Message Queues Made Easy with Python and RabbitMQ

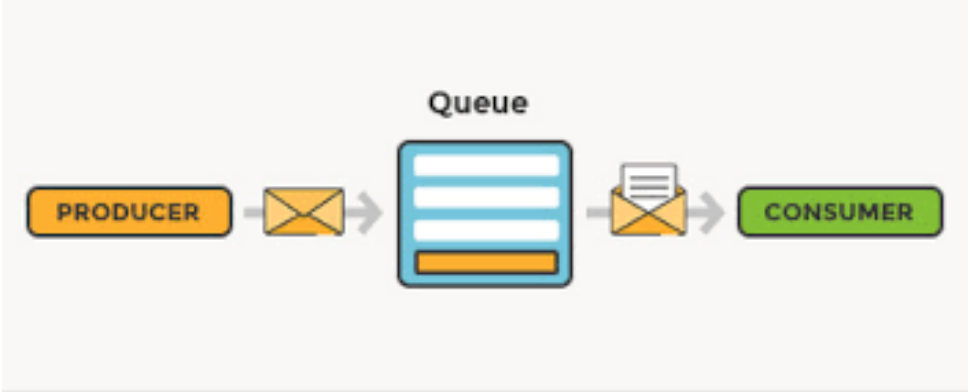
Ahtesham Zaidi
Software Engineer @L&T Technology Services (R&D)

Agenda

- Introduction to Messaging Queues
- Message Brokers
- Traditional Approaches: Drawbacks
- Advantages of Messaging Queues
- Use Cases for Messaging Queues
- Types of message broker
- RabbitMQ
- Code Walkthrough
- Demo

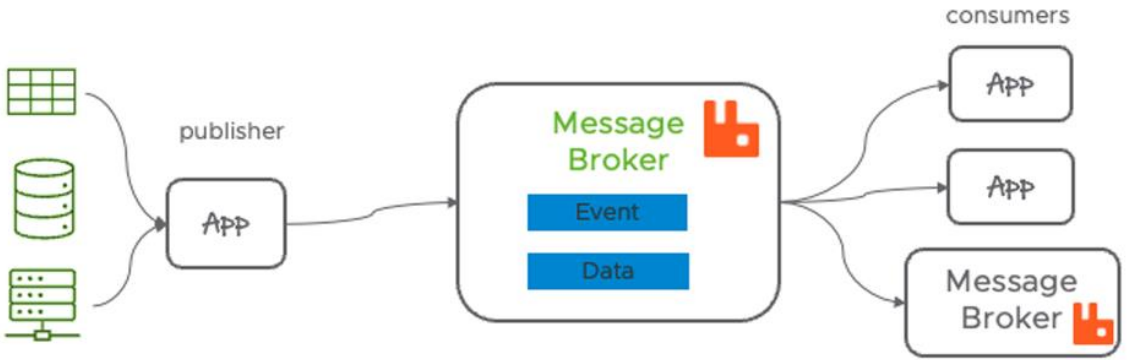
Introduction: Messaging Queues

- Act as a Buffer.
- Message retrieval & process.
- Asynchronous communication.



Message Brokers

- Centralized Hub
- Routing
- Transformation

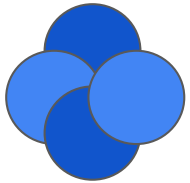


- Example, a message would be the details of an order that could be added to the message queue and then later processed by the payment service.

Why we need Messaging Queues

- Tight Coupling: Traditional systems often exhibit tight coupling, making it hard to adapt or scale.

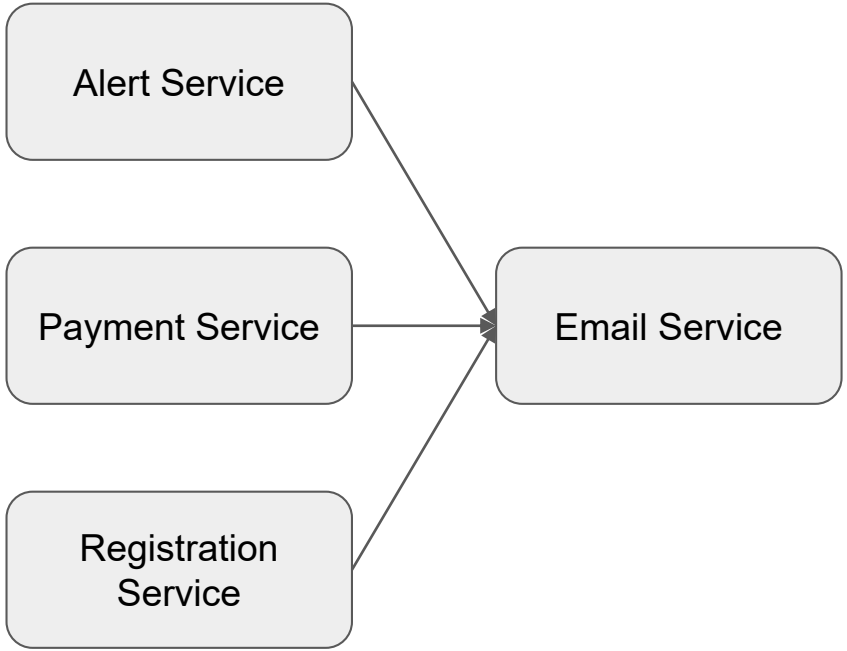
- Point-to-Point: Direct communication between components may lead to dependencies and bottlenecks.



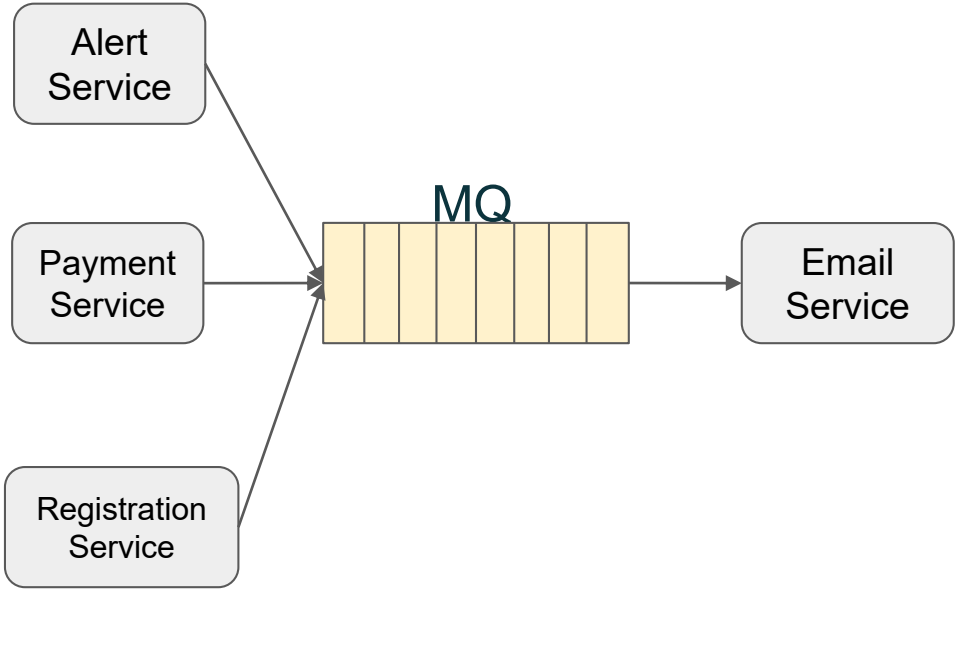
Tight Coupling

More Interdependency

Traditional Approach



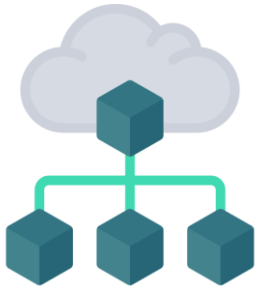
With MQ



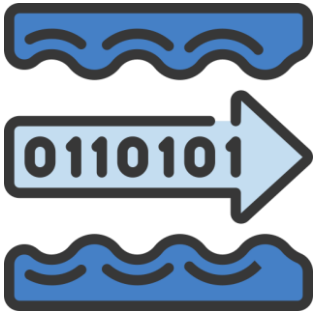
Advantages of Messaging Queues

- Decoupling: Producer and consumer operate independently, improving scalability and resilience.
- Asynchronous communication: Enables parallel processing, optimizing resource utilization.
- Message persistence: Ensures reliable delivery even if components fail or restart.
- Load balancing: Distributes messages across multiple consumers for better handling of peak loads.

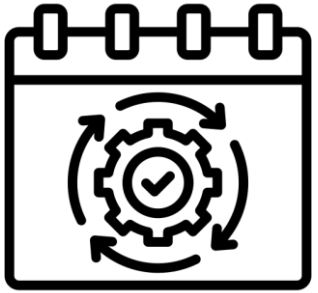
Use Cases for Messaging Queues



Microservices



Data Streaming

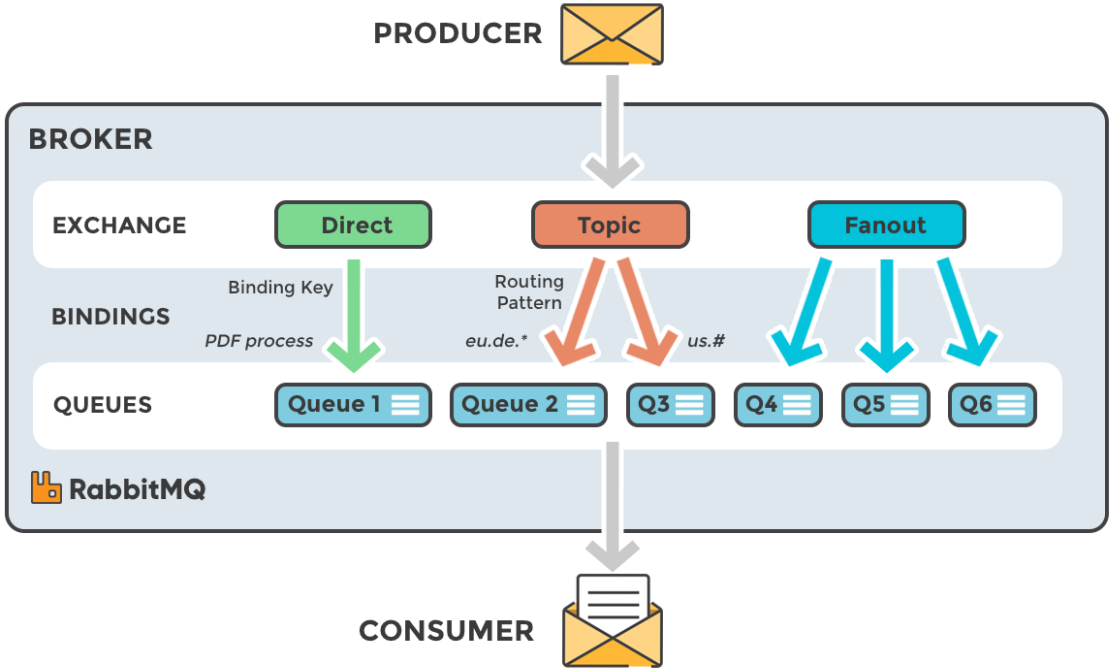


Event Driven

Types of Message Brokers

- Open-source message broker
- Written in Erlang
- Multiprotocol support (AMQP, STOMP, MQTT)
- Provides client libraries for various programming languages, including Java, Python, .NET, Ruby, and more.

RabbitMQ Architecture



Features

1. Cross Language
2. Security
3. Message Ack
4. Management
5. Cloud Friendly
6. Plug-In

Code Walkthrough

Producer

producer.py > ...

```python
# Import the pika library for connecting to RabbitMQ
import pika

# Establish a connection to the RabbitMQ server running on localhost
connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))

# Create a channel within the connection
channel = connection.channel()

# Declare a queue named "welcome" where messages will be sent
channel.queue_declare(queue='welcome')

# Publish a message to the "welcome" queue with the routing key "welcome"
# The routing key is used to direct messages to specific consumers if exchanges are involved
# In this case, we're leaving the exchange empty, meaning the message will be delivered to any consumer listening on the "welcome" queue
channel.basic_publish(exchange='',
                      routing_key='welcome',
                      body='Welcome to RabbitMQ!')

# Print a confirmation message
print("Sent -> 'Welcome to RabbitMQ!'")

# Close the connection to the RabbitMQ server
connection.close()
```

Consumer

consumer.py > ...

```python
# Import the pika library for connecting to RabbitMQ
import pika

# Establish a connection to the RabbitMQ server running on localhost
connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))

# Create a channel within the connection
channel = connection.channel()

# Declare a queue named "welcome" where messages will be received
channel.queue_declare(queue='welcome')

# Define a callback function to process received messages
def callback(ch, method, properties, body):
    """
    This function is called whenever a message is received on the "welcome" queue.
    It simply prints the message body.
    :param body: The message body (content).
    """
    print(f" Received {body}")

# Start consuming messages from the "welcome" queue
channel.basic_consume(queue='welcome',
                      auto_ack=True,  # Automatically acknowledge messages after processing
                      on_message_callback=callback)

# Start the consumption process
channel.start_consuming()
```

Code Gist

https://bit.ly/3I88RLJ

Thank You

You ‘ve been an amazing audience !!

Ahtesham Zaidi
Software Engineer