DATA ENGINEER UNTUK ENTRY LEVEL

DigitalSkola

FINAL PROJECT

BY : DREAM (KELOMPOK 1)

Data Realm Engineers And Maestros

DigitalSkola

NAMA KELOMPOK

01 Afroh Fauziah

02 Andi Rosilala

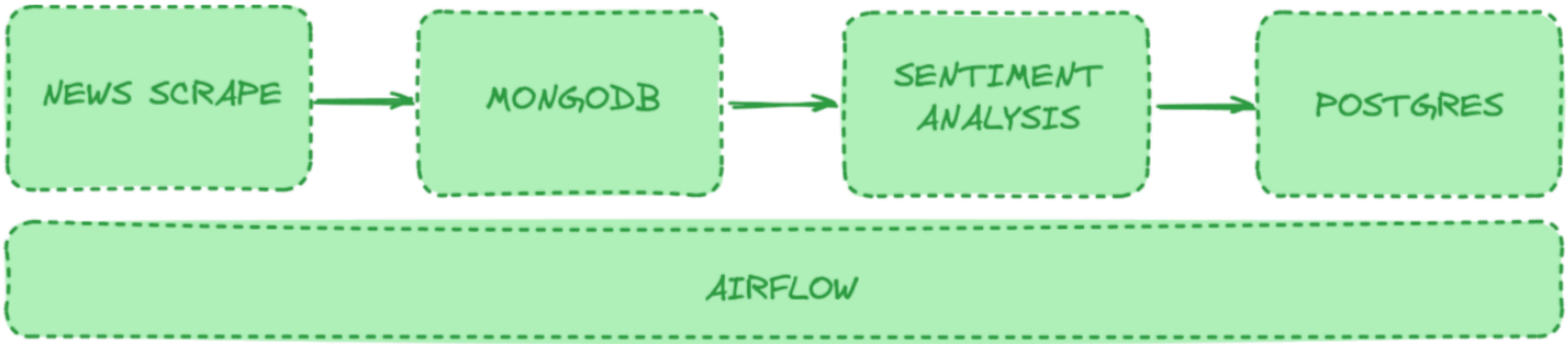
03 Althaf Nawadir Taqiyyah

04 Andrew Fortino Mahardika Suadnya

TUJUAN PROJEK

- **Membangun pipeline otomatis untuk mengakuisisi data *news* dari API Finnhub.**
- **Melakukan analisis sentimen terhadap data *news* yang telah dikumpulkan.**
- **Menyimpan hasil analisis data *news* dalam database.**

~ “CASE STUDY” ~



Membuat Batch Data Pipeline

Dimulai dari schedule menggunakan airflow dengan data source New Scrape lalu di load ke MongoDB Atlas. Dengan Sentiment Analysis kemudian di load ke Postgres hingga menjadi data warehouse.

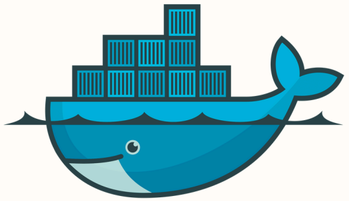
~ TOOLS ~

Finnhub

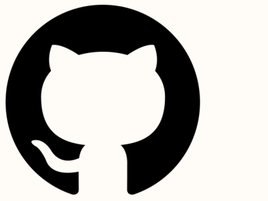
Apache Airflow

mongoDB.

PostgreSQL



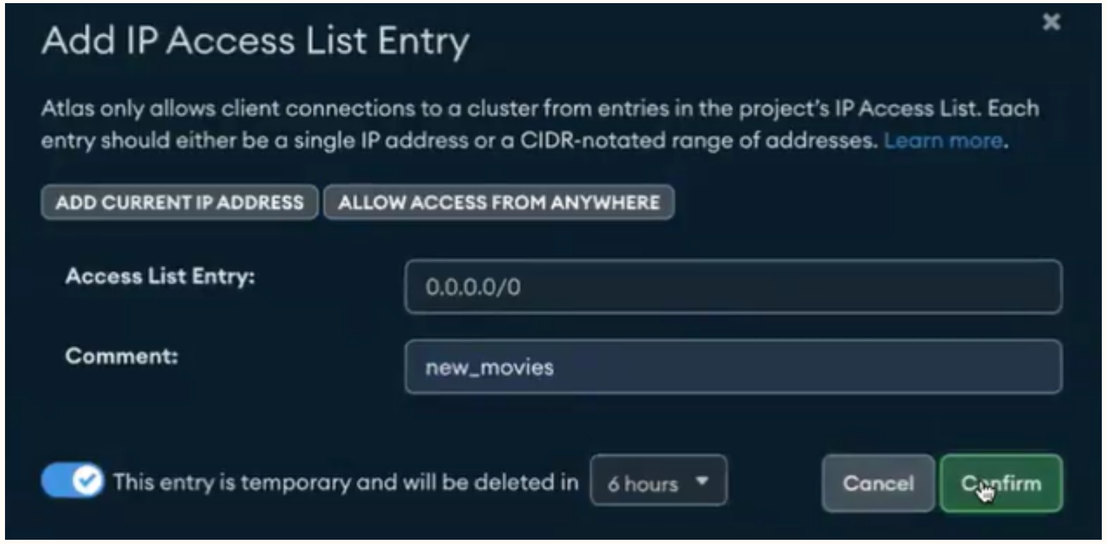
docker



GitHub

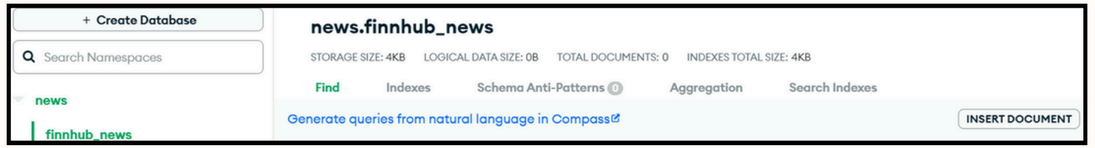
~ MONGODB ~

1. Sign Up / Login ke Finnhub ‘https://filearnnnhub.io/docs/api/symbol-search’ dan juga MongoDB Atlas ‘https://cloud.mongodb.com’
2. Lalu masuk ke Network Access dan buat Ip Address baru.
3. Setelah itu buka database dan masuk ke collections learn mongo untuk create database. Juga dapat membuat database access user baru.

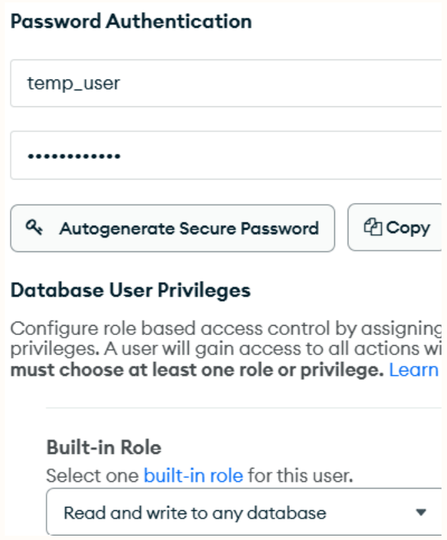


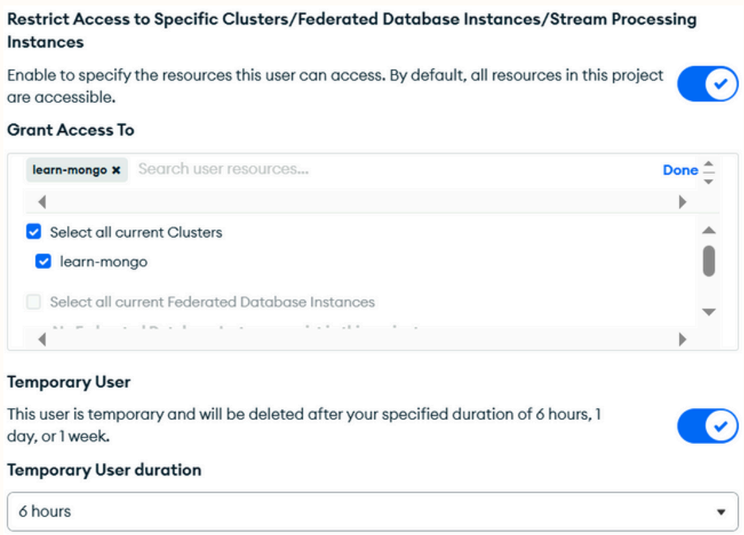
Network Access

Database



Database Access





~ DOCKER-COMPOSE.YML ~

Membuat file docker-compose.yml untuk menjalankan docker dimana terdapat services postgres 12, airflow init, webserver, & scheduler.

docker-compose.yml

```yaml
version: '3'
x-airflow-common:
  &airflow-common
  image: apache/airflow:2.8.4-python3.10
  environment:
    - AIRFLOW__CORE__EXECUTOR=LocalExecutor
    - AIRFLOW__CORE__SQL_ALCHEMY_CONN=postgresql+psycopg2://airflow:airflow@postgres:5432/airflow
    - AIRFLOW__CORE__FERNET_KEY=FB0o_zt4e3Ziq3LdUUO7F2Z95cvFFx16hU8jTeR1ASM=
    - AIRFLOW__CORE__LOAD_EXAMPLES=False
    - AIRFLOW__CORE__LOGGING_LEVEL=INFO
  volumes:
    - ./dags:/opt/airflow/dags
    - ./source:/opt/airflow/source
    - ./output:/opt/airflow/output
    - ./logs:/opt/airflow/logs
    - ./plugins:/opt/airflow/plugins
    - ./requirements.txt:/requirements.txt

services:
  postgres:
    image: postgres:12
    environment:
      - POSTGRES_USER=airflow
      - POSTGRES_PASSWORD=airflow
      - POSTGRES_DB=airflow
      - POSTGRES_PORT=5432
    volumes:
      - postgres-db-volume:/var/lib/postgresql/data
    healthcheck:
      test: ["CMD", "pg_isready", "-U", "airflow"]
      interval: 5s
      retries: 5
    ports:
      - "5432:5432"

  airflow-init:
    << : *airflow-common
    container_name: airflow_init
    entrypoint: /bin/bash
    command:
      - -c
      - airflow db init &&
        airflow users create
          --role Admin
          --username airflow
          --password airflow
          --email airflow@airflow.com
          --firstname airflow
          --lastname airflow
    restart: on-failure
    depends_on:
      postgres:
        condition: service_healthy

  airflow-webserver:
    << : *airflow-common
    command: >
      bash -c "python3 -m pip install --upgrade pip &&
      pip install --no-cache-dir -r /requirements.txt &&
      airflow webserver"
    ports:
      - 8081:8080
    container_name: airflow_webserver
    depends_on:
      airflow-init:
        condition: service_completed_successfully
    restart: always

  airflow-scheduler:
    << : *airflow-common
    command: >
      bash -c "python3 -m pip install --upgrade pip &&
      pip install --no-cache-dir -r /requirements.txt &&
      airflow scheduler"
    container_name: airflow_scheduler
    depends_on:
      airflow-init:
        condition: service_completed_successfully
    restart: always

volumes:
  postgres-db-volume:
```

~ DOCKER-COMPOSE.YML ~

Untuk menjalankannya gunakan ‘docker compose up’ dan tunggu sampai semua berhasil running.

```
PS D:\SIB6\hands on\FINALPROJECT> docker-compose up --build
time="2024-06-24T13:05:50+07:00" level=warning msg="D:\\SIB6\\hands on\\FINALPROJECT\\docker-compose.yml: `version` is obso
ete"
[+] Running 40/4
 ✓ airflow-scheduler Pulled                                                                                    611.8s
 ✓ airflow-webserver Pulled                                                                                    611.8s
 ✓ postgres Pulled                                                                                             435.6s
 ✓ airflow-init Pulled                                                                                         611.6s
[+] Running 2/6
 ✓ Network finalproject_default             Created                                                             0.3s
 ✓ Volume "finalproject_postgres-db-volume" Created                                                             0.1s
 - Container finalproject-postgres-1        Created                                                             2.6s
 - Container airflow_init                   Created                                                             1.3s
 - Container airflow_webserver              Created                                                             0.9s
 - Container airflow_scheduler              Created                                                             0.9s
Attaching to airflow_init, airflow_scheduler, airflow_webserver, postgres-1
postgres-1          | The files belonging to this database system will be owned by user "postgres".
```

Cek statusnya pada docker desktop / dengan ‘docker ps’ ataupun localhost webserver yang sesuai.

```
PS D:\SIB6\hands on\FINALPROJECT> docker ps
CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS
    PORTS                    NAMES
50ceee6f51fc   apache/airflow:2.8.4-python3.10   "/usr/bin/dumb-init …"   9 minutes ago    Up 7 minutes
    0.0.0.0:8081->8080/tcp   airflow_webserver
d6e92dcabb5d   apache/airflow:2.8.4-python3.10   "/usr/bin/dumb-init …"   11 minutes ago   Up 10 minutes
    8080/tcp                 airflow_scheduler
c0dd7a127ead   apache/airflow:2.8.4-python3.10   "/bin/bash -c 'airfl…"   11 minutes ago   Up 15 seconds
    8080/tcp                 airflow_init
54d19f78cf8d   postgres:12                    "docker-entrypoint.s…"   11 minutes ago   Up 11 minutes (healthy
)   0.0.0.0:5432->5432/tcp   finalproject-postgres-1
```

DigitalSkola
Uncover The World of Digital Skills with Us

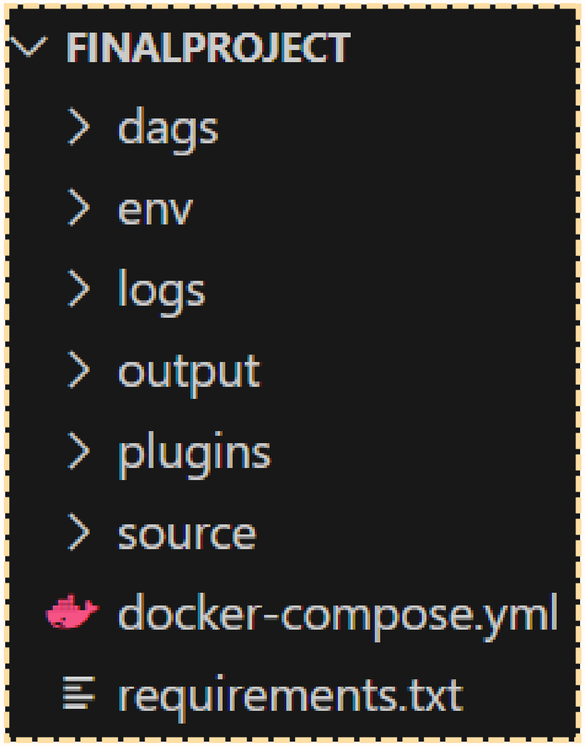
“REQUIREMENTS”

requirements.txt

```
pandas
pymongo
Flask-SQLAlchemy
textblob
finnhub-python
```

Buat file requirements.txt untuk mendefinisikan dependensi yang digunakan dalam konteks pengelolaan lingkungan virtual untuk diinstall.

Tampilan saat docker berhasil running akan otomatis membuat volumes sebagaimana query.



plugins > mongodb_loader.py > ...

```python
import pymongo

def get_mongo_client(mongo_uri):
    """Establish connection to the MongoDB."""
    try:
        client = pymongo.MongoClient(mongo_uri)
        # Send a ping to confirm a successful connection
        try:
            client.admin.command('ping')
            print("Pinged your deployment. You successfully connected to MongoDB!")

            return client
        except Exception as e:
            print(e)
    except pymongo.errors.ConnectionFailure as e:
        print(f"Connection failed: {e}")
        return None

def load(database, collection):
    mongo_uri = "mongodb+srv://temp_user:temp_user123@learn-mongo.[redacted]"
    if not mongo_uri:
        print("MONGO_URI not set in environment variables")

    mongo_client = get_mongo_client(mongo_uri)

    # Ingest data into MongoDB
    db = mongo_client[database]
    col = db[collection]

    return col
```

Di dalam querynya terdapat koneksi, mengakses, dsb.

Dengan membuat file mongodb_loader.py dalam folder plugins ini akan berfungsi untuk mengelola proses memasukkan (load) data ke dalam database MongoDB.

“MONGODB LOADER”

DigitalSkola

FINNHUB LOADER

File finnhub_loader.py ini memiliki fungsi untuk mengambil berita umum dari Finnhub, sebuah layanan API yang menyediakan data pasar keuangan.

```
plugins > finnhub_loader.py > ...
1  import finnhub
2
3  def scrape_news():
4      finnhub_client = finnhub.Client(api_key="[redacted]")
5
6      news = finnhub_client.general_news('general', min_id=0)
7
8      return news
```

Dengan membuat objek finnhub client menggunakan api key, memanggil general news dan panggilan data return news

POSTGRES LOADER

File postgres_loader.py memiliki fungsi untuk memuat data ke dalam tabel PostgreSQL. Dengan mengimport pustaka yang dibutuhkan, memuat data ke dalam tabel PostgreSQL, Variabel & String Koneksi, Membuat Objek Koneksi juga pesan jika sukses dijalankan.

```
plugins > postgres_loader.py > ...
1  from sqlalchemy import create_engine
2
3
4  def load(data, table_name):
5      user = 'airflow'
6      passwd = 'airflow'
7      hostname = 'postgres'
8      database = 'data_warehouse'
9
10     conn_string = f'pestgresql://{user}:{passwd}@{hostname}:5432/{database}'
11
12     db = create_engine(conn_string)
13     conn = db.connect()
14
15     data.to.sql(table_name, con=conn, if_exists='append',
16                 index=False)
17
18     print("Successfully loaded to postgres")
```

SENTIMENT ANALYSIS

File sentiment_analysis.py memiliki fungsi untuk melakukan analisis sentimen terhadap teks yang diberikan menggunakan pustaka TextBlob. Terdapat class SentimentAnalysis, metode execute untuk melakukan analisis sentimen pada teks yang diberikan, membuat objek TextBlob juga set sentiment. File ini nantinya akan mempengaruhi sentiment analysis loader untuk menjalankannya.

plugins > sentiment_analysis.py > ...

```python
import re
from textblob import TextBlob


class SentimentAnalysis:

    def __init__(self, text):
        self.text = text

    # only for english language
    def execute(self):
        # create TextBlob object of passed tweet text
        analysis = TextBlob(self.text)

        # set sentiment
        if analysis.sentiment.polarity > 0:
            data = {'text': self.text, 'sentiment': 'positive'}
        elif analysis.sentiment.polarity == 0:
            data = {'text': self.text, 'sentiment': 'neutral'}
        else:
            data = {'text': self.text, 'sentiment': 'negative'}

        return data


if __name__ == "__main__":
    # calling main function
    SentimentAnalysis('hard to learn NLTK').execute()
```

FINNHUB MONGODB LOADER

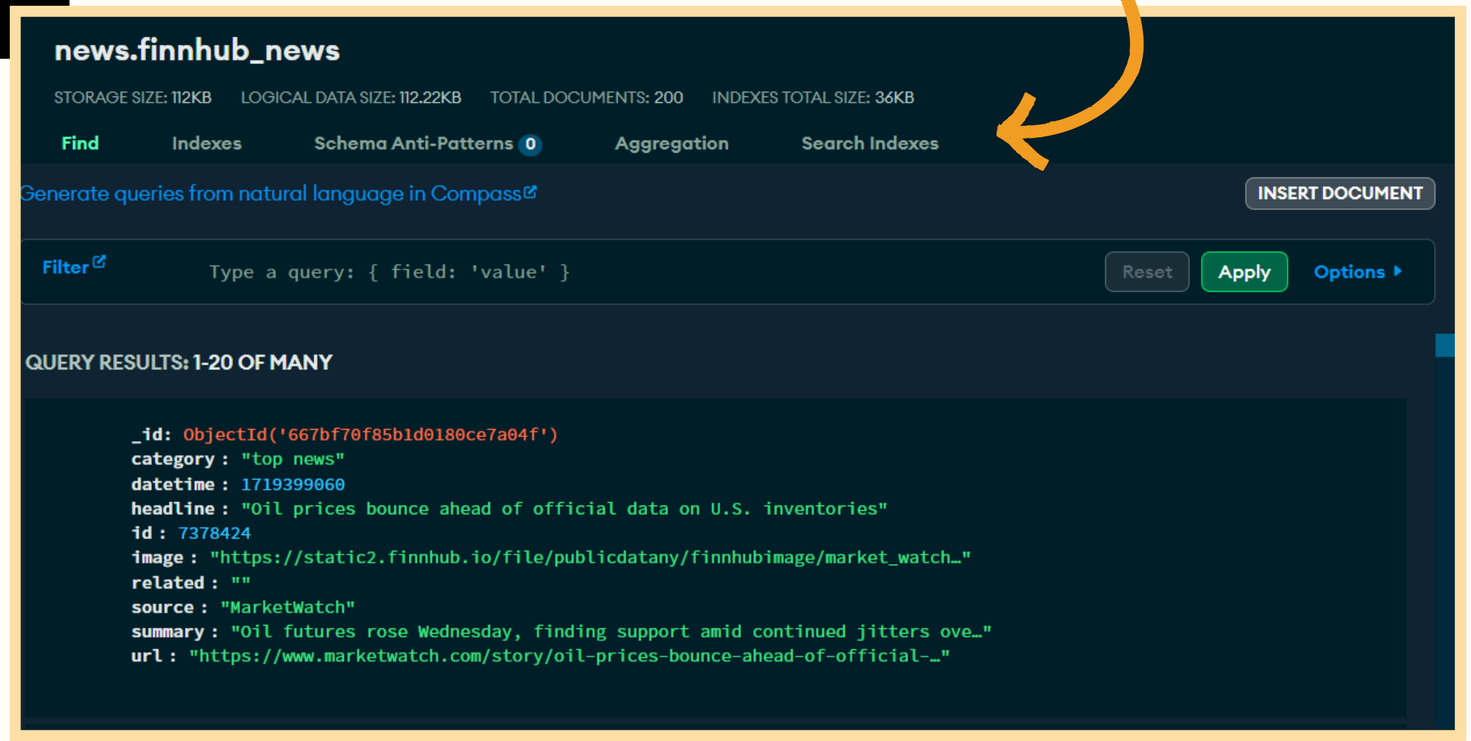
plugins > finnhub_mongodb_loader.py > ...

```python
import finnhub_loader
import mongodb_loader


def extract_load():
    news = finnhub_loader.scrape_news()

    collection = mongodb_loader.load('news', 'finnhub_news')
    collection.insert_many(news)

    print("Successfully load news to MongoDB")


if __name__ == "__main__":
    extract_load()
```

File finnhub_mongodb_loader ini digunakan untuk mengekstraksi berita dari sumber data tertentu menggunakan modul finnhub_loader dan kemudian memuat data tersebut ke dalam koleksi MongoDB menggunakan modul mongodb_loader.

```
(andrew) D:\Downloads\FinalProject_DE_DigitalSkola>python D:\Downloads\FinalProjec
t_DE_DigitalSkola\plugins\finnhub_mongodb_loader.py
Pinged your deployment. You successfully connected to MongoDB!
Successfully load news to MongoDB
```

Setelah query dibuat maka selanjutnya jalankan python sampai berhasil dan cek status query berhasl masuk tidaknya pada MongoDB Atlas.



DigitalSkola

SENTIMENT ANALYSIS LOADER

plugins > sentiment_analysis_loader.py > ...

```python
import mongodb_loader
import pandas as pd
from sentiment_analysis import SentimentAnalysis
import postgres_loader

def run():
    db = mongodb_loader.get_data("news", "finnhub_news")

    news = [x for x in db.finnhub_news.find()]

    output = []
    for news_summary in news:
        output.append(SentimentAnalysis(text=news_summary["summary"]).execute())

        print(f"Summary {news_summary['summary']} successfully analized")

    sentiment_output = pd.DataFrame(output)

    postgres_loader.load(sentiment_output, "sentiment_news_analysis")

    print("Successfully loaded to Postgres")

if __name__ == "__main__":
    run()
```

Skrip Python untuk mengekstraksi database news dari MongoDB, melakukan analisis sentimen pada data tersebut, dan kemudian memuat hasil analisis ke dalam database PostgreSQL.

Cek hasil python dari sentiment_analysis_loader.py sampai berhasil keakses.

```
(andrew) D:\Downloads\FinalProject_DE_DigitalSkola>python D:\Downloads\FinalProject_DE_DigitalSkola\plugins\sentiment_analysis_loader.py
Pinged your deployment. You successfully connected to MongoDB!
Summary Oil futures rose Wednesday, finding support amid continued jitters over the potential for a broader Middle East conflict as trader
 succesfully analized
```

HASIL ANALISIS SENTIMENT

Summary succesfully analized

	text	sentiment
0	Oil futures rose Wednesday, finding support am...	negative
1	Societe Generale’s U.S. equity team says inves...	positive
2	A senior research economist at the Federal Res...	negative
3	“We have no debt and live comfortably on $150,...	positive
4	Citadel founder Ken Griffin received a rare pu...	positive
..	...	...
195	Former Treasury Secretary Larry Summers on Mon...	neutral
196	Are rising jobless claims a sign of emerging w...	negative
197	“Recent election surprises in Mexico, India, a...	neutral
198	The brewer of Corona and Modelo is among the b...	positive
199		neutral

[200 rows x 2 columns]

finalproject_de_digitalskola-postgres-1

postgres:12

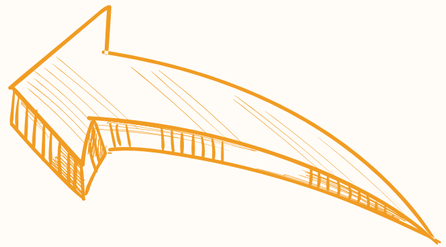
324ac8819086

5432:5432

Logs | Inspect | Bind mounts | Exec | Files | Stats

```
# psql -Uairflow
psql (12.19 (Debian 12.19-1.pgdg120+1))
Type "help" for help.

airflow=# create database data_warehouse
airflow-# \l
                                  List of databases
      Name       |  Owner   | Encoding |  Collate   |   Ctype    |  Access privileges
-----------------+----------+----------+------------+------------+---------------------
 airflow         | airflow | UTF8     | en_US.utf8 | en_US.utf8 |
 data_warehouse  | airflow | UTF8     | en_US.utf8 | en_US.utf8 |
 postgres        | airflow | UTF8     | en_US.utf8 | en_US.utf8 |
 template0       | airflow | UTF8     | en_US.utf8 | en_US.utf8 | =c/airflow          +
                 |          |          |            |            | airflow=CTc/airflow
 template1       | airflow | UTF8     | en_US.utf8 | en_US.utf8 | =c/airflow          +
                 |          |          |            |            | airflow=CTc/airflow
(5 rows)
```

LOAD HASIL ANALISIS SENTIMENT KE DATA_WAREHOUSE



Tampilan pada docker desktop untuk melihat data warehouse.

```python
from airflow import DAG
from airflow.operators.python import PythonOperator
from datetime import datetime, timedelta

import finnhub_mongodb_loader
import sentiment_analysis_loader



default_args = {
    'owner': 'de-team',
    'depends_on_past': False,
    'start_date': datetime(2023, 9, 21),
    'wait_for_downstream': False,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
    'sla': timedelta(days=1),
}

schedule_interval = '0 0 * * *'

with DAG(
    'dag_sentiment_analysis',
    default_args=default_args,
    schedule_interval=schedule_interval,
    catchup=False,
    tags=['machine-learning']
) as dag:
    extract_load = PythonOperator(
        task_id=f'extract_load',
        python_callable=finnhub_mongodb_loader.extract_load
    )

    sa_load = PythonOperator(
        task_id=f'sa_load',
        python_callable=sentiment_analysis_loader.run
    )

    extract_load >> sa_load
```

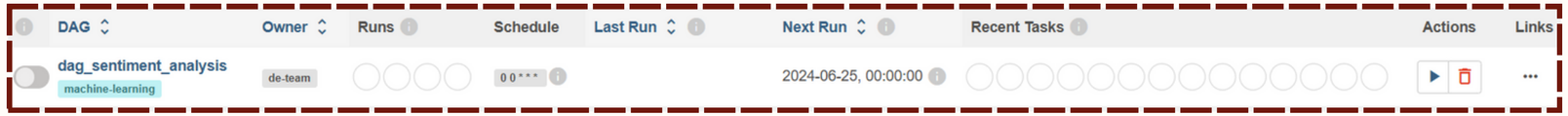
SENTIMENT ANALYSIS FOR AIRFLOW SCHEDULE

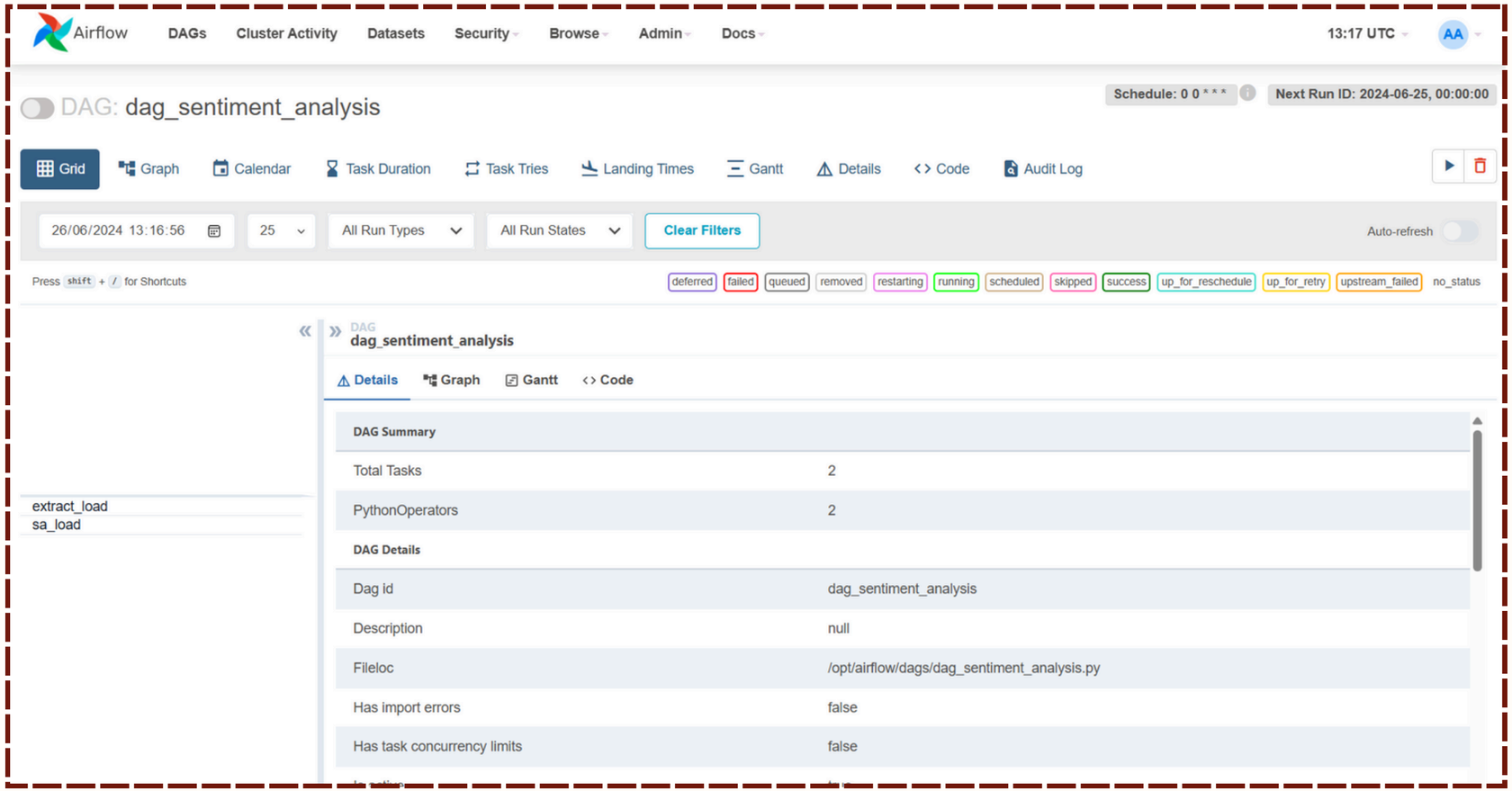
Script python ini berguna untuk membuat dan mengelola Directed Acyclic Graph (DAG) menggunakan Apache Airflow yang berfungsi untuk mengotomatisasi proses ekstraksi data dari database news, melakukan analisis sentimen, dan pemuatan data ke dalam MongoDB dan PostgreSQL.

DigitalSkola

HASIL SCHEDULED AIRFLOW

DigitalSkola





Tampilan dags saat berhasil memanggil dan menjalankan airflow.

DigitalSkola

TERIMA KASIH

ATAS PERHATIANNYA

BY : DREAM (KELOMPOK 1)