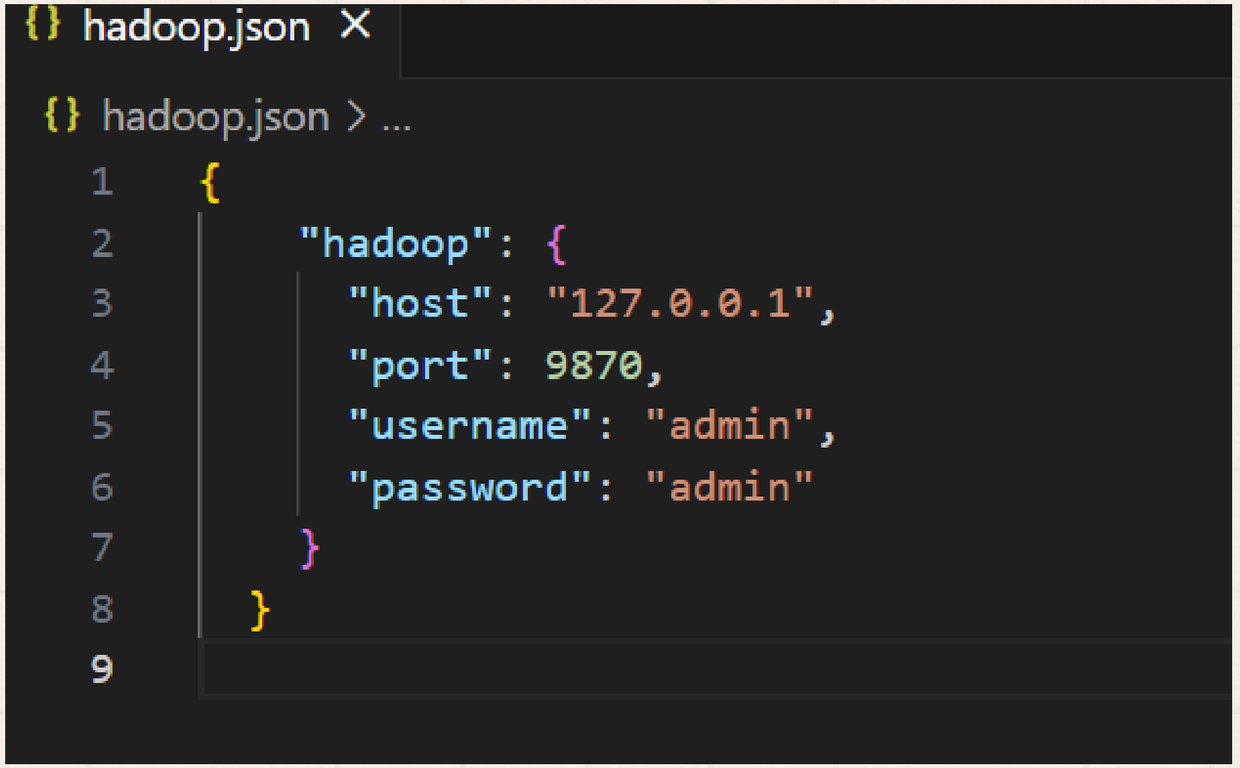
DATA ANALYTICS WITH HADOOP PROJECT

Kelompok 8 - INFINITY

FLOW DIAGRAM ETL

MENAMBAH KONEKSI JSON HADOOP



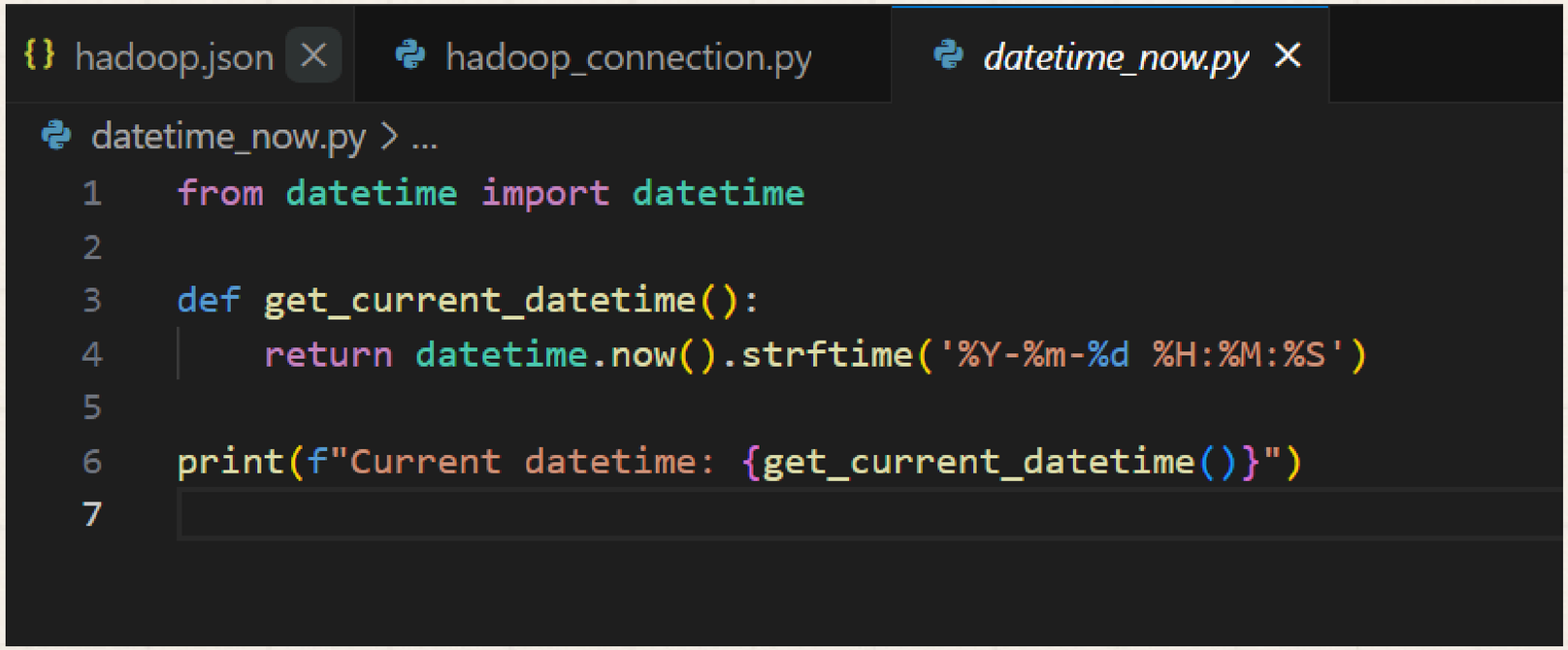
MEMBUAT SCRIPT KONEKSI HADOOP

hadoop.json | hadoop_connection.py

hadoop_connection.py > ...

```python
import json
import requests
from hdfs import InsecureClient

# Load Hadoop connection details from json
with open('hadoop.json') as f:
    hadoop_config = json.load(f)['hadoop']

client = InsecureClient(f"http://{hadoop_config['host']}:{hadoop_config['port']}", user=hadoop_config['username'])

# Example function to upload file to HDFS
def upload_to_hdfs(local_path, hdfs_path):
    client.upload(hdfs_path, local_path)
    print(f"Uploaded {local_path} to {hdfs_path}")

# Usage
upload_to_hdfs('orders.csv', '/user/orders.csv')

```

MENDEFINISIKAN DATETIME NOW



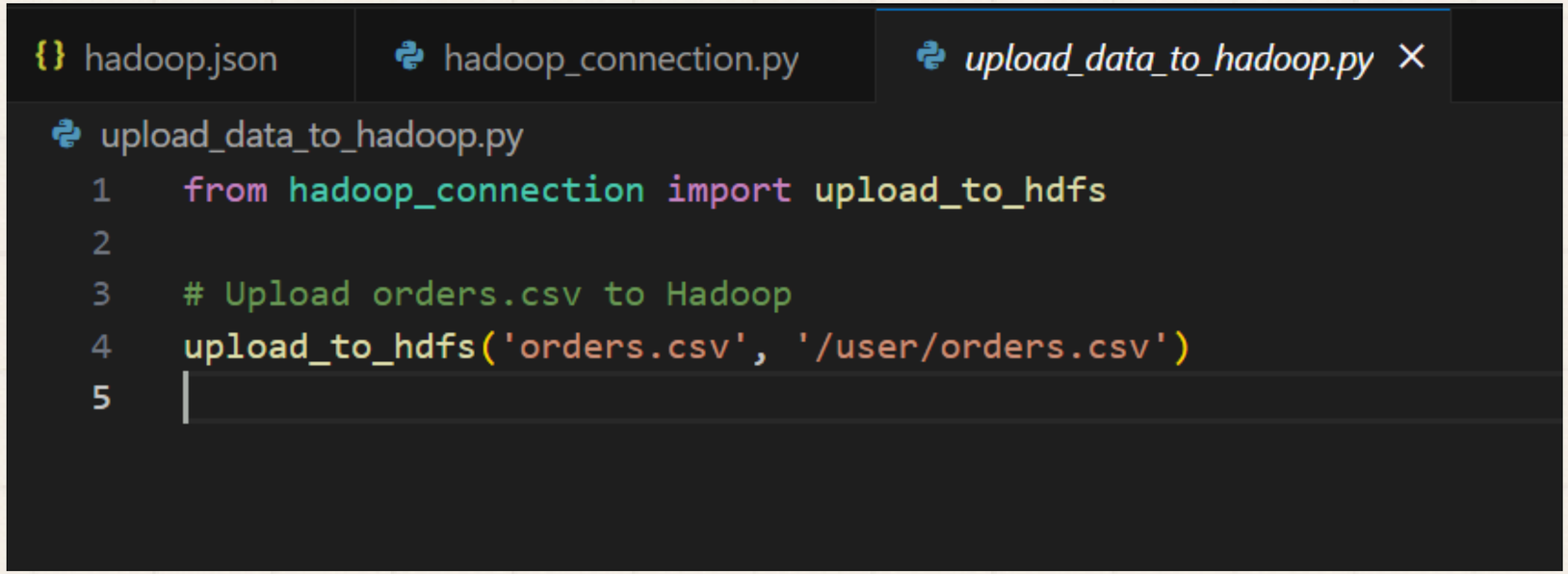
MEMBUAT DATA DUMMY

hadoop.json | hadoop_connection.py | *dummy_data.py*

dummy_data.py > ...

```python
import csv
import datetime
import random

# Buat data dummy
header = ['order_id', 'order_date', 'user_id', 'payment_name', 'shipper_name', 'order_price', 'order_discount', 'vou
data = []

for i in range(1, 1001):
    order_date = datetime.datetime(2023, random.randint(1, 12), random.randint(1, 28)).strftime('%Y-%m-%d')
    data.append([i, order_date, random.randint(1, 100), 'Credit Card', 'FedEx', random.uniform(10.0, 500.0), random.

# Tulis data dummy ke CSV
with open('orders.csv', 'w', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(header)
    writer.writerows(data)

print("Dummy data created and saved to orders.csv")
```

MEMBUAT SCRIPT UPLOAD DATA KE HADOOP



MEMBUAT SCRIPT TRANSFORM DENGAN MAPREDUCE

hadoop.json | hadoop_connection.py | MapReduce.py

MapReduce.py > ...

```python
from mrjob.job import MRJob
from mrjob.step import MRStep
import csv

cols = 'order_id,order_date,user_id,payment_name,shipper_name,order_price,order_discount,voucher_name,voucher_price

def csv_readline(line):
    for row in csv.reader([line]):
        return row

class OrderDateCount(MRJob):
    def steps(self):
        return [
            MRStep(mapper=self.mapper, reducer=self.reducer),
            MRStep(reducer=self.sort)
        ]

    def mapper(self, _, line):
        row = dict(zip(cols, csv_readline(line)))
        if row['order_id'] != 'order_id':
            yield row['order_date'][0:7], 1

    def reducer(self, key, values):
        yield None, (key, sum(values))

    def sort(self, key, values):
        data = []
        for order_date, order_count in values:
            data.append((order_date, order_count))
        data.sort()

        for order_date, order_count in data:
            yield order_date, order_count
```

OUTPUT

```
"2023-01" 85

"2023-02" 78

"2023-03" 91
```