

Juli 2024

# PROJECT 4

## STEP BY STEP



Prepared By:

**Spiderweb team**

Kelompok 5

**FTDE - 01 DigitalSkola  
Bootcamp**



# TEAM MEMBER

K E L O M P O K   5   F T D E - 0 1



Hafidha  
Harfiana



Humairoh



Hermawan



I Gusti Agung  
Krishna



I Putu Ferry  
Wstika



Fenny  
Inriana

FTDE-01 DigitalSkola



## System Requirements :

1. install kafka docker zookeper dengan menonton video:

<https://www.youtube.com/watch?v=Zq8aMrRnvQE>

Kafka zookeper di docker sudah up and running

```
PS C:\Users\user\.spiderweb_project_4\kafka-stack-docker-compose-master> docker-compose -f ./zk-single-kafka-single.yml ps
time="2024-07-09T11:01:08+03:00" level=warning msg="C:\\\\Users\\\\user\\\\.spiderweb_project_4\\\\kafka-stack-docker-compose-master\\\\zk-single-kafka-single.yml: 'version' is obsolete"
NAME           IMAGE          COMMAND                  SERVICE    CREATED        STATUS       PORTS
kafka1         confluentinc/cp-kafka:7.3.2   "/etc/confluent/dock..."  kafka1    3 minutes ago  Up 3 minutes  0.0.0.0:9092->9092/tcp,
0.0.0.0:9999->9999/tcp, 0.0.0.0:29092->29092/tcp
zool           confluentinc/cp-zookeeper:7.3.2  "/etc/confluent/dock..."  zool      3 minutes ago  Up 3 minutes  2888/tcp, 0.0.0.0:2181->
2181/tcp, 3888/tcp
PS C:\Users\user\.spiderweb_project_4\kafka-stack-docker-compose-master>
```

2. running mongodb sudah dilakukan melalui docker dan git bash

```
mongod
69829491c794

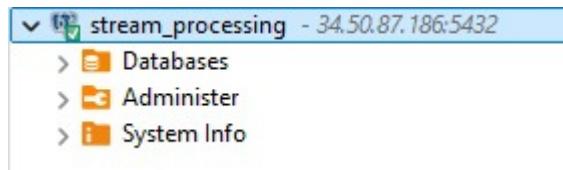
user@DESKTOP-39Q6MSE MINGW64 ~ (master)
$ docker container ls
CONTAINER ID   IMAGE
3a7c6749544c   confluentinc/cp-kafka:7.3.2
5c990889060b   confluentinc/cp-zookeeper:7.3.2
69829491c794   mongodb/mongodb-community-server:5.0-ubuntu2004
                mongo
NAMES          COMMAND                  CREATED        STATUS       PORTS
            "/etc/confluent/dock..."  24 minutes ago  Up 24 minutes  0.0.0.0:9092->9092/tcp
            "/etc/confluent/dock..."  24 minutes ago  Up 24 minutes  2888/tcp, 0.0.0.0:2181->
2181/tcp, 3888/tcp
            "python3 /usr/local/..."  3 weeks ago   Up 3 minutes   0.0.0.0:27017->27017/t
p
```

3. running postgres melalui docker sudah dilakukan

```
postgres
6068f13eaa1a2

user@DESKTOP-39Q6MSE MINGW64 ~ (master)
$ docker container ls
CONTAINER ID   IMAGE
3a7c6749544c   confluentinc/cp-kafka:7.3.2
5c990889060b   confluentinc/cp-zookeeper:7.3.2
69829491c794   mongodb/mongodb-community-server:5.0-ubuntu2004
                mongo
7/tcp
6068f13eaa1a2   postgres
NAMES          COMMAND                  CREATED        STATUS       PORTS
            "/etc/confluent/dock..."  27 minutes ago  Up 27 minutes  0.0.0.0:9092->9092/tcp
            "/etc/confluent/dock..."  27 minutes ago  Up 27 minutes  2888/tcp, 0.0.0.0:2181->
2181/tcp, 3888/tcp
            "python3 /usr/local/..."  3 weeks ago   Up 6 minutes   0.0.0.0:27017->27017/t
p
```

4. Connect database server PostgreSQL sudah dilakukan :



## Langkah - langkah :

1. Running airflow dengan menonton video youtube:

<https://youtu.be/Sva8rDtIWi4?si=sdo0Dbnn2aKJ0Lzp>

Dibawah ini adalah bukti airflow sudah running di docker. saatnya untuk memberikan file DAG python untuk mengkoneksikan dengan server postgre SQL

2. Lalu membuat file DAG python untuk mengorkestrasikan data dari PostgreSQL ke Apache Airflow. File python bernama “load\_postgres\_table.py”. Setelah itu trigger DAG di UI Airflow.

3. Selanjutnya adalah mengerjakan Data Pipeline. Kami running Tes-Model.ipynb melalui Jupyter lokal untuk akses file FraudModel.py

dari folder modelling.

[60]:

```
from modelling import FraudModel
```

4. Sekarang kami akan memakai modelnya ambil dari data transaksi pada file FraudModel.py untuk inputan data

[62]:

```
#parameter inputan data
new_data = {
    'step': 1,
    'type': 'PAYMENT',
    'amount': 9839.64,
    'oldbalanceOrg': 170136.0,
    'newbalanceOrig': 160296.36,
    'oldbalanceDest': 0.0,
    'newbalanceDest': 0.0
}

new_data
```

[62]:

```
{'step': 1,
 'type': 'PAYMENT',
 'amount': 9839.64,
 'oldbalanceOrg': 170136.0,
 'newbalanceOrig': 160296.36,
 'oldbalanceDest': 0.0,
 'newbalanceDest': 0.0}
```

5. Lalu kami akan memakai modelnya ambil dari data transaksi pada file FraudModel.py untuk inputan path

[72]:

```
#parameter inputan path
import os

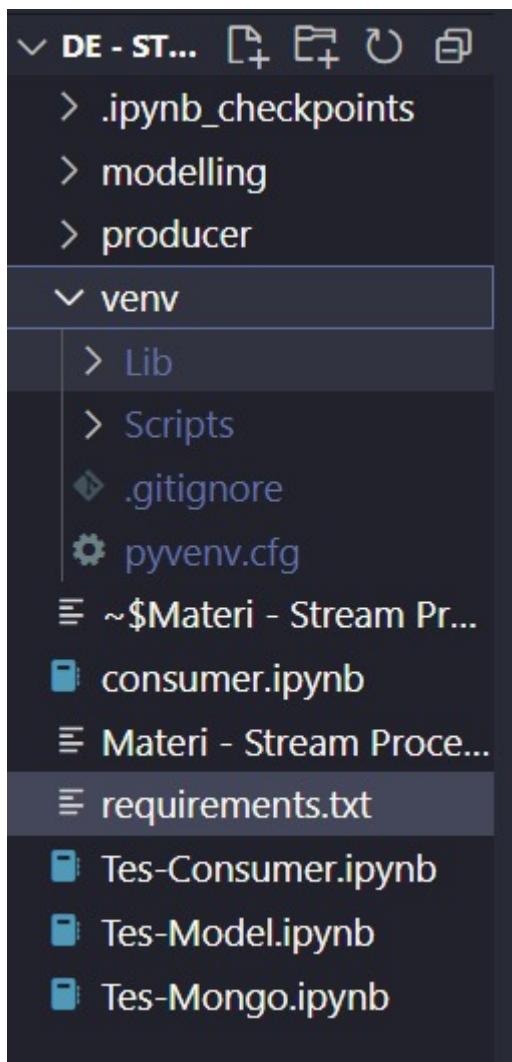
path = os.getcwd()
path = path + "\\modelling\\"
path
```

[72]:

```
'C:\\Users\\user\\DE - STREAM PROCESSING\\modelling\\'
```

dari sini kita dapat melihat bahwa posisi FraudModel.py sudah benar, yaitu pada folder modelling.

Namun, kami mendapatkan sebuah insight bahwa banyak library pada folder ‘DE - STREAM PROCESSING’ yang memiliki versi berbeda dengan library yang dimiliki oleh Jupyter lokal ( Anaconda ) oleh karena itu kami akan membuat virtual environment ( venv ) di VSCode dan melanjutkan codingan di VSCode. Dengan virtual environment, kami bisa mengisolasi versi Python dan library yang digunakan dalam proyek kami tanpa mempengaruhi versi global Python di sistem kami



virtual environment sudah dipasang.

6. Kami melakukan analisis bahwa terdapat 9 Library Python yang digunakan di folder 'DE - STREAM PROCESSING' yang harus kami install terlebih dahulu untuk menjalankan kode.

```
(venv) PS C:\Users\user\Downloads\DE - STREAM PROCESSING
> pip install -r requirements.txt
(venv) PS C:\Users\user\Downloads\DE - STREAM PROCESSING
> pip freeze
```

7. Kami akan running file Tes-Model.ipynb dan kode berhasil dijalankan.
8. Kami masuk ke dalam folder producer dan menemui fakta connection PostgreSQL dengan rincian :  
ip : 34.50.87.186  
db : stream\_processing  
username : ftde01  
password : ftde01!@#

Lalu kami mengakses datadump.ipynb untuk masukin data PostgreSQL ini ke database lokal. Lalu kami akan melakukan running pada file ini dengan hasil

```

    df.to_sql(table_name, engine, if_exists='append', index=False)
    print(f"Data telah dimasukkan ke tabel {table_name}.")
except Exception as e:
    print(f"Terjadi kesalahan: {e}")

if __name__ == "__main__":
    csv_path = "Old_Information.csv"
    data = pd.read_csv(csv_path)

    # Informasi koneksi ke PostgreSQL
    username = "ftde01"
    password = "ftde01!@#"
    host = "34.50.87.186"
    port = "5432"
    database = "stream_processing"
    password = urllib.parse.quote_plus(password)

    # URL koneksi ke PostgreSQL
    db_url = f"postgresql://{username}:{password}@{host}:{port}/{database}"

    table_name = "old_information"
    insert_data_to_postgresql(data, table_name, db_url)

```

[2] ✓ 10.5s

... Data telah dimasukkan ke tabel old\_information.

Data dump berhasil di running dan success.

9. Kami akan membuka folder producer dan mengakses file producer.ipynb untuk melakukan simulasi transaksi yang terjadi di mobile dan terjadi di kafka. Jadi ada producer, lalu lempar data ke kafka dengan nama **ftde01-project4**.

Ketika running producer, terdapat error dengan keterangan

**ModuleNotFoundError: No module named 'kafka.vendor.six.moves'**

Error **ModuleNotFoundError: No module named 'kafka.vendor.six.moves'** terjadi karena Kafka library yang digunakan mencoba mengimpor modul yang seharusnya terdapat dalam versi Kafka yang lebih lama.

Oleh karena itu kami install kafka-python-3.1.2.2 untuk menghindari error ini berdasarkan sumber :

<https://stackoverflow.com/questions/77287622/modulenotfounderror-no-module-named-kafka-vendor-six-moves-in-dockerized-django> . Namun, terdapat error lagi :

**NoBrokersAvailable: NoBrokersAvailable.** Oleh karena itu kami menambahkan api\_version

```

if __name__ == "__main__":
    # Baca data dari file CSV
    data = pd.read_csv('New_Information.csv')
    json_data = data.to_dict(orient='records')

    # Inisialisasi Kafka Producer dengan Bootstrap Server
    producer = KafkaProducer(bootstrap_servers=['localhost:29092'], api_version=(7,3,2), value_serializer=json_serializer)

```

Kami mengubah kode menjadi kode yang menyesuaikan versi Python venv yang terbaru yaitu Python 3.12.2. Running berhasil

```
■ Tes-Model.ipynb ■ producer.ipynb ●
producer > producer.ipynb > ...
+ Code + Markdown | ⌂ Interrupt ⌂ Restart ⌂ Clear All Outputs ⌂ Go To | ⌂ Variables ⌂ Outline ...
Python 3.12.2

# Baca data dari file CSV
data = pd.read_csv('New_Information.csv')
json_data = data.to_dict(orient='records')

# Inisialisasi Kafka Producer dengan Bootstrap Server
producer = KafkaProducer(bootstrap_servers=['localhost:29092'], api_version=(7,3,2), value_serializer=json_serializer)

while True:
    # Looping untuk mengirim setiap data ke Kafka topic
    for data_item in json_data:
        print(data_item)
        producer.send("ftde01-project4", data_item)
        time.sleep(1) # Tunggu 1 detik sebelum mengirim data selanjutnya

[1] 0 1m 55.s Python
...
{"step": 1, "type": "PAYMENT", "amount": 9839.64, "nameOrig": "C1231006815", "newbalanceOrig": 160296.36, "nameDest": "M1979787155", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 1864.28, "nameOrig": "C1666544295", "newbalanceOrig": 19384.72, "nameDest": "M2044282225", "newbalanceDest": 0.0}
{"step": 1, "type": "TRANSFER", "amount": 181.0, "nameOrig": "C1305486145", "newbalanceOrig": 0.0, "nameDest": "C553264065", "newbalanceDest": 0.0}
{"step": 1, "type": "CASH_OUT", "amount": 181.0, "nameOrig": "C846083671", "newbalanceOrig": 0.0, "nameDest": "C38997010", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 11668.14, "nameOrig": "C2848537728", "newbalanceOrig": 29885.86, "nameDest": "M1230701703", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 7817.71, "nameOrig": "C908045638", "newbalanceOrig": 46042.29, "nameDest": "M573487274", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 7107.77, "nameOrig": "C154988899", "newbalanceOrig": 176087.23, "nameDest": "M408069119", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 7861.64, "nameOrig": "C1912850431", "newbalanceOrig": 168225.59, "nameDest": "M633326333", "newbalanceDest": 0.0}
{"step": 1, "type": "PAYMENT", "amount": 4024.36, "nameOrig": "C1265012928", "newbalanceOrig": 0.0, "nameDest": "M1176932104", "newbalanceDest": 0.0}
{"step": 1, "type": "DEBIT", "amount": 533.77, "nameOrig": "C712410124", "newbalanceOrig": 36382.23, "nameDest": "C195600860", "newbalanceDest": 40348.79}
```

10. Lalu kami akan mengambil data ini di Kafka, sehingga kami mengakses notebook consumer.ipynb dan notebook Tes-Consumer.ipynb. Pada consumer.ipynb kami menyesuaikan kode agar sesuai dengan versi Python terbaru 3.12.2. Dan kode blok ini berhasil dijalankan ( data tersebut berhasil ditangkap )

```
import json
from kafka import KafkaConsumer

if __name__ == "__main__":
    consumer = KafkaConsumer("ftde01-project4", bootstrap_servers='localhost:29092')
    print("Starting the consumer")

    for msg in consumer:
        print(f"Records = {json.loads(msg.value)}")

        # Parsing pesan Kafka
        data = json.loads(msg.value)

[1] ⓘ 194s Python
```

Pada saat ini kami akan memberhentikan producer.ipynb untuk mendapat data hasil producer yang secukupnya.

```
    untuk mengirim setiap data ke Kafka topic
    item in json_data:
        data_item)
        er.send("ftde01-project4", data_item)
        sleep(1) # Tunggu 1 detik sebelum mengirim data selanjutnya
```

```

import json
from kafka import KafkaConsumer

if __name__ == "__main__":
    consumer = KafkaConsumer("ftde01-project4", bootstrap_servers='localhost:9092')
    print("Starting the consumer")

    for msg in consumer:
        print(f"Records = {json.loads(msg.value)}")

        # Parsing pesan Kafka
        data = json.loads(msg.value)

```

[1] 5m 0.2s Python

... Starting the consumer

```

Records = {'step': 1, 'type': 'PAYMENT', 'amount': 1795.67, 'nameOr...
Records = {'step': 1, 'type': 'PAYMENT', 'amount': 3199.06, 'nameOr...
Records = {'step': 1, 'type': 'PAYMENT', 'amount': 10265.17, 'nameC...
Records = {'step': 1, 'type': 'PAYMENT', 'amount': 9029.12, 'nameOr...
Records = {'step': 1, 'type': 'PAYMENT', 'amount': 7600.57, 'nameOr...

```

11. Kami mengeksekusi setiap cell di Tes-Consumer. Kami berhasil running di kedua kode blok ini :

[2] data  
✓ 0.0s

... {'step': 1,  
'type': 'TRANSFER',  
'amount': 522264.81,  
'nameOrig': 'C1927963027',  
'newbalanceOrig': 0.0,  
'nameDest': 'C1234776885',  
'newbalanceDest': 2025098.66}

[3] ✓ 2.4s

```

import pandas as pd
import urllib.parse

from sqlalchemy import create_engine

# Informasi koneksi ke PostgreSQL
username = "ftde01"
password = "ftde01!@#"
host = "34.50.87.186"
port = "5432"
database = "stream_processing"
password = urllib.parse.quote_plus(password)

# URL koneksi ke PostgreSQL
db_url = f"postgresql://{{username}}:{{password}}@{{host}}:{{port}}/{{database}}"
engine = create_engine(db_url)

```

Namun tidak dengan

```
✓ df = pd.read_sql_query('SELECT * FROM old_information;', engine)
  df.head(5)

⊗ 0.1s
```

karena error : `AttributeError: 'Engine' object has no attribute 'cursor'`. Maka dari itu kami mencoba solusi lain dengan mencari solusi pada stack

overflow dan kami sudah menemukan solusinya lalu running berhasil :

```
▷ connection = engine.raw_connection()
  df = pd.read_sql_query('SELECT * FROM old_information;', con=connection)
  df.head(5)

[12] ✓ 8.6s

... :\Users\user\AppData\Local\Temp\ipykernel_21712\3931741001.py:2: UserWarning: pa
  df = pd.read_sql_query('SELECT * FROM old_information;', con=connection)

...      nameOrig  oldbalanceOrg      nameDest  oldbalanceDest
0   C1231006815        170136.0    M1979787155           0.0
1   C1666544295        21249.0     M2044282225           0.0
2   C1305486145         181.0      C553264065           0.0
3   C840083671          181.0      C38997010           21182.0
4   C2048537720        41554.0     M1230701703           0.0
```

12. Consumer sudah berhasil kami ambil, kami akan melakukan join dengan mengubah data producer dalam bentuk dict ke dataframe.

### Join Data From Producer Kafka and Database PostgreSQL

```
▷ produder = pd.DataFrame([data])
  produder

[8] ✓ 0.0s

...      step  type  amount  nameOrig  newbalanceOrig  nameDest  newbalanceDest
0       1  DEBIT  4874.49  C811207775           0.0  C1971489295           0.0

▷ data = produder.merge(df, how='inner', on=['nameOrig', 'nameDest'])
  predict = data.drop(['nameOrig', 'nameDest'], axis=1)
  predict = predict.to_dict('index')[0]
  predict

[9] ✓ 0.0s

... {'step': 1,
  'type': 'DEBIT',
  'amount': 4874.49,
  'newbalanceOrig': 0.0,
  'newbalanceDest': 0.0,
  'oldbalanceOrg': 153.0,
  'oldbalanceDest': 253104.0}
```

Join data sudah berhasil dilakukan, lalu kami akan melakukan prediksi Fraud :

## Precit Fraud Detection

```
[10]    from modelling import FraudModel
[10]    ✓ 0.0s

▷ ▾
#parameter inputan path
import os

path = os.getcwd()
path = path + "\\modelling\\"
path
[11]    ✓ 0.0s
...
... 'c:\\\\Users\\\\user\\\\.spiderweb_project_4\\\\DE - STREAM PROCESSING\\\\modelling\\'

[12]    result = FraudModel.runModel(predict, path)
result
[12]    ✓ 34.7s
...
... c:\\\\Users\\\\user\\\\OneDrive\\\\Masa\\\\üstü\\\\repo_8\\\\Lib\\\\site-packages\\\\sklearn\\\\base.py:376: InconsistentVersionWarning:\\nhttps://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
...     warnings.warn(warning, stacklevel=2)
```

Prediksi berhasil dilakukan :

```
▷ ▾
data['predict'] = result
data
[13]    ✓ 0.0s
...
...   step type amount nameOrig newbalanceOrig nameDest newbalanceDest oldbalanceOrg oldbalanceDest predict
...   0   1 DEBIT 4874.49 C811207775          0.0 C1971489295          0.0      153.0    253104.0 White List
...   1   1 DEBIT 4874.49 C811207775          0.0 C1971489295          0.0      153.0    253104.0 White List
...   2   1 DEBIT 4874.49 C811207775          0.0 C1971489295          0.0      153.0    253104.0 White List
...   3   1 DEBIT 4874.49 C811207775          0.0 C1971489295          0.0      153.0    253104.0 White List
```

13. Langkah terakhir adalah insert ke MongoDB kami set terlebih dahulu username dan passwordnya :

```
test> use admin
switched to db admin
admin> db.createUser({ user: "admin", pwd: passwordPrompt(), roles: [{rrole: "root", db:"admin"}]})
Enter password
*****{ ok: 1 }
admin>
```

```
from pymongo import MongoClient

# Mengatur koneksi ke MongoDB
mongo_client = MongoClient("mongodb://admin:password@localhost:2717")

db = mongo_client["ftde01"]
collection = db["project4-collection"]

if isinstance(data.to_dict('index')[0], list):
    collection.insert_many(data.to_dict('index')[0])
else:
    collection.insert_one(data.to_dict('index')[0])
    print("Data telah disimpan ke MongoDB")
[25] ✓ 0.1s
...
... Data telah disimpan ke MongoDB
```

Kami sudah menarik datanya dari MongoDB

```
▷ mongo_client = MongoClient("mongodb://admin:password@localhost:2717/")
  db = mongo_client["ftde01"]
  collection = db["project4-collection"]

  # Membaca data dari MongoDB ke dalam list of dictionaries
  data_from_mongo = list(collection.find())

  # Membaca data ke dalam DataFrame
  df = pd.DataFrame(data_from_mongo)
  df
[26] ✓ 0.0s
...
...      _id  step  type   amount  nameOrig  newbalanceOrig  nameDest  newbalanceDest  oldbalanceOrg  oldbalanceDest  predict
0  66900df2c091f21df1b32d66     1  DEBIT  4874.49  C811207775           0.0  C1971489295           0.0        153.0      253104.0  White List
```

Dari data ini dipindahkan lagi ke dalam dalam google sheet. lalu visualisasikan ke google studio.

Thank  
you

