *scraping* data dari website shopee khususnya pada toko *smartphone* untuk mengetahui nama produk, harga, stok, dan nilai bintang produk.

2.1.2 Python



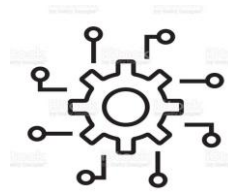
Gambar 2. Logo Python

Python merupakan salah satu bahasa pemrograman yang banyak digunakan oleh perusahaan besar maupun para developer untuk mengembangkan berbagai macam aplikasi berbasis desktop, website, dan mobile. Python menyediakan beberapa pustaka yang berguna untuk melakukan scraping data, seperti BeautifulSoup, Scrapy, atau Selenium. Dengan menggunakan pustaka-pustaka ini, kita dapat mengekstrak data dari situs web atau sumber data lainnya dengan mudah. Python juga dapat digunakan untuk melakukan migrasi data dalam konteks Apache Airflow. Pustaka yang bisa digunakan seperti SQLAlchemy untuk berinteraksi dengan berbagai jenis basis data dan melakukan migrasi data antar sistem. Dalam Apache Airflow, Anda dapat menggunakan PythonOperator untuk menjalankan kode migrasi data sebagai bagian dari aliran kerja yang terjadwal.

2.1.3 ETL Pipeline

ETL atau (Ekstraksi, Transformasi, dan Load) adalah sekumpulan proses untuk mengambil dan memproses data dari satu atau banyak sumber data menjadi sumber data baru. ETL merupakan proses inti dari integrasi data dan biasanya terkait dengan *scraping*. Tools ETL mengekstrak data dari sumber yang dipilih, mengubahnya menjadi format baru sesuai kebutuhan dan memuatnya ke dalam struktur data target. Dengan ETL, data yang diambil dari sumber tertentu dapat dimasukkan ke dalam database.

2.1.4 Otomatisasi



Gambar 3. Otomatisasi

Otomatisasi sudah menjadi sebuah keharusan karena dengan serba otomatis itu banyak sekali yang bisa dicapai, diantaranya efisiensi dan penghematan waktu. Dalam Airflow, *Directed Acyclic Graph* (DAG) digunakan sebagai representasi struktural dari alur kerja yang harus diotomatisasi. DAG adalah representasi visual dari alur kerja yang terdiri dari tugas-tugas yang saling terkait dan harus dieksekusi dalam urutan tertentu. Sedangkan otomatisasi menggunakan alur kerja tersebut untuk menjalankan tugas-tugas secara terjadwal dan terotomatisasi. Dengan kombinasi DAG dan otomatisasi, pengguna dapat meningkatkan efisiensi, konsistensi, dan skalabilitas dalam proses pengelolaan alur kerja.

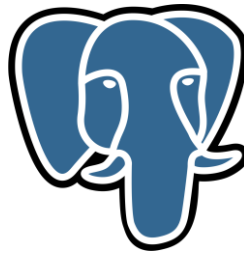
2.1.5 Airflow



Gambar 4. Logo Airflow

Pesatnya perkembangan ilmu data membuat perangkat pipeline/alur kerja manager semakin banyak digunakan. Apache Airflow adalah aplikasi Workflow Management System (WMS) berbasis tugas yang ditulis menggunakan bahasa Python. Apache Airflow digunakan untuk melakukan migrasi data dari web ke database, membuat laporan harian, menggabungkan beberapa tabel, menghitung beberapa metrik pengukuran, dan menuliskan data ke tempat penyimpanan yang diinginkan.

2.1.6 PostgreSQL



Gambar 5. Logo Postgreesql

PostgreSQL adalah salah satu database yang bersifat open source yang didistribusikan secara bebas sesuai dengan perjanjian lisensi BSD. PostgreSQL merupakan sebuah turunan dari database SQL (*Structured Query Language*). PostgreSQL adalah sebuah sistem database yang memiliki kemampuan sangat bagus dengan dilengkapi fitur-fitur yang mendukung untuk database aplikasi dengan skala besar. Fitur-fitur yang disediakan oleh PostgreSQL antara lain Triggers, DB Mirror, PGPool, Slony, PGCluster, dan lain-lain (Rahardja, 2022).

2.2 Penelitian Terkait

2.2.1 Membangun Spotify ETL menggunakan Python dan Airflow (Desember, 2022)

Jurnal ini membahas tentang proyek yang cocok bagi pemula yang ingin memulai dengan membangun saluran pipa sederhana dan mengotomatiskannya menggunakan aliran udara. Tahap pertama akan berfokus pada pembangunan pipa secara keseluruhan, dan kemudian proyek dapat diperluas dengan mengintegrasikannya dengan sistem Airflow. Ada beberapa batasan dalam proyek ini yang dapat diatasi dengan menggunakan token refresh untuk memperbarui token secara otomatis. Selain itu, kita dapat mengatur aliran udara di layanan cloud agar berjalan setiap hari dan mengambil data sekali sehari untuk mengurangi beban harian.

2.2.2 Akuisisi Data Prediksi Curah hujan secara Periodik Menggunakan Apache Airflow (Mei, 2022)

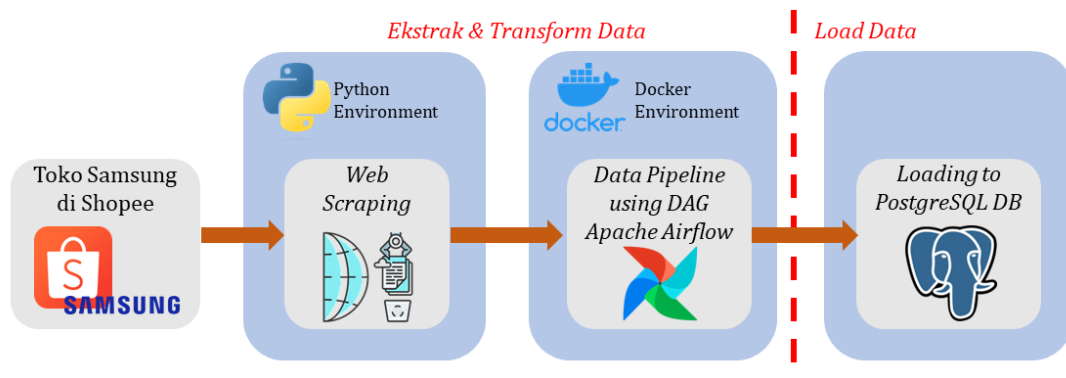
Pada jurnal ini membahas tentang data curah hujan yang diambil dan akan disimpan di komputer lokal menggunakan alat atau aplikasi otomasi yang disebut Apache Airflow. Proses akuisisi data dari server BMKG ke komputer lokal dilakukan secara otomatis dua kali sehari, pada pukul 00.00 dan 12.00. Dalam

Apache Airflow, terdapat dua tugas (task) yang dibuat dalam Directed Acyclic Graph (DAG) untuk proses ini. Tugas pertama berfungsi sebagai sensor untuk memeriksa ketersediaan data, sedangkan tugas kedua bertanggung jawab untuk menjalankan proses utama akuisisi data. Sebagai saran pengembangan selanjutnya, dapat lebih memanfaatkan fitur-fitur yang disediakan oleh Apache Airflow. Salah satu fitur yang bermanfaat adalah kemampuan Apache Airflow untuk memberikan notifikasi kepada pengguna jika ada tugas yang berhasil atau gagal.

2.2.3 Otomatisasi Pembuatan Pipeline di Apache Airflow (Juni, 2020)

Pada jurnal ini membahas solusi UI dari awal hingga akhir pembuatan pipeline hingga pemantauan. Jurnal ini mengusulkan untuk memberikan pengalaman pengguna yang canggih dan memungkinkan pengguna non-teknis untuk memanfaatkan keunggulan alat orkestrasi data. Jurnal ini menggunakan Airflow, alat orkestrasi data sumber terbuka dan modul python bernama "DAG-factory" untuk mengimplementasikan solusi yang diusulkan. Jurnal ini memiliki tujuan akhir yang ingin dicapai yaitu membuat platform bagi setiap pengguna untuk dapat membuat pipa data dengan mengklik tombol atau eksekusi baris perintah dengan argumen yang diteruskan sebagai spesifikasi. Oleh karena itu, jurnal ini berfokus pada pengguna pemrosesan data sebagai pengguna utamanya untuk memberikan pengalaman pengguna yang terbaik.

3. BAB III: METODOLOGI PENELITIAN



Gambar 6. Desain sistem yang diusulkan dalam proyek

3.1 Toko Samsung di Shopee

Langkah pertama dalam proyek ini adalah melakukan pengambilan data secara real-time dari situs *marketplace* Shopee. Pengambilan data dari Shopee adalah proses untuk mendapatkan informasi terkait dengan nama produk, harga, dan informasi lainnya yang tersedia pada situs *marketplace* Shopee. Kami mengambil data secara spesifik yaitu data elektronik dari toko Samsung. Langkah untuk mengambil data adalah mengakses halaman populer dan mengambil informasi produk elektronik pada toko Samsung.

3.2 Web scraping

Proses pengambilan data dilakukan dengan mengakses API (Application Programming Interface) pada situs Shopee, yang kemudian dilakukan metode *web scraping* untuk mengambil informasi yang diinginkan dari halaman web Shopee. Pada proyek ini, kami melakukan *web scraping* menggunakan bahasa pemrograman Python untuk mendapatkan informasi produk elektronik pada toko Samsung di *marketplace* Shopee. Data yang diambil melalui API dapat mencakup seluruh informasi yang tersedia pada halaman web tersebut, sehingga perlu dilakukan pengolahan lebih lanjut agar mendapatkan informasi yang terstruktur.

3.3 Proses Ekstraksi dan Transformasi

Ekstraksi data adalah proses mengambil informasi yang relevan dari sumber data yang ada. Dalam konteks web scraping, ekstraksi data dilakukan dengan menggunakan teknik dan alat yang tepat untuk mengambil informasi yang diinginkan dari halaman web. Pada saat scraping seluruh data yang berada pada

halaman akan ter scraping semua tanpa terkecuali. Oleh karena itu, kami melakukan ekstraksi data berupa pemilihan kolom yang penting dan membuang data yang tidak memuat informasi penting. Hasil ekstraksi data tersebut kami tampilkan dalam bentuk data frame yang terstruktur, yang terdiri dari kolom yang kami pilih yaitu nama produk, harga, stok barang, dan rating bintang.

Setelah proses ekstraksi data, dilakukan proses transformasi. Transformasi adalah langkah di mana data yang telah diambil pada tahap sebelumnya diolah dan diubah menjadi format yang sesuai dengan kebutuhan. Salah satu masalah yang mungkin muncul adalah tipe data yang tidak sesuai. Oleh karena itu, kami akan melakukan transformasi data dengan mengubah jumlah angka di belakang koma atau pembulatan pada kolom `rating_star`. Dengan memperoleh representasi yang lebih sederhana dan mudah dibaca dari suatu angka. Kami berhasil memanfaatkan web scraping untuk mendapatkan informasi produk elektronik dari official store Samsung di Shopee dan siap digunakan dalam proses ETL selanjutnya.

3.4 Data Pipeline using Apache Airflow

Setelah melakukan ekstrak dan transformasi data, maka sekarang akan melanjutkan proses data pipeline dengan Apache Airflow menggunakan Docker Environment. Kami menggunakan Docker karena lebih mudah untuk menginstal dan memelihara selain itu Docker juga independen dari OS. Tujuan Airflow yaitu sebagai pipeline atau penghubung untuk menghubungkan data informasi toko Samsung ke PostgreSQL. Setelah Airflow terinstall saatnya membuat DAG yang diperlukan untuk proyek kami yang merupakan sekumpulan tugas yang ditentukan dalam urutan eksekusi.

3.5 Load to PostgreSQL

Setelah pipeline berhasil dibangun, maka data akan tersimpan dalam basis data PostgreSQL. PostgreSQL menyediakan struktur relasional yang kuat dan dapat diakses menggunakan bahasa SQL. Sehingga dengan menggunakan PostgreSQL, dapat memudahkan pengolahan data berikutnya seperti, membuat tabel, mengatur skema, dan menyimpan data dengan menggunakan perintah-perintah SQL.

4. BAB IV: HASIL DAN PEMBAHASAN

4.1 Web Scraping

- Membuat program untuk ekstrak data

```
import pandas as pd
import numpy as np
import requests

def scrape_produk_samsung():

    url =
'https://shopee.co.id/api/v4/shop/rcmd_items?bundle=s
hop_page_category_tab_main&limit=100&offset=0&shop_id
=52635036&sort_type=1&upstream='
    response = requests.get(url).json()

    names = []
    prices = []
    currencies = []
    stocks = []
    rating_stars = []

    for item in response['data']['items']:
        names.append(item['name'])
        prices.append(item['price'])
        currencies.append(item['currency'])
        stocks.append(item['stock'])

    rating_stars.append(item['item_rating']['rating_star'
])

    data = {
        'name': names,
        'price': prices,
        'currency': currencies,
        'stock': stocks,
        'rating_star': rating_stars
    }
    df = pd.DataFrame(data)

    return df
```

Pada skrip di atas, kami melakukan *scraping* data shopee dengan menggunakan API. Kami ingin mengambil informasi nama produk (*names*), harga (*prices*), stok (*stocks*), dan nilai bintang produk (*rating star*). Informasi-informasi tersebut kemudian akan dibuat menjadi sebuah dataframe dengan menggunakan fungsi 'return df'.

- Membuat program untuk transformasi data

```
def Data_Quality(load_df):
    if load_df.empty:
        print('No Songs Extracted')
        return False

    #Checking for Nulls in our data frame
    if load_df.isnull().values.any():
        raise Exception("Null values found")

def Transform_df(load_df):
    transformed_df = load_df.copy()

    return transformed_df[['name', 'price',
    'currency', 'stock', 'rating_star']]
```

Dari hasil ekstraksi data akan dilakukan transformasi data. Pada tahap ini, dibuat 2 fungsi yaitu:

- def Data_Quality(load_df) : fungsi ini digunakan untuk memeriksa data frame yang kosong, menerapkan kendala unik, memeriksa nilai null. Karena data ini dapat merusak database kami, penting untuk menerapkan pemeriksaan kualitas data ini.
 - def Transform_df(load_df) : fungsi ini digunakan untuk memuat hasil data yang telah diperiksa kualitasnya.
- Menggabungkan file ekstrak dan transformasi menjadi samsung_etl.py

```
import pandas as pd
import numpy as np
import requests

def scrape_produk_samsung():

    url =
    'https://shopee.co.id/api/v4/shop/rcmd_items?bundle=s
hop_page_category_tab_main&limit=100&offset=0&shop_id
=52635036&sort_type=1&upstream='
    response = requests.get(url).json()

    names = []
    prices = []
    currencies = []
    stocks = []
    rating_stars = []
```

```

        for item in response['data']['items']:
            names.append(item['name'])
            prices.append(item['price'])
            currencies.append(item['currency'])
            stocks.append(item['stock'])

rating_stars.append(item['item_rating']['rating_star'
])

    data = {
        'nama': names,
        'harga': prices,
        'currency': currencies,
        'stock': stocks,
        'rating': rating_stars
    }
    df = pd.DataFrame(data)

    return df

def Data_Quality(load_df):
    if load_df.empty:
        print('No Data Extracted')
        return False

    if load_df.isnull().values.any():
        raise Exception("Null values found")

def Transform_df(load_df):
    transformed_df = load_df.copy()
    transformed_df['rating'] =
transformed_df['rating'].round(2)

    return transformed_df[['nama', 'harga',
'currency', 'stock', 'rating']]

def samsung_etl():
    load_df=scrape_produk_samsung()
    Data_Quality(load_df)
    Transformed_df=Transform_df(load_df)
    print(Transformed_df)
    return Transformed_df

samsung_etl()

```

Setelah berhasil membuat program ekstrak dan transformasi, maka kita satukan atau gabung program tersebut dengan nama 'samsung_etl.py'. File ini yang nanti akan digunakan dalam program.

- Membuat file samsung_dag.py

```
import datetime as dt
import pandas as pd
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.hooks.base_hook import BaseHook
from airflow.providers.postgres.hooks.postgres import PostgresHook
from airflow.providers.postgres.operators.postgres import PostgresOperator
from sqlalchemy import create_engine

from airflow.utils.dates import days_ago
from samsung_etl import samsung_etl

default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': dt.datetime(2023,6,11),
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
    'retries': 1,
    'retry_delay': dt.timedelta(minutes=1)
}

dag = DAG(
    'samsung_dag',
    default_args=default_args,
    description='Shopee etl',
    schedule_interval=dt.timedelta(minutes=50),
)

def ETL():
    print("started")
    df= samsung_etl()
    #print(df)
    conn = BaseHook.get_connection('postgre_sql')
    engine =
create_engine(f'postgresql://{conn.login}:{conn.password}@{conn.host}:{conn.port}/{conn.schema}')
    df.to_sql('toko_shopee', engine,
if_exists='replace')

with dag:
    create_table= PostgresOperator(
        task_id='create_table',
```

```

        postgres_conn_id='postgre_sql',
        sql="""
            CREATE TABLE IF NOT EXISTS toko_shopee(
            nama VARCHAR(200),
            harga VARCHAR(200),
            currency VARCHAR(200),
            stock VARCHAR(200),
            rating VARCHAR(200)

            )
        """
    )

    run_etl = PythonOperator(
        task_id='samsung_etl_final',
        python_callable=ETL,
        dag=dag,
    )

    create_table >> run_etl

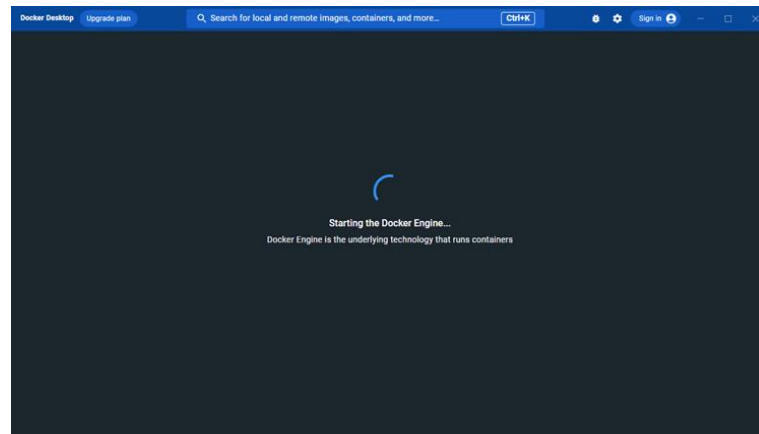
```

- **from airflow.operators.python_operator import PythonOperator:** Kami menggunakan operator Python untuk menjalankan fungsi Python seperti menyisipkan DataFrame ke dalam tabel.
- **from airflow.providers.postgres.operators.postgres import PostgresOperator:** Kami menggunakan operator Postgres untuk membuat tabel di database Postgres kami.
- **from airflow.hooks.base_hook import BaseHook:** Hook adalah abstraksi dari API tertentu yang memungkinkan Airflow berinteraksi dengan sistem eksternal. Hook terintegrasi ke dalam banyak operator, tetapi juga dapat digunakan langsung dalam kode DAG. Kami menggunakan hook di sini untuk menghubungkan Database Postgres dari fungsi Python kami.
- **from samsung_etl import samsung_etl:** Mengimpor fungsi samsung_etl dari file samsung_etl.py.

4.2 Data Pipeline using Apache Airflow

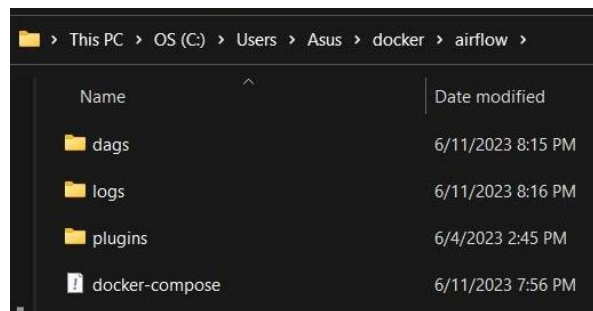
- Instalasi Docker

Source link: <https://www.docker.com/products/docker-desktop/>



Gambar 7. Instalasi docker

- Membuat direktori untuk program



Gambar 8. Membuat direktori untuk program

Struktur direktori:

C:\USERS\ASUS\DOCKER\AIRFLOW

```
| docker-compose.yml
|-----dags
|   | samsung_dag.py
|   | samsung_etl.py
|-----logs
|-----plugins
```

File-file yang telah disiapkan pada tahap-tahap sebelumnya, disusun menjadi seperti skema di atas.

- Membangun Airflow
 - a. Membuat file docker-compose.yaml

```
# Licensed to the Apache Software Foundation (ASF)
under one
# or more contributor license agreements.  See the
NOTICE file
# distributed with this work for additional
information
# regarding copyright ownership.  The ASF licenses
this file
# to you under the Apache License, Version 2.0
(the
# "License"); you may not use this file except in
compliance
# with the License.  You may obtain a copy of the
License at
#
#   http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to
in writing,
# software distributed under the License is
distributed on an
# "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS
OF ANY
# KIND, either express or implied.  See the
License for the
# specific language governing permissions and
limitations
# under the License.
#

# Basic Airflow cluster configuration for
CeleryExecutor with Redis and PostgreSQL.
#
# WARNING: This configuration is for local
development. Do not use it in a production
deployment.
#
# This configuration supports basic configuration
using environment variables or an .env file
# The following variables are supported:
#
# AIRFLOW_IMAGE_NAME          - Docker image name
used to run Airflow.
#                               Default:
apache/airflow:2.5.1
# AIRFLOW_UID                 - User ID in
Airflow containers
```

```

#                                     Default: 50000
# AIRFLOW_PROJ_DIR                   - Base path to
which all the files will be volumed.
#                                     Default: .
# Those configurations are useful mostly in case
of standalone testing/running Airflow in test/try-
out mode
#
# _AIRFLOW_WWW_USER_USERNAME         - Username for the
administrator account (if requested).
#                                     Default: airflow
# _AIRFLOW_WWW_USER_PASSWORD         - Password for the
administrator account (if requested).
#                                     Default: airflow
# _PIP_ADDITIONAL_REQUIREMENTS       - Additional PIP
requirements to add when starting all containers.
#                                     Default: ''
#
# Feel free to modify this file to suit your
needs.
---
version: '3'
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:2.5.1}
  # build: .
  environment:
    &airflow-common-env
    AIRFLOW__CORE__EXECUTOR: LocalExecutor
    AIRFLOW__DATABASE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/air
flow
    # For backward compatibility, with Airflow
<2.3
    AIRFLOW__CORE__SQL_ALCHEMY_CONN:
postgresql+psycopg2://airflow:airflow@postgres/air
flow
    AIRFLOW__CORE__FERNET_KEY: ''
    AIRFLOW__CORE__DAGS_ARE_PAUSED_AT_CREATION:
'true'
    AIRFLOW__CORE__LOAD_EXAMPLES: 'true'
    AIRFLOW__API__AUTH_BACKENDS:

```

```

'airflow.api.auth.backend.basic_auth,airflow.api.a
uth.backend.session'
    _PIP_ADDITIONAL_REQUIREMENTS:
${_PIP_ADDITIONAL_REQUIREMENTS:-}
    volumes:
      - ${AIRFLOW_PROJ_DIR:-
./dags:/opt/airflow/dags
      - ${AIRFLOW_PROJ_DIR:-
./logs:/opt/airflow/logs
      - ${AIRFLOW_PROJ_DIR:-
./plugins:/opt/airflow/plugins
    user: "${AIRFLOW_UID:-50000}:0"
    depends_on:
      &airflow-common-depends-on
    postgres:
      condition: service_healthy

services:
  postgres:
    image: postgres:13
    environment:
      POSTGRES_USER: airflow
      POSTGRES_PASSWORD: airflow
      POSTGRES_DB: airflow
    volumes:
      - postgres-db-
volume:/var/lib/postgresql/data
    ports:
      - 5432:5432
    healthcheck:
      test: ["CMD", "pg_isready", "-U", "airflow"]
      interval: 5s
      retries: 5
      restart: always

  airflow-webserver:
    <<: *airflow-common
    command: webserver
    ports:
      - 8080:8080
    healthcheck:
      test: ["CMD", "curl", "--fail",
"http://localhost:8080/health"]
      interval: 10s
      timeout: 10s
      retries: 5
      restart: always
    depends_on:
      <<: *airflow-common-depends-on
      airflow-init:

```

```

        condition: service_completed_successfully

airflow-scheduler:
  <<: *airflow-common
  command: scheduler
  healthcheck:
    test: ["CMD-SHELL", 'airflow jobs check --
job-type SchedulerJob --hostname "${HOSTNAME}"]
    interval: 10s
    timeout: 10s
    retries: 5
  restart: always
  depends_on:
    <<: *airflow-common-depends-on
  airflow-init:
    condition: service_completed_successfully

airflow-triggerer:
  <<: *airflow-common
  command: triggerer
  healthcheck:
    test: ["CMD-SHELL", 'airflow jobs check --
job-type TriggererJob --hostname "${HOSTNAME}"]
    interval: 10s
    timeout: 10s
    retries: 5
  restart: always
  depends_on:
    <<: *airflow-common-depends-on
  airflow-init:
    condition: service_completed_successfully

airflow-init:
  <<: *airflow-common
  entrypoint: /bin/bash
  environment:
    <<: *airflow-common-env
    _AIRFLOW_DB_UPGRADE: 'true'
    _AIRFLOW_WWW_USER_CREATE: 'true'
    _AIRFLOW_WWW_USER_USERNAME:
${_AIRFLOW_WWW_USER_USERNAME:-airflow}
    _AIRFLOW_WWW_USER_PASSWORD:
${_AIRFLOW_WWW_USER_PASSWORD:-airflow}
    _PIP_ADDITIONAL_REQUIREMENTS: ''
    user: "0:0"
  volumes:
    - ${AIRFLOW_PROJ_DIR:-.}:/sources

airflow-cli:
  <<: *airflow-common

```

```
profiles:
  - debug
environment:
  <<: *airflow-common-env
  CONNECTION_CHECK_MAX_COUNT: "0"
  # Workaround for entrypoint issue. See:
  https://github.com/apache/airflow/issues/16252
command:
  - bash
  - -c
  - airflow

volumes:
  postgres-db-volume:
```

File `docker-compose.yml` digunakan untuk mendefinisikan layanan dan konfigurasi Docker yang akan dijalankan sebagai bagian dari aplikasi atau infrastruktur yang menggunakan Docker Compose. Dengan menggunakan file `docker-compose.yml`, kami dapat dengan mudah mengelola dan menjalankan beberapa kontainer Docker yang saling berinteraksi, menjaga konfigurasi dan pengaturan aplikasi tetap terstruktur, dan mempercepat proses pengembangan dan penyebaran aplikasi yang menggunakan Docker.

b. Mengkonfigurasi di Docker menggunakan Power Shell

- Buka Power Shell

Pertama, ketikkan 'docker-compose up airflow-init'. Jika sudah berhasil, dapat dilanjutkan ke tahap berikutnya.

```
PS C:\Users\Acer\docker\airflow> docker-compose up airflow-init
```

Gambar 9. Inisialisasi lingkungan Airflow

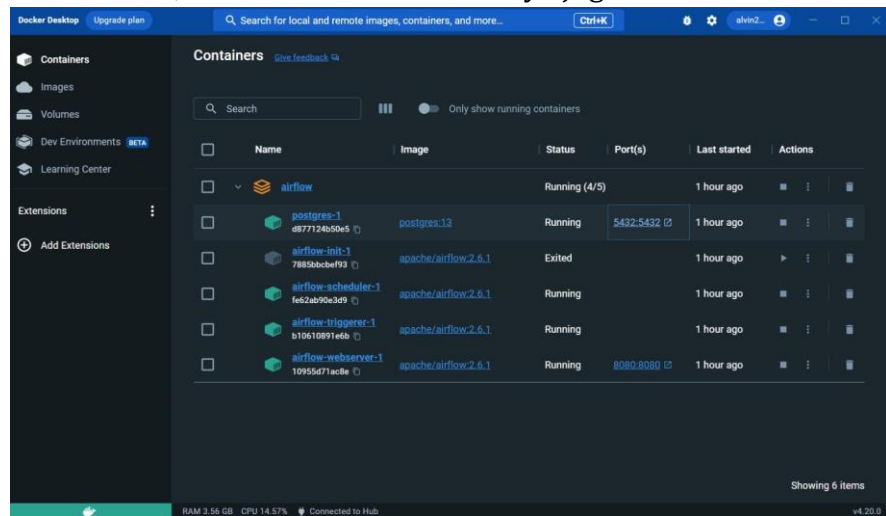
Kedua, ketikkan 'docker-compose up'. Jika sudah berhasil, maka proses membangun sistem airflow dan kontainer-kontainer telah berhasil.

```
PS C:\Users\Acer\docker\airflow> docker-compose up
```

Gambar 10. Membuat dan menjalankan kontainer

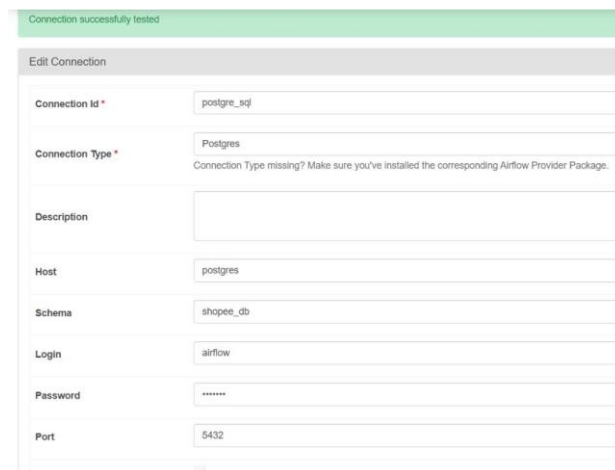
- Tampilan di Docker

Buka aplikasi Docker. Maka akan terlihat bahwa sistem airflow sudah terbentuk, dan kontainer di dalamnya juga telah terbentuk.



Gambar 11. Tampilan docker

- Mengatur koneksi Postgres di antarmuka Airflow



Gambar 12. Koneksi di Airflow

Untuk mengatur koneksi Postgres di antarmuka Airflow, dilakukan langkah-langkah berikut:

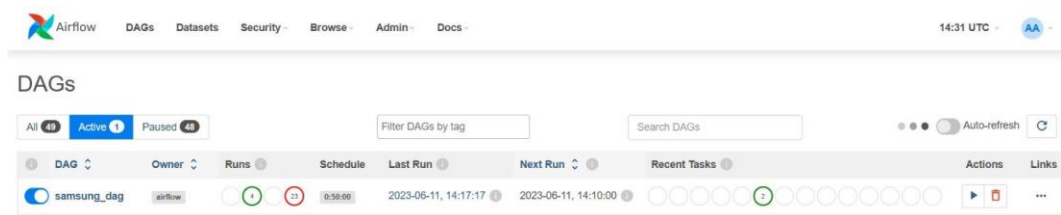
1. Buka antarmuka local <http://localhost:8080/home>
2. Di menu utama, pilih "Admin".
3. Pilih "Connections" dari menu dropdown yang muncul.
4. Di halaman Connections, klik tombol "Create".
5. Di bagian "Conn Id", masukkan sebuah ID unik untuk koneksi Postgres sesuai dengan yang telah dibuat pada skrip `samsung_dag.py` yaitu, "postgres_sql".
6. Pada "Conn Type", pilih "Postgres".

7. Di bagian "Host", masukkan alamat host dari server yaitu, postgres.
8. Masukkan nomor "Port" yang digunakan oleh server Postgres. Secara default, port Postgres adalah 5432.
9. Masukkan "Schema" yang akan digunakan untuk koneksi yaitu, shopee_db.
10. Di bagian "Login", masukkan nama pengguna yang digunakan untuk mengakses database Postgres.
11. Masukkan "Password" yang sesuai dengan nama pengguna yang telah dimasukkan sebelumnya.
12. Setelah semua informasi telah dimasukkan dengan benar, klik tombol "Save" untuk menyimpan koneksi Postgres.

Koneksi Postgres sekarang telah dikonfigurasi dan dapat digunakan dalam DAG atau task di Airflow UI.

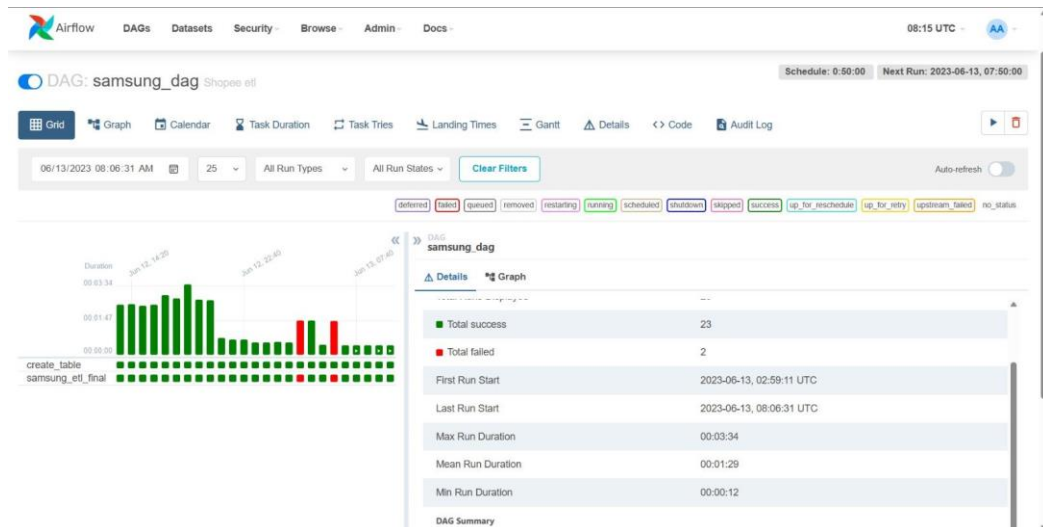
- Me-load DAGs pada localhost:8080

Klik menu 'DAGs', maka sistem akan memuat ulang. Dari hasil yang diperoleh, file `samsung_dag` sudah terdeteksi.



Gambar 13. Tampilan DAG

Kemudian, sistem akan mencoba menjalankan tugas dari skrip `samsung_dag.py` dan `samsung_etl.py`. Ketika tugas `create_table` dan `samsung_etl_final` telah berwarna hijau artinya tugas tersebut telah sukses, maka kita dapat mencoba melihat hasil dari menu Logs.



Gambar 14. Menjalankan tugas di DAG

4.3 Load to PostgreSQL

- Mengecek hasil database dengan Power Shell

Untuk memvalidasi hasil database, kami melakukan koneksi dengan PostgreSQL melalui terminal Power Shell. Berikut ini adalah langkah-langkah yang kami lakukan:

- Masuk ke 'airflow'

```
(airflow)psql -h postgres -p 5432 -U airflow -d airflow
Password for user airflow:
psql (15.3 (Debian 15.3-1.pgdg110+1), server 13.11 (Debian 13.11-1.pgdg110+1))
Type "help" for help.
```

Gambar 15. Masuk ke Airflow

Mengetik **psql -h postgres -p 5432 -U airflow -d airflow**

h berarti host, p berarti port, U berarti user, dan d berarti database.

- Membuat database

```
airflow=# \c shopee_db
psql (15.3 (Debian 15.3-1.pgdg110+1), server 13.11 (Debian 13.11-1.pgdg110+1))
You are now connected to database "shopee_db" as user "airflow".
shopee_db=# \d
```

Gambar 16. Membuat database

Untuk membuat database, ketik **\c shopee_db**. Maka database berhasil terbuat dan terkoneksi.

- Melihat isi dari shopee_db

Untuk melihat isi dari shopee_db, ketikkan **\d**, maka akan terlihat hasil sebagai berikut. Dari hasil tersebut, terlihat bahwa terdapat tabel **toko_shopee**.

```
shopee_db=# \d
List of relations
Schema | Name | Type | Owner
-----+-----+-----+-----
public | toko_shopee | table | airflow
(1 row)
```

Gambar 17. Melihat isi file database

d. Menampilkan hasil database

Untuk menampilkan hasil database, dapat ketikkan `SELECT * toko_shopee`. Maka akan didapatkan hasil sebagai berikut.

```
shopee_db=# SELECT * FROM toko_shopee;
```

index	name	price	currency	stock	rating_star
0	Samsung Galaxy A14 6/128GB - Silver	267210000000	IDR	53	4.988178
1	Samsung UA43N5091AH Full HD LED TV [43 Inch]	319000000000	IDR	23	4.867889
2	Samsung Galaxy Tab A8 WiFi 3/32 GB - Gray	299900000000	IDR	14	4.879227
3	Samsung Galaxy A14 5G 6/128GB - Dark Red	299900000000	IDR	169	4.887981
4	Samsung Galaxy A14 5G 6/128GB - Silver	299900000000	IDR	98	4.893733
5	Samsung Galaxy M23 5G 6/128GB - Light Blue	331900000000	IDR	1345	4.885819
6	Samsung Galaxy Tab A7 Lite WiFi 3/32GB - Gray	186900000000	IDR	12	4.887931
7	Samsung Galaxy A14 6/128GB - Black	263630000000	IDR	10	4.889043
8	Samsung Smart TV 43 inch UHD 4K AU7082 dengan PurColor - UA43AU7082KXXD	417900000000	IDR	68	4.908193
9	Samsung Galaxy A14 4/128GB - Dark Red	249900000000	IDR	527	4.899329
10	Samsung Galaxy A04e 3/32GB - Light Blue	124900000000	IDR	266	4.87939
11	Samsung Galaxy Tab A7 Lite 3/32GB - Silver	226900000000	IDR	8	4.885159
12	Samsung Galaxy Tab A7 Lite 3/32GB - Gray	226900000000	IDR	19	4.899379
13	Samsung Galaxy A54 5G 8/256GB - Awesome White	639900000000	IDR	58	4.94359
14	Samsung HD TV 32" UA32T4001 (2020)	176900000000	IDR	493	4.88391
15	Samsung Galaxy M14 5G 4/128GB - Blue	277900000000	IDR	225	4.728339
16	Samsung Galaxy Z Flip4 5G 8/256GB - Bora Purple	149990000000	IDR	7	4.768116
17	Samsung Smart TV 58 inch UHD 4K AU7082 dengan PurColor - UA58AU7082KXXD	596900000000	IDR	72	4.933638
18	Samsung Galaxy A14 4/128GB - Silver	249900000000	IDR	256	4.914343
19	Samsung Galaxy A04s 4/64GB - Black	189900000000	IDR	38	4.908915
20	Samsung Galaxy A23 5G 6/128GB - Light Blue	399900000000	IDR	1	4.911765
21	Samsung Galaxy M14 5G 4/128GB - Silver	277900000000	IDR	18	4.754545
22	Samsung Galaxy A34 5G 8/128GB - Awesome Violet	494910000000	IDR	15	4.885135
23	Samsung Galaxy A04e 3/64GB - Light Blue	137900000000	IDR	52	4.936396
24	Samsung 32 inch HD Smart TV T4501 dengan PurColor	233900000000	IDR	38	5
25	Samsung Galaxy M14 5G 4/64GB - Silver	259900000000	IDR	49	0
26	Samsung Galaxy A04 4/64GB - Black	159900000000	IDR	2	4.913129
27	Samsung Galaxy A04e 3/32GB - Black	124900000000	IDR	691	4.883989
28	Samsung Galaxy A54 5G 8/128GB - Awesome Lime	599900000000	IDR	12	4.888889
29	Samsung Galaxy A04s 4/64GB - Green	289900000000	IDR	2	4.914844
30	Samsung Galaxy A14 4/128GB - Black	249900000000	IDR	369	4.92623
31	Samsung Smart HD TV 32" T4500 - UA32T4500AH	256900000000	IDR	12	4.883429
32	Samsung Galaxy A14 5G 6/128GB - Black	299900000000	IDR	224	4.918415
33	Samsung Galaxy A54 5G 8/128GB - Awesome Violet	599900000000	IDR	71	4.953357
34	Samsung Galaxy A23 5G 6/128GB - Silver	395918000000	IDR	3	4.875
35	Samsung Galaxy A04 4/64GB - Copper	159900000000	IDR	8	4.91358
36	Samsung Smart TV 65 inch UHD 4K AU7082 dengan PurColor - UA65AU7082KXXD	834900000000	IDR	12	4.898989
37	Samsung Galaxy Z Flip4 5G 8/128GB - Graphite	124900000000	IDR	1	5
38	Samsung Galaxy M23 5G 6/128GB - Orange Copper	331900000000	IDR	895	4.836043
39	Samsung Galaxy A54 5G 8/128GB - Awesome White	599900000000	IDR	11	4.915385
40	Samsung Galaxy A54 5G 8/256GB - Awesome Violet	639900000000	IDR	14	4.948217
41	Samsung Galaxy A34 5G 8/128GB - Awesome Silver	494910000000	IDR	25	4.851485
42	Samsung Galaxy A04 4/64GB - Green	159900000000	IDR	20	4.902862

Gambar 18. Menampilkan isi database Shopee

5. BAB V: KESIMPULAN

Pada proyek ini, telah dilakukan pembangunan ETL pipeline yang terintegrasi untuk toko elektronik Samsung di marketplace Shopee. Proses ETL melibatkan ekstraksi data produk, transformasi data, dan pemuatan data ke dalam database PostgreSQL. Data seperti nama produk, harga, stok, dan nilai bintang produk diekstrak menggunakan teknik scraping melalui API Shopee. Selanjutnya, dibangun sebuah pipeline menggunakan Apache Airflow dalam lingkungan Docker untuk mengatur alur kerja ETL secara otomatis. Setelah pipeline selesai dibangun, koneksi ke database PostgreSQL ditetapkan untuk menjalankan sistem ETL secara otomatis. Hasilnya adalah database PostgreSQL yang berisi informasi lengkap tentang produk elektronik toko Samsung, seperti nama produk, harga, jenis mata uang, stok, dan peringkat bintang. Dengan adanya ETL pipeline ini, pengelolaan data produk toko Samsung pada Shopee menjadi lebih mudah dan data dapat dianalisis untuk pengambilan keputusan yang lebih baik.

6. DAFTAR PUSTAKA

- Cindy Mutia Annur (2022, November). Pengiriman smartphone ri Lesu di Q3 2022, ini Penguasa Pasar HP di Indonesia?: Databoks. Pusat Data Ekonomi dan Bisnis Indonesia.
- Erwin E.W., Muhammad A., Andi D., Danang E.N, dkk (2022, Mei). Akuisisi Data prediksi Curah Hujan Secara Periodik menggunakan apache Airflow. Journal of Informatics Information System Software Engineering and Applications (INISTA).
- I Putu Widia P., I Nyoman Hary K. (2021, Juni). Implementasi ETL (Extract, Transform, Load) pada Data warehouse Penjualan Menggunakan Tools Pentaho. View of data warehouse implementasi ETL (extract, transform, load) Pada data warehouse Penjualan Menggunakan tools pentaho. TIERS Informations Technology Journal.
- Muhammad Romzi, Budi Kurniawan (2021, Desember). PEMBELAJARAN PEMROGRAMAN PYTHON DENGAN PENDEKATAN LOGIKA ALGORITMA. Tampilan Pembelajaran Pemrograman Python Dengan Pendekatan logika algoritma. Jurnal TeknikInformatikaMahakarya.
- Samsung Newsroom Indonesia (2022, November 4). Samsung Electronics Masuk Lima Besar best global brands 2022. <https://news.samsung.com/id/samsung-electronics-masuk-lima-besar-best-global-brands-2022>
- Sidharth Ramalingam (2023, January 30). Data Engineering project-2: Building Spotify ETL using python and airflow. Medium. <https://blog.devgenius.io/data-engineering-project-2-building-spotify-etl-using-python-and-airflow-432dd8e4ffa3>
- Similarweb(2023, May). Top marketplace websites ranking in Indonesia in May 2023. <https://www.similarweb.com/top-websites/indonesia/e-commerce-and-shopping/marketplace/>

Stephen Kusumo, Ramos Somya (2022, Desember). Penerapan web scraping
DESKRIPSI produk Menggunakan selenium python dan . JATISI (Jurnal Teknik
Informatika dan Sistem Informasi).

7. LAMPIRAN

Link video YouTube: <https://youtu.be/SxpUN4C-00A>

Link kode script:

https://drive.google.com/drive/folders/19J3K1t0i592Xkci_5nN3nuNaF50odXJ-?usp=sharing