

# DATA ANALYTICS WITH HADOOP PROJECT

Kelompok 8 - INFINITY



# FLOW DIAGRAM ETL



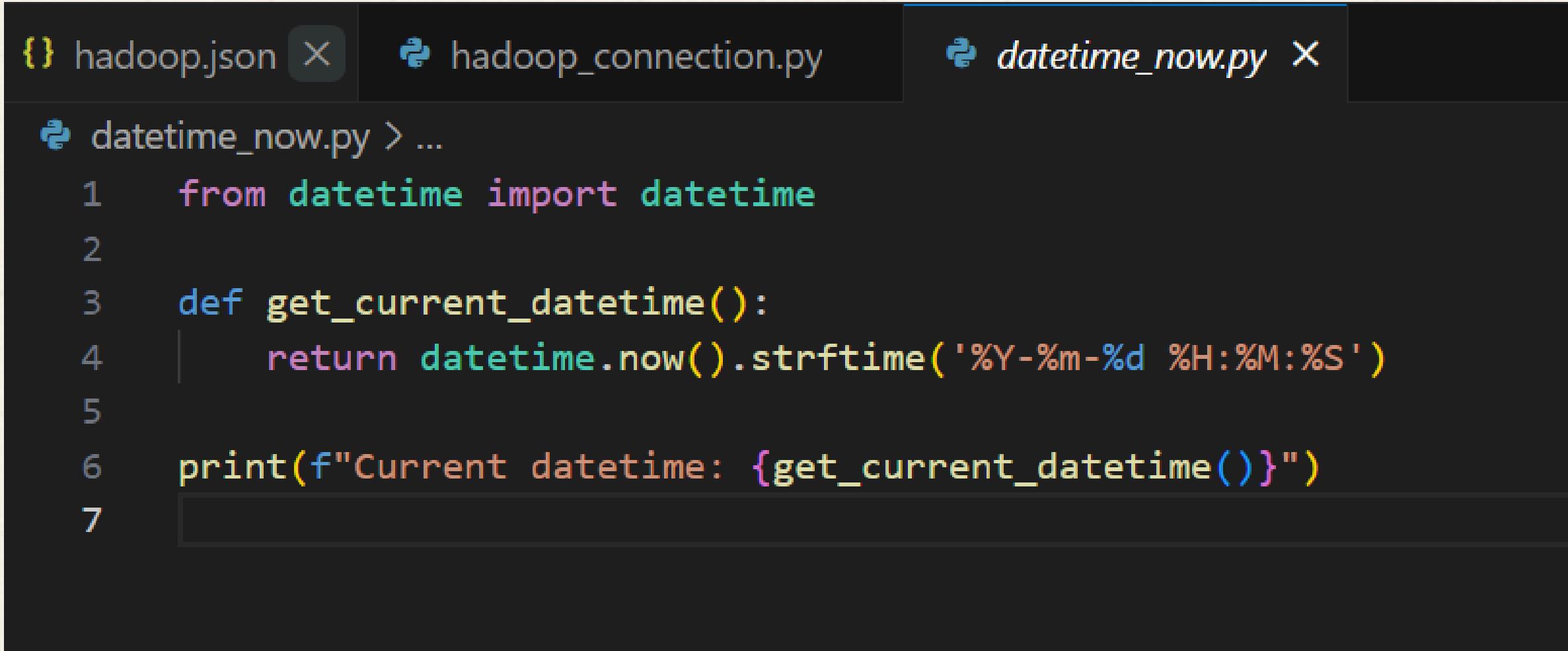
# MENAMBAH KONEKSI JSON HADOOP

```
{} hadoop.json X  
{} hadoop.json > ...  
1 {  
2     "hadoop": {  
3         "host": "127.0.0.1",  
4         "port": 9870,  
5         "username": "admin",  
6         "password": "admin"  
7     }  
8 }  
9
```

# MEMBUAT SCRIPT KONEKSI HADOOP

```
{} hadoop.json      ⌂ hadoop_connection.py X  
⠃ hadoop_connection.py > ...  
1  import json  
2  import requests  
3  from hdfs import InsecureClient  
4  
5  # Load Hadoop connection details from json  
6  with open('hadoop.json') as f:  
7      hadoop_config = json.load(f)['hadoop']  
8  
9  client = InsecureClient(f"http://{hadoop_config['host']}:{hadoop_config['port']}", user=hadoop_config['username'])  
10  
11 # Example function to upload file to HDFS  
12 def upload_to_hdfs(local_path, hdfs_path):  
13     client.upload(hdfs_path, local_path)  
14     print(f"Uploaded {local_path} to {hdfs_path}")  
15  
16 # Usage  
17 upload_to_hdfs('orders.csv', '/user/orders.csv')  
18
```

# MENDEFINISIKAN DATETIME NOW



The image shows a code editor interface with three tabs at the top: 'hadoop.json' (closed), 'hadoop\_connection.py' (closed), and 'datetime\_now.py'. The 'datetime\_now.py' tab is active and displays the following Python code:

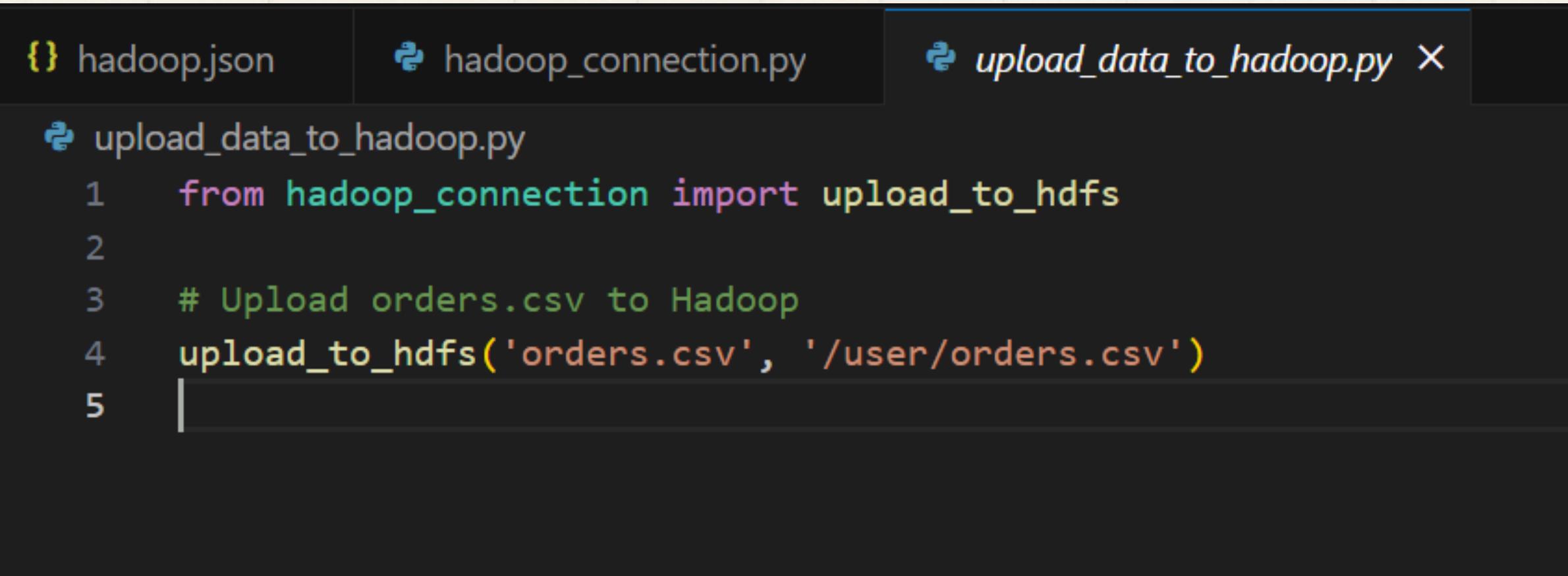
```
1 from datetime import datetime
2
3 def get_current_datetime():
4     return datetime.now().strftime('%Y-%m-%d %H:%M:%S')
5
6 print(f"Current datetime: {get_current_datetime()}")
7
```

# MEMBUAT DATA DUMMY

```
{} hadoop.json      ⚡ hadoop_connection.py      ⚡ dummy_data.py X
⚡ dummy_data.py > ...

1  import csv
2  import datetime
3  import random
4
5  # Buat data dummy
6  header = ['order_id', 'order_date', 'user_id', 'payment_name', 'shipper_name', 'order_price', 'order_discount', 'vo
7  data = []
8
9  for i in range(1, 1001):
10     order_date = datetime.datetime(2023, random.randint(1, 12), random.randint(1, 28)).strftime('%Y-%m-%d')
11     data.append([i, order_date, random.randint(1, 100), 'Credit Card', 'FedEx', random.uniform(10.0, 500.0), random.
12
13 # Tulis data dummy ke CSV
14 with open('orders.csv', 'w', newline='') as f:
15     writer = csv.writer(f)
16     writer.writerow(header)
17     writer.writerows(data)
18
19 print("Dummy data created and saved to orders.csv")
20
```

# MEMBUAT SCRIPT UPLOAD DATA KE HADOOP



The image shows a code editor interface with three tabs:

- {} hadoop.json
- py hadoop\_connection.py
- py *upload\_data\_to\_hadoop.py* X

The *upload\_data\_to\_hadoop.py* file contains the following Python code:

```
1 from hadoop_connection import upload_to_hdfs
2
3 # Upload orders.csv to Hadoop
4 upload_to_hdfs('orders.csv', '/user/orders.csv')
5
```

# MEMBUAT SCRIPT TRANSFORM DENGAN MAPREDUCE

```
{} hadoop.json      hadoop_connection.py      MapReduce.py X  
MapReduce.py > ...  
1  from mrjob.job import MRJob  
2  from mrjob.step import MRStep  
3  import csv  
4  |  
5  cols = 'order_id,order_date,user_id,payment_name,shipper_name,order_price,order_discount,voucher_name,voucher_price'  
6  |  
7  def csv_readline(line):  
8  |  for row in csv.reader([line]):  
9  |    return row  
10 |  
11 class OrderDateCount(MRJob):  
12   def steps(self):  
13     return [  
14       |  MRStep(mapper=self.mapper, reducer=self.reducer),  
15       |  MRStep(reducer=self.sort)  
16     ]  
17 |  
18   def mapper(self, _, line):  
19     |  row = dict(zip(cols, csv_readline(line)))  
20     |  if row['order_id'] != 'order_id':  
21     |    yield row['order_date'][0:7], 1  
22 |  
23   def reducer(self, key, values):  
24     |  yield None, (key, sum(values))  
25 |  
26   def sort(self, key, values):  
27     |  data = []  
28     |  for order_date, order_count in values:  
29     |    data.append((order_date, order_count))  
30     |  data.sort()  
31 |  
32   for order_date, order_count in data:  
33     |  yield order_date, order_count
```

# OUTPUT

"2023-01" 85

"2023-02" 78

"2023-03" 91