

Develop Message Queue Driver and Verify

A message queue is a **component of messaging middleware solutions** that enables independent applications and services to exchange information. Message queues store “messages”—packets of data that applications create for other applications to consume—in the order they are transmitted until the consuming application can process them. This enables messages to wait safely until the receiving application is ready, so if there is a problem with the network or receiving application, the messages in the message queue are not lost.

This model, known as **asynchronous messaging**, prevents data loss and enables **systems to continue to function if processes or connections fail**. This allows **developers** to keep processes and applications separate, keeping their communications self-contained and event-driven to make the architecture more reliable.

Please install RabbitMQ and run the Producer and Consumer applications 10000. And capture Message counts to make sure no message is lost.

Steps -

1> Install erlang

2> Install rabbitMQ

3> Enable the rabbit MQ management plugin, and access it on the localhost

RabbitMQ Management

127.0.0.1:15672/#/queues

RabbitMQ 3.12.6 Erlang 26.1.1

Refreshed 2023-10-08 17:14:25 Refresh every 5 seconds

Virtual host All

Cluster rabbit@USCS-704

User guest Log out

Overview Connections Channels Exchanges **Queues and Streams** Admin

Queues

All queues (1)

Pagination

Page 1 of 1 Filter: ☐ Regex

Displaying 1 item, page size up to: 100

Overview				Messages			Message rates				
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	Incoming	deliver	get	ack
/	hello	classic		running	22,133	0	22,133	937/s	2,702/s	0.00/s	

Add a new queue

HTTP API Documentation Tutorials New releases Commercial edition Commercial support Google Group Discord Slack Plugins GitHub

4> Put one message on the queue and review status

The screenshot shows the RabbitMQ Management UI Overview page. The browser address bar displays `127.0.0.1:15672/#/`. The page header includes the RabbitMQ logo, version 3.12.6, and Erlang 26.1.1. The navigation bar has tabs for Overview, Connections, Channels, Exchanges, Queues and Streams, and Admin. The Overview section shows a 'Totals' summary with a 'Queued messages' graph (last minute) and a 'Message rates' graph (last minute). The 'Global counts' section shows 0 connections, 0 channels, 7 exchanges, 1 queue, and 0 consumers. The 'Nodes' section shows a table with columns for Name, File descriptors, Socket descriptors, Erlang processes, Memory, Disk space, Uptime, Info, and Reset stats. The 'Churn statistics' section shows a table with columns for Name, File descriptors, Socket descriptors, Erlang processes, Memory, Disk space, Uptime, Info, and Reset stats.

Overview

Totals

Queued messages [last minute](#)

Message rates [last minute](#)

Global counts

Connections: 0 Channels: 0 Exchanges: 7 Queues: 1 Consumers: 0

Nodes

Name	File descriptors	Socket descriptors	Erlang processes	Memory	Disk space	Uptime	Info	Reset stats
rabbit@USCS-704	0	0	401	73 MiB	52 GiB	18m 35s	basic disc 1 198	This node All nodes

Churn statistics

Ports and contexts

Export definitions

Import definitions

[HTTP API](#) [Documentation](#) [Tutorials](#) [New releases](#) [Commercial edition](#) [Commercial support](#) [Google Group](#) [Discord](#) [Slack](#) [Plugins](#) [GitHub](#)

The screenshot shows the RabbitMQ Management UI Queues page. The browser address bar displays `127.0.0.1:15672/#/queues`. The page header includes the RabbitMQ logo, version 3.12.6, and Erlang 26.1.1. The navigation bar has tabs for Overview, Connections, Channels, Exchanges, Queues and Streams, and Admin. The Queues section shows a 'All queues (1)' summary with a pagination bar. The 'Overview' section shows a table with columns for Virtual host, Name, Type, Features, State, Ready, Unacked, Total, and Message rates. The 'Add a new queue' button is visible.

Queues

All queues (1)

Pagination

Page 1 of 1 - Filter: ☐ Regex

Displaying 1 item, page size up to: 100

Overview

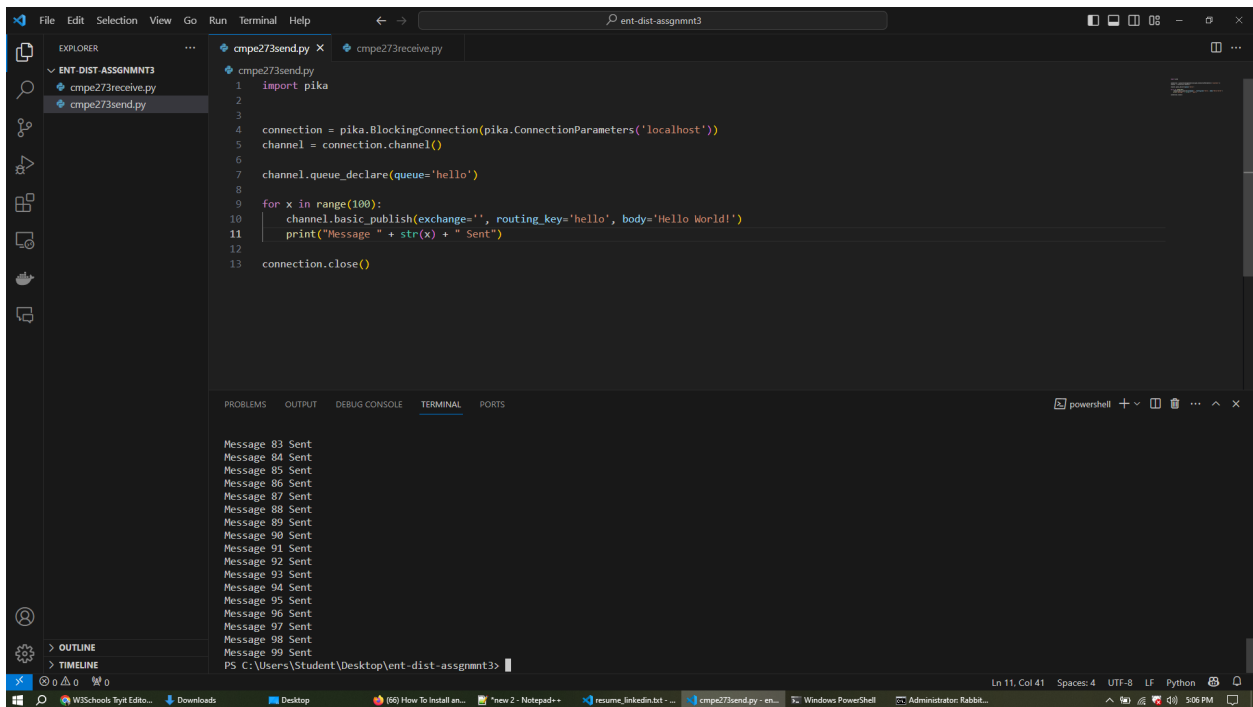
Virtual host	Name	Type	Features	State	Ready	Unacked	Total	Message rates
/	hello	classic		running	22,133	0	22,133	957/s 2,702/s 0.00/s

[Add a new queue](#)

[HTTP API](#) [Documentation](#) [Tutorials](#) [New releases](#) [Commercial edition](#) [Commercial support](#) [Google Group](#) [Discord](#) [Slack](#) [Plugins](#) [GitHub](#)

The screenshot shows the Windows taskbar with various open applications including Python For Loops, RabbitMQ Management, Downloads, Desktop, Screenshots, How To Install, new 2 - Notepad++, resume_linkedin, cmpc27send.py, Windows PowerShell, and Administrator: RabbitMQ Management. The system clock shows 5:14 PM.

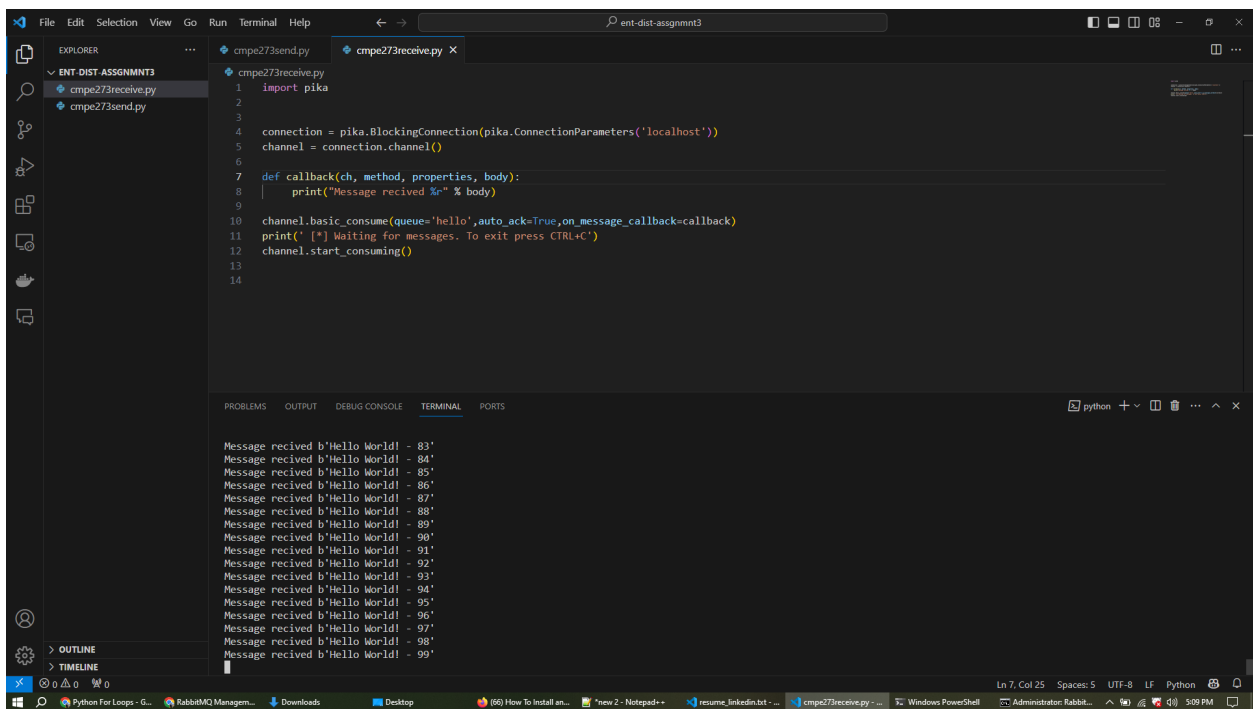
5> Put 100 messages on the queue and review the status



The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the project structure: `ENT-DIST-ASSGNMNT3` containing `cmpe273receive.py` and `cmpe273send.py`. The `cmpe273send.py` file is open in the editor, showing a Python script that uses the `pika` library to connect to a RabbitMQ instance on localhost, declare a queue named 'hello', and publish 100 messages with the body 'Hello World!'. The terminal at the bottom shows the output of the script, displaying 'Message 83 Sent' through 'Message 99 Sent'.

```
1 import pika
2
3
4 connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
5 channel = connection.channel()
6
7 channel.queue_declare(queue='hello')
8
9 for x in range(100):
10     channel.basic_publish(exchange='', routing_key='hello', body='Hello World!')
11     print("Message " + str(x) + " Sent")
12
13 connection.close()
```

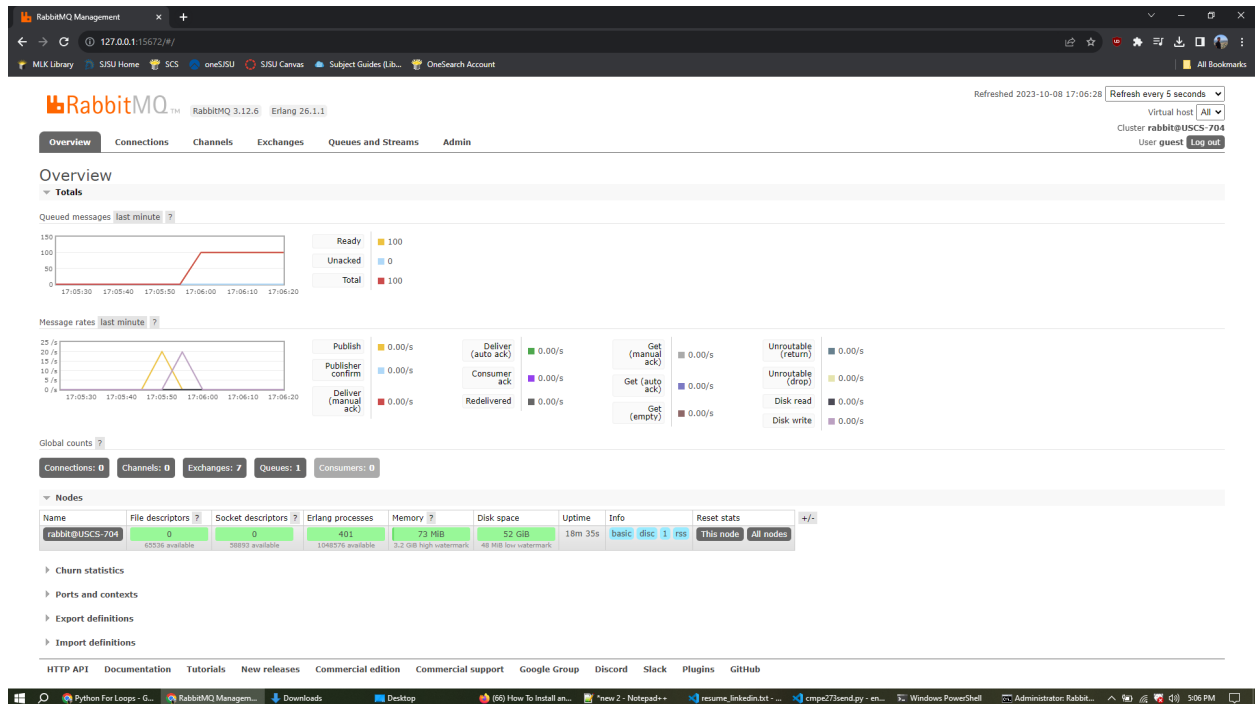
Message 83 Sent
Message 84 Sent
Message 85 Sent
Message 86 Sent
Message 87 Sent
Message 88 Sent
Message 89 Sent
Message 90 Sent
Message 91 Sent
Message 92 Sent
Message 93 Sent
Message 94 Sent
Message 95 Sent
Message 96 Sent
Message 97 Sent
Message 98 Sent
Message 99 Sent
PS C:\Users\Student\Desktop\ent-dist-assgnmnt3>



The screenshot shows the Visual Studio Code editor with the file explorer on the left displaying the project structure: `ENT-DIST-ASSGNMNT3` containing `cmpe273receive.py` and `cmpe273send.py`. The `cmpe273receive.py` file is open in the editor, showing a Python script that uses the `pika` library to connect to a RabbitMQ instance on localhost, declare a queue named 'hello', and consume messages. The terminal at the bottom shows the output of the script, displaying 'Message received b'Hello World! - 83' through 'Message received b'Hello World! - 99'.

```
1 import pika
2
3
4 connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
5 channel = connection.channel()
6
7 def callback(ch, method, properties, body):
8     print("Message received %s" % body)
9
10 channel.basic_consume(queue='hello', auto_ack=True, on_message_callback=callback)
11 print(' [*] Waiting for messages. To exit press CTRL+C')
12 channel.start_consuming()
13
14
```

Message received b'Hello World! - 83'
Message received b'Hello World! - 84'
Message received b'Hello World! - 85'
Message received b'Hello World! - 86'
Message received b'Hello World! - 87'
Message received b'Hello World! - 88'
Message received b'Hello World! - 89'
Message received b'Hello World! - 90'
Message received b'Hello World! - 91'
Message received b'Hello World! - 92'
Message received b'Hello World! - 93'
Message received b'Hello World! - 94'
Message received b'Hello World! - 95'
Message received b'Hello World! - 96'
Message received b'Hello World! - 97'
Message received b'Hello World! - 98'
Message received b'Hello World! - 99'



6> Put 10000 messages on the queue and review

```
File Edit Selection View Go Run Terminal Help
ent-dist-assignmt3

EXPLORER
ENT-DIST-ASSIGNMT3
  cmpe273receive.py
  cmpe273send.py

cmpe273send.py
1 import pika
2
3
4 connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
5 channel = connection.channel()
6 channel.queue_declare(queue='hello')
7
8
9 for x in range(10000):
10     channel.basic_publish(exchange='', routing_key='hello', body='Hello World! - ' + str(x))
11     print("Message " + str(x) + " Sent")
12
13 connection.close()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Message 9983 Sent
Message 9984 Sent
Message 9985 Sent
Message 9986 Sent
Message 9987 Sent
Message 9988 Sent
Message 9989 Sent
Message 9990 Sent
Message 9991 Sent
Message 9992 Sent
Message 9993 Sent
Message 9994 Sent
Message 9995 Sent
Message 9996 Sent
Message 9997 Sent
Message 9998 Sent
Message 9999 Sent
PS C:\Users\Student\Desktop\ent-dist-assignmt3>
```

```
File Edit Selection View Go Run Terminal Help
ent-dist-assignmt3

EXPLORER
ENT-DIST-ASSIGNMT3
  cmpe273receive.py
  cmpe273send.py

cmpe273send.py
1 import pika
2
3
4 connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
5 channel = connection.channel()
6
7 channel.queue_declare(queue='hello')
8
9 for x in range(10000):
10     channel.basic_publish(exchange='', routing_key='hello', body='Hello World! - ' + str(x))
11     print("Message " + str(x) + " Sent")
12
13 connection.close()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
python

Message recived b'Hello World! - 9983'
Message recived b'Hello World! - 9984'
Message recived b'Hello World! - 9985'
Message recived b'Hello World! - 9986'
Message recived b'Hello World! - 9987'
Message recived b'Hello World! - 9988'
Message recived b'Hello World! - 9989'
Message recived b'Hello World! - 9990'
Message recived b'Hello World! - 9991'
Message recived b'Hello World! - 9992'
Message recived b'Hello World! - 9993'
Message recived b'Hello World! - 9994'
Message recived b'Hello World! - 9995'
Message recived b'Hello World! - 9996'
Message recived b'Hello World! - 9997'
Message recived b'Hello World! - 9998'
Message recived b'Hello World! - 9999'
```

RabbitMQ Management

127.0.0.1:15672/#/

Overview Connections Channels Exchanges Queues and Streams Admin

Overview

Totals

Queued messages **last minute**

Ready: 81,678
Unacked: 0
Total: 81,678

Message rates **last minute**

Global counts

Connections: 2 Channels: 2 Exchanges: 7 Queues: 1 Consumers: 1

Nodes

Name	File descriptors	Socket descriptors	Erlang processes	Memory	Disk space	Uptime	Info	Reset state
rabbit@USCS-704	0	2	424	97 MiB	52 GiB	25m 37s	basic disc 1 fs	This node All nodes

Churn statistics

Ports and contexts

Export definitions

Import definitions

HTTP API Documentation Tutorials New releases Commercial edition Commercial support Google Group Discord Slack Plugins GitHub