Jawaban Task 2

from airflow import DAG

from airflow.operators.http\_operator import SimpleHttpOperator

from airflow.operators.postgres\_operator import PostgresOperator

from airflow.providers.postgres.hooks.postgres import PostgresHook

from airflow.utils.dates import days\_ago

from datetime import datetime

import json

default\_args = {

'owner': 'airflow',

'depends\_on\_past': False,

'start\_date': days\_ago(1),

'retries': 1,

'retry\_delay': timedelta(minutes=5),

}

dag = DAG(

'gender\_prediction\_dag',

default\_args=default\_args,

description='A DAG for gender prediction and data loading to PostgreSQL',

schedule\_interval='@daily',

)

def predict\_gender(\*\*kwargs):

url = 'https://api.genderize.io/'

names = ['Sandra', 'John', 'Alice'] # Example names for prediction

for name in names:

response = requests.get(url, params={'name': name})

prediction\_data = response.json()

print(prediction\_data)

predict\_task = SimpleHttpOperator(

task\_id='predict\_gender',

http\_conn\_id='gender\_api',

endpoint='predict',

method='GET',

headers={"Content-Type": "application/json"},

xcom\_push=True,

dag=dag,

)

def create\_table(\*\*kwargs):

create\_table\_sql = """

CREATE TABLE IF NOT EXISTS gender\_name\_prediction (

input JSONB,

details JSONB,

result\_found BOOLEAN,

first\_name VARCHAR(100),

probability FLOAT,

gender VARCHAR(10),

timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

"""

pg\_hook = PostgresHook(postgres\_conn\_id='postgres\_conn')

pg\_hook.run(sql=create\_table\_sql)

create\_table\_task = PythonOperator(

task\_id='create\_table',

python\_callable=create\_table,

dag=dag,

)

def load\_to\_postgres(\*\*kwargs):

ti = kwargs['ti']

prediction\_results = ti.xcom\_pull(task\_ids='predict\_gender')

pg\_hook = PostgresHook(postgres\_conn\_id='postgres\_conn')

for result in prediction\_results:

pg\_hook.run(

f"""

INSERT INTO gender\_name\_prediction (input, details, result\_found, first\_name, probability, gender)

VALUES (

'{json.dumps(result["input"])}',

'{json.dumps(result["details"])}',

{result["result\_found"]},

'{result["first\_name"]}',

{result["probability"]},

'{result["gender"]}'

);

"""

)

load\_to\_postgres\_task = PythonOperator(

task\_id='load\_to\_postgres',

python\_callable=load\_to\_postgres,

dag=dag,

)

create\_table\_task >> predict\_task >> load\_to\_postgres\_task