

**NAMA : FIDELIA PING**  
**NIM : 245410012**  
**KELAS : INFORMATIKA 1**

## MODUL 9

### IMPLEMENTASI PUB/SUB (BROADCASTING)

#### DASAR TEORI

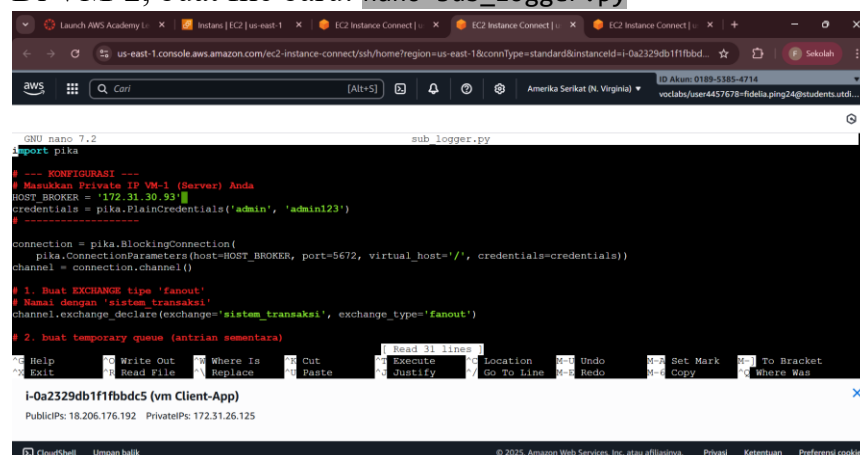
Publish/Subscribe (Pub/Sub) adalah sebuah messaging pattern di mana pengirim pesan (publisher) tidak mengirim pesan langsung kepada penerima tertentu, tetapi mengirimkannya ke sebuah channel, topic, atau exchange. Penerima pesan (subscriber) kemudian menerima pesan dari channel/topic tersebut sesuai dengan langganannya. Pada model ini, publisher dan subscriber tidak saling mengetahui satu sama lain, sehingga komunikasi menjadi lebih fleksibel, terdesentralisasi, dan skalabel.

#### PRAKTIK

##### BAGIAN 2: MEMBUAT SUBSCRIBER (PENERIMA)

##### Langkah 1: Buat Script Logger (sub\_logger.py)

Di VM-2, buat file baru: `nano sub_logger.py`



```
GNU nano 7.2 sub_logger.py
import pika

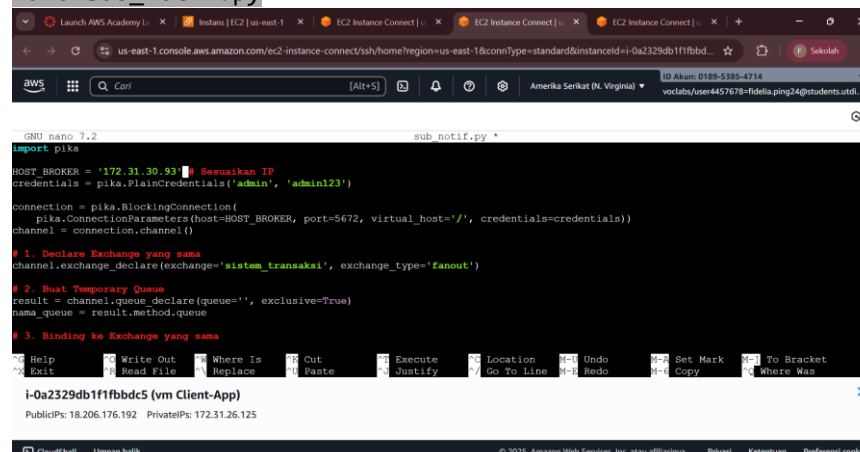
# --- KONFIGURASI ---
# Masukkan Private IP VM-1 (Server) Anda
HOST_BROKER = '172.31.30.93'
credentials = pika.PlainCredentials('admin', 'admin123')

# 1. Buat EXCHANGE tipe 'fanout'
# Rinci dengan 'sistem_transaksi'
channel.exchange_declare(exchange='sistem_transaksi', exchange_type='fanout')

# 2. buat temporary queue (antrian sementara)
channel.queue_declare(queue='antrian_sementara')
```

##### Langkah 2: Buat Script Notifikasi (sub\_notif.py)

`nano sub_notif.py`



```
GNU nano 7.2 sub_notif.py
import pika

HOST_BROKER = '172.31.30.93'
credentials = pika.PlainCredentials('admin', 'admin123')

connection = pika.BlockingConnection(
    pika.ConnectionParameters(host=HOST_BROKER, port=5672, virtual_host='/', credentials=credentials))
channel = connection.channel()

# 1. Declare Exchange yang sama
channel.exchange_declare(exchange='sistem_transaksi', exchange_type='fanout')

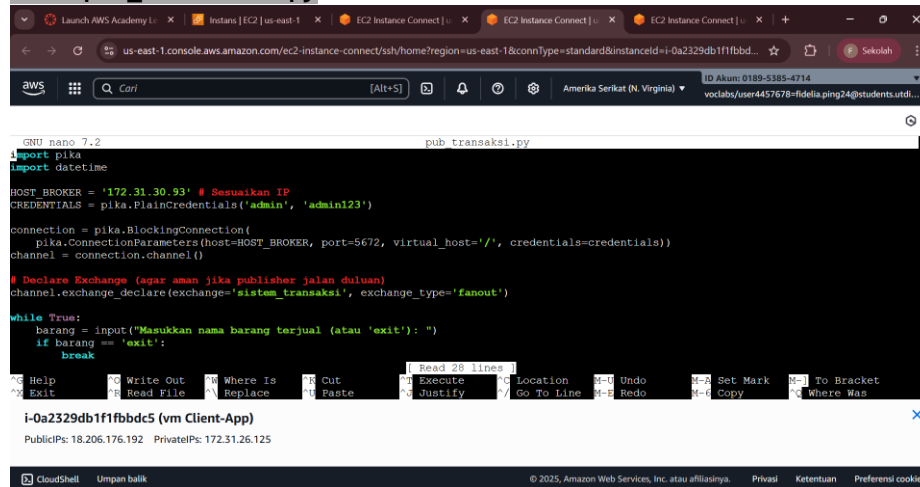
# 2. Buat Temporary Queue
result = channel.queue_declare(queue='antrian_sementara', exclusive=True)
nama_queue = result.method.queue

# 3. Binding ke Exchange yang sama
channel.queue_bind(exchange='sistem_transaksi', queue=nama_queue)
```

## BAGIAN 3: MEMBUAT PUBLISHER (PENGIRIM)

### Langkah 3: Buat Script Transaksi (pub\_transaksi.py)

nano pub\_transaksi.py



```
#!/usr/bin/env python3
import pika
import datetime

HOST_BROKER = '172.31.30.93' # Sesuaikan IP
CREDENTIALS = pika.PlainCredentials('admin', 'admin123')

connection = pika.BlockingConnection(
    pika.ConnectionParameters(host=HOST_BROKER, port=5672, virtual_host='/', credentials=CREDENTIALS))
channel = connection.channel()

# Declare Exchange (agar aman jika publisher jalan duluan)
channel.exchange_declare(exchange='sistem_transaksi', exchange_type='fanout')

while True:
    barang = input("Masukkan nama barang terjual (atau 'exit'): ")
    if barang == 'exit':
        break

i-0a2329db1f1fbbdc5 (vm Client-App)
PublicIPs: 18.206.176.192 PrivateIPs: 172.31.26.125
```

## BAGIAN 4: UJI COBA

### 1. Terminal 1 (Logger):

Jalankan: `python3 sub_logger.py`

(Akan muncul: Waiting...)

```
ubuntu@ip-172-31-26-125:~$ python3 sub_notif.py
[*] SISTEM NOTIFIKASI siap. Menunggu alert...
```

i-0a2329db1f1fbbdc5 (vm Client-App)

PublicIPs: 18.206.176.192 PrivateIPs: 172.31.26.125

### 2. Terminal 2 (Notif):

Jalankan: `python3 sub_notif.py`

(Akan muncul: Waiting...)

```
Last login: Thu Dec 4 08:16:04 2025 from 18.206.107.29
ubuntu@ip-172-31-26-125:~$ python3 sub_logger.py
[*] SISTEM LOGGER siap. Menunggu data transaksi...
```

i-0a2329db1f1fbbdc5 (vm Client-App)

PublicIPs: 18.206.176.192 PrivateIPs: 172.31.26.125

### 3. Terminal 3 (Publisher):

Jalankan: `python3 pub_transaksi.py`

- Ketik: Laptop Gaming -> Enter.
- Ketik: Mouse Wireless -> Enter.

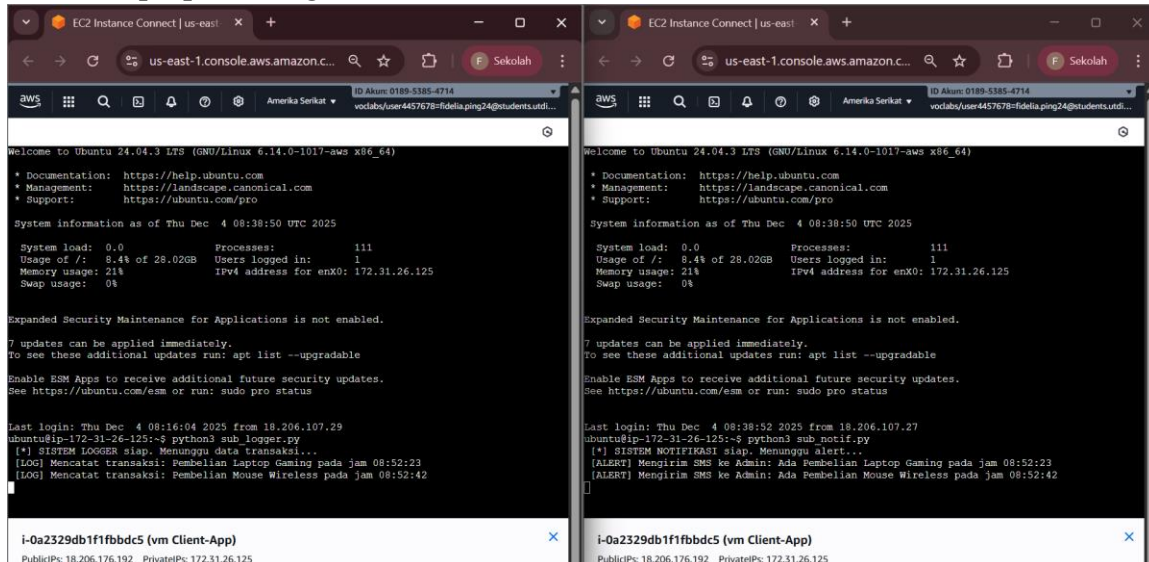
```
ubuntu@ip-172-31-26-125:~$ python3 pub_transaksi.py
Masukkan nama barang terjual (atau 'exit'): Laptop Gaming
[x] Data terkirim ke sentral: 'Laptop Gaming'
Masukkan nama barang terjual (atau 'exit'): Mouse Wireless
[x] Data terkirim ke sentral: 'Mouse Wireless'
Masukkan nama barang terjual (atau 'exit'):
```

i-0a2329db1f1fbbdc5 (vm Client-App)

PublicIPs: 18.206.176.192 PrivateIPs: 172.31.26.125

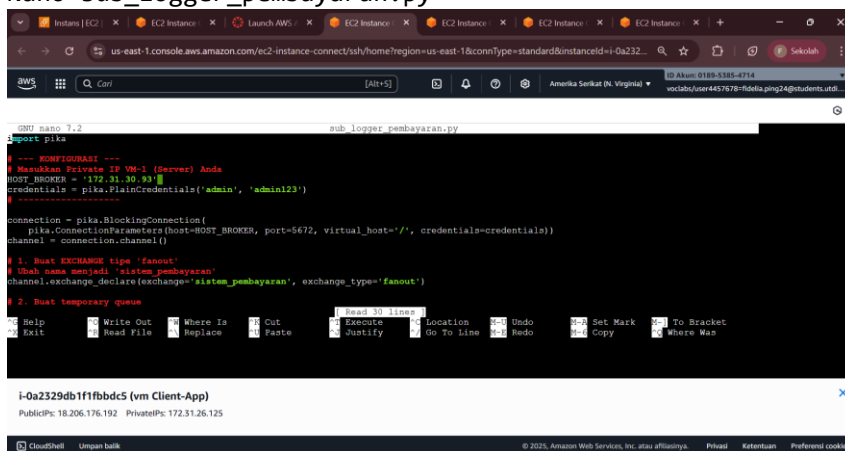
Lihat Terminal 1 dan Terminal 2 secara bersamaan.

- Terminal 1 akan muncul: [LOG] Mencatat transaksi: Pembelian Laptop Gaming...
- Terminal 2 akan muncul: [ALERT] Mengirim SMS ke Admin: Ada Pembelian Laptop Gaming...

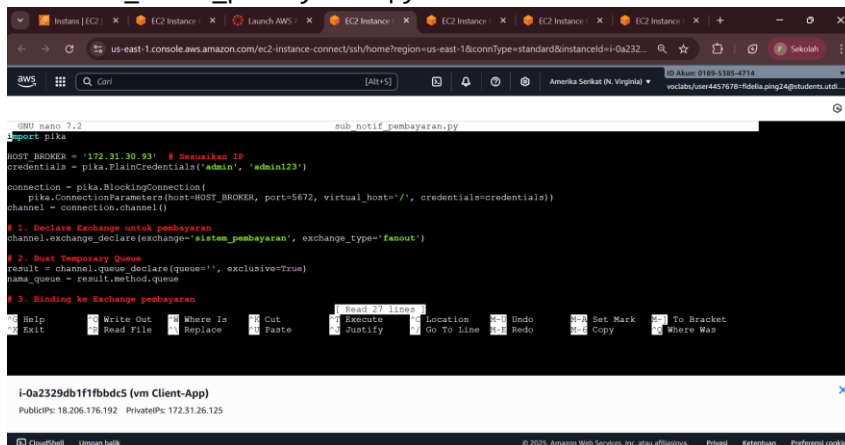


## TUGAS

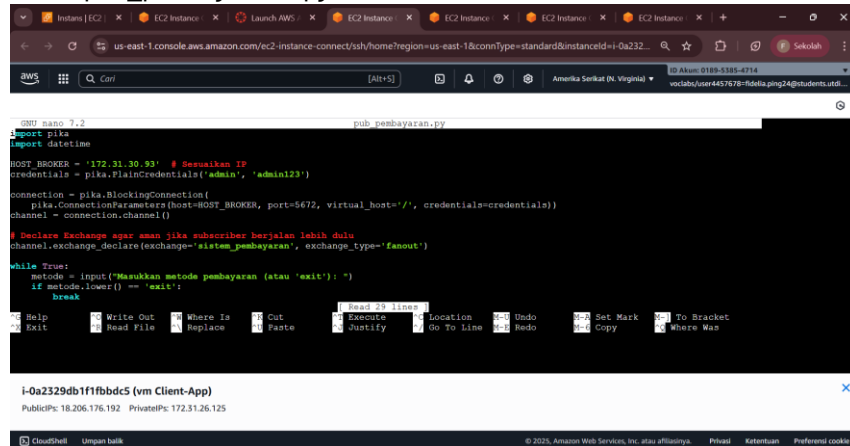
Nano sub\_logger\_pembayaran.py



nano sub\_notif\_pembayaran.py



## nano pub\_pembayaran.py



```
GNU nano 2.9.2 pub_pembayaran.py
import pika
import datetime

HOST_BROKER = '172.31.30.83' # Sesuaikan IP
credentials = pika.PlainCredentials('admin', 'admin123')

connection = pika.BlockingConnection(
    pika.ConnectionParameters(host=HOST_BROKER, port=5672, virtual_host='/', credentials=credentials))
channel = connection.channel()

# Declare Exchange agar aman jika subscriber berjalan lebih dulu
channel.exchange_declare(exchange='sistem_pembayaran', exchange_type='fanout')

while True:
    metode = input("Masukkan metode pembayaran (atau 'exit'): ")
    if metode.lower() == 'exit':
        break

    # Read 25 lines
    # Execute
    # Justify
    # Location
    # Go To Line
    # Undo
    # Redo
    # Set Mark
    # To Bracket
    # Where Was
```

## python3 sub\_logger\_pembayaran.py

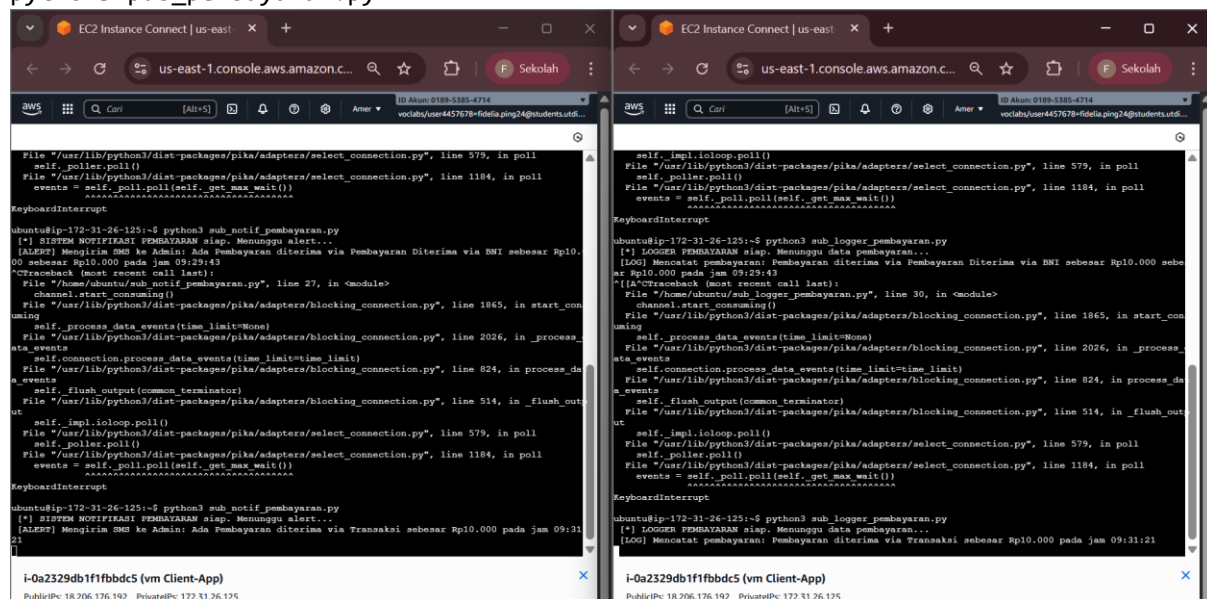
```
ubuntu@ip-172-31-26-125:~$ python3 pub_pembayaran.py
Masukkan metode pembayaran (atau 'exit'): Transaksi
Masukkan nominal pembayaran: Rp10.000
[x] Data pembayaran terkirim: 'Pembayaran diterima via Transaksi sebesar Rp10.000 pada jam 09:31:21'
Masukkan metode pembayaran (atau 'exit'):
```

## i-0a2329db1f1fbbdc5 (vm Client-App)

PublicIPs: 18.206.176.192 PrivateIPs: 172.31.26.125

## python3 sub\_notif\_pembayaran.py

## python3 pub\_pembayaran.py



```
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 579, in poll
self._poller.poll()
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 1184, in poll
events = self._poll.poll(self._get_max_wait())
KeyboardInterrupt

ubuntu@ip-172-31-26-125:~$ python3 sub_notif_pembayaran.py
[*] SISTEM NOTIFIKASI PEMBAYARAN siap. Menunggu alert...
[ALERT] Mengirim SMS ke Admin: Ada Pembayaran diterima via Transaksi sebesar Rp10.000 pada jam 09:31:21
[*] Traceback (most recent call last):
File "/home/ubuntu/sub_notif_pembayaran.py", line 27, in <module>
channel.start_consuming()
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 1865, in start_consuming
self._process_data_events(time_limit=None)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 2026, in _process_data_events
self.connection.process_data_events(time_limit=time_limit)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 824, in process_data_events
self._flush_output(common_terminator)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 514, in _flush_output
self._impl.ioloop.poll()
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 579, in poll
self._poller.poll()
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 1184, in poll
events = self._poll.poll(self._get_max_wait())
KeyboardInterrupt

ubuntu@ip-172-31-26-125:~$ python3 sub_logger_pembayaran.py
[*] LOGGER PEMBAYARAN siap. Menunggu data pembayaran...
[LOG] Mencatat pembayaran: Pembayaran diterima via Transaksi sebesar Rp10.000 pada jam 09:31:21
[*] Traceback (most recent call last):
File "/home/ubuntu/sub_logger_pembayaran.py", line 30, in <module>
channel.start_consuming()
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 1865, in start_consuming
self._process_data_events(time_limit=None)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 2026, in _process_data_events
self.connection.process_data_events(time_limit=time_limit)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 824, in process_data_events
self._flush_output(common_terminator)
File "/usr/lib/python3/dist-packages/pika/adapters/blocking_connection.py", line 514, in _flush_output
self._impl.ioloop.poll()
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 579, in poll
self._poller.poll()
File "/usr/lib/python3/dist-packages/pika/adapters/select_connection.py", line 1184, in poll
events = self._poll.poll(self._get_max_wait())
KeyboardInterrupt

ubuntu@ip-172-31-26-125:~$ python3 sub_logger_pembayaran.py
[*] LOGGER PEMBAYARAN siap. Menunggu data pembayaran...
[LOG] Mencatat pembayaran: Pembayaran diterima via Transaksi sebesar Rp10.000 pada jam 09:31:21
```

## KESIMPULAN

Saya membuat tiga komponen utama, yaitu publisher, logger, dan notifikasi. Publisher bertugas mengirimkan pesan terkait transaksi atau pembayaran, sedangkan subscriber logger mencatat pesan tersebut, dan subscriber notifikasi menampilkan peringatan seolah mengirimkan SMS atau notifikasi kepada admin. Dari hasil uji coba, saya melihat bahwa setiap pesan yang saya kirimkan melalui publisher dapat langsung diterima oleh kedua subscriber tanpa adanya konflik atau keterikatan langsung antar komponen.