Assignment 4: Message Broker

Installing Rabbitmq:

At first we need to update the system by running:

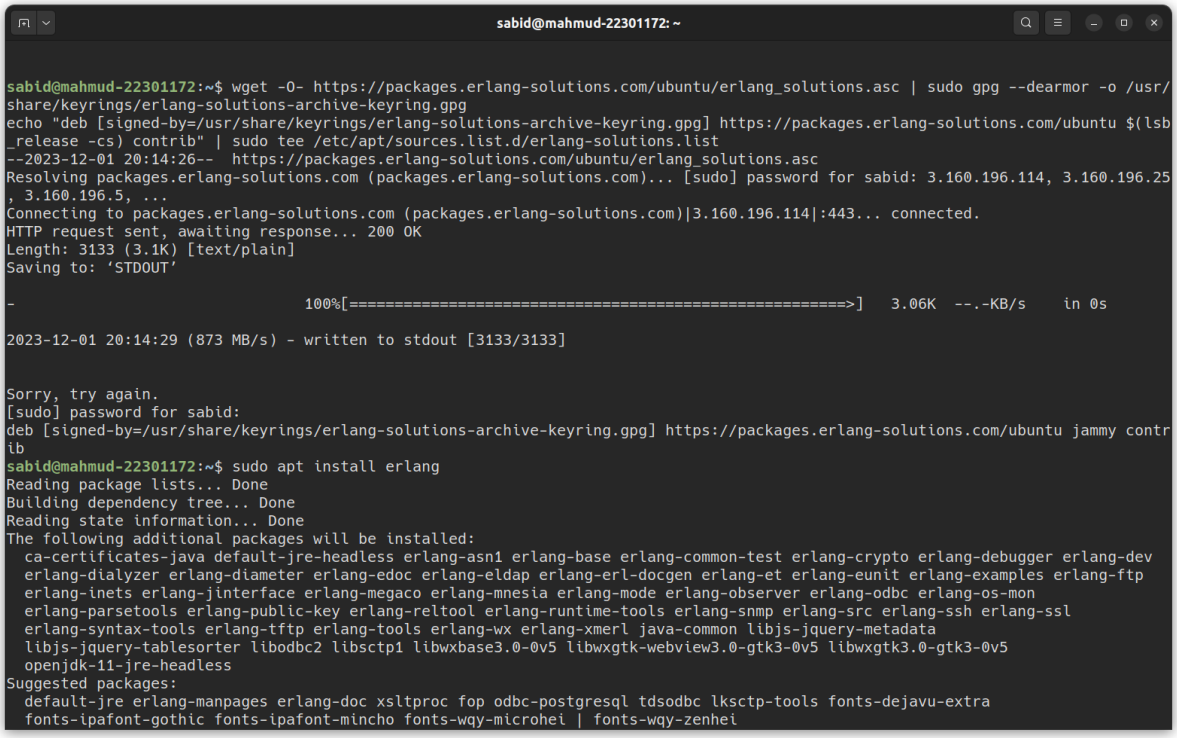
```
sudo apt update && upgrade
```

Installing Erlang:

RabbitMQ requires Erlang. So, we need to install the erlang first.

To do that, we can run the following commands:

```
wget -O- https://packages.erlang-solutions.com/ubuntu/erlang_solutions.asc | sudo gpg
--dearmor -o /usr/share/keyrings/erlang-solutions-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/erlang-solutions-archive-keyring.gpg] http
s://packages.erlang-solutions.com/ubuntu $(lsb_release -cs) contrib" | sudo tee /etc/a
pt/sources.list.d/erlang-solutions.list
sudo apt install erlang
```

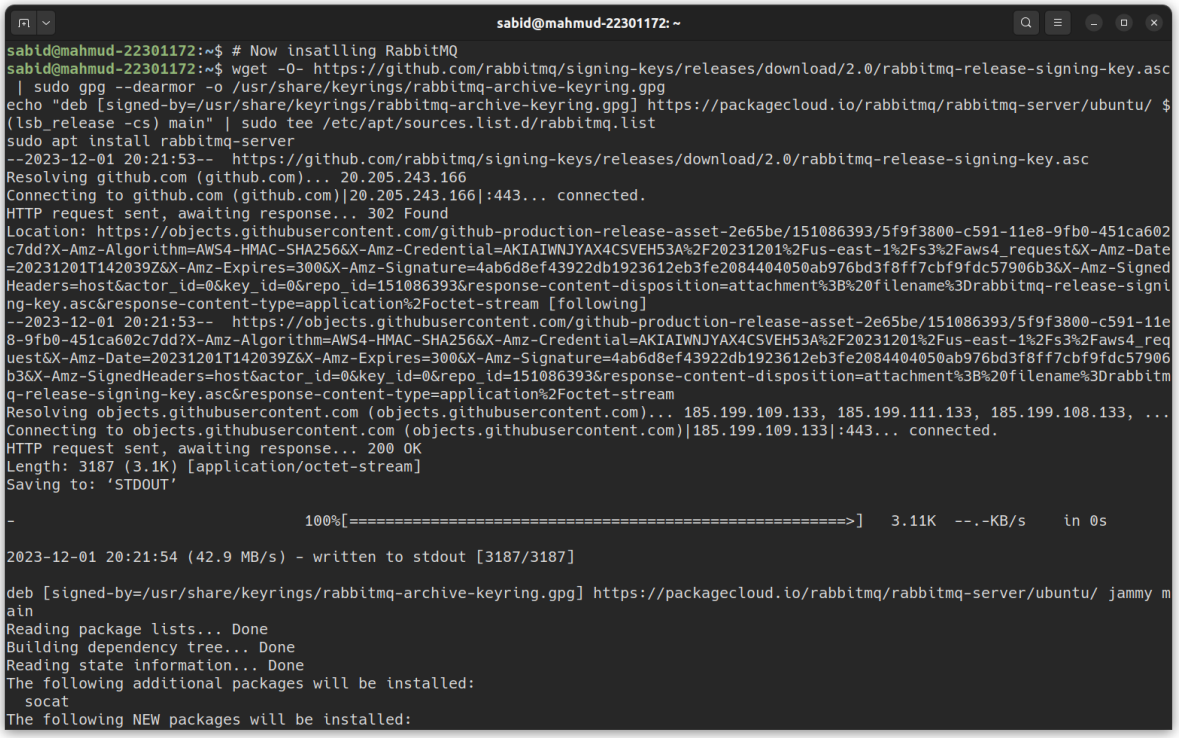


```
sabid@mahmud-22301172: ~

sabid@mahmud-22301172:~$ wget -O- https://packages.erlang-solutions.com/ubuntu/erlang_solutions.asc | sudo gpg --dearmor -o /usr/
share/keyrings/erlang-solutions-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/erlang-solutions-archive-keyring.gpg] https://packages.erlang-solutions.com/ubuntu $(lsb
_release -cs) contrib" | sudo tee /etc/apt/sources.list.d/erlang-solutions.list
--2023-12-01 20:14:26--  https://packages.erlang-solutions.com/ubuntu/erlang_solutions.asc
Resolving packages.erlang-solutions.com (packages.erlang-solutions.com)... [sudo] password for sabid: 3.160.196.114, 3.160.196.25
, 3.160.196.5, ...
Connecting to packages.erlang-solutions.com (packages.erlang-solutions.com)|3.160.196.114|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3133 (3.1K) [text/plain]
Saving to: 'STDOUT'

-                                   100%[===================================================>]   3.06K  --.-KB/s    in 0s

2023-12-01 20:14:29 (873 MB/s) - written to stdout [3133/3133]


Sorry, try again.
[sudo] password for sabid:
deb [signed-by=/usr/share/keyrings/erlang-solutions-archive-keyring.gpg] https://packages.erlang-solutions.com/ubuntu jammy contr
ib
sabid@mahmud-22301172:~$ sudo apt install erlang
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java default-jre-headless erlang-asn1 erlang-base erlang-common-test erlang-crypto erlang-debugger erlang-dev
  erlang-dialyzer erlang-diameter erlang-edoc erlang-eldap erlang-erl-docgen erlang-et erlang-eunit erlang-examples erlang-ftp
  erlang-inets erlang-jinterface erlang-megaco erlang-mnesia erlang-mode erlang-observer erlang-odbc erlang-os-mon
  erlang-parsetools erlang-public-key erlang-reltool erlang-runtime-tools erlang-snmp erlang-src erlang-ssh erlang-ssl
  erlang-syntax-tools erlang-tftp erlang-tools erlang-wx erlang-xmerl java-common libjs-jquery-metadata
  libjs-jquery-tablesorter libodbc2 libsctp1 libwxbase3.0-0v5 libwxgtk-webview3.0-gtk3-0v5 libwxgtk3.0-gtk3-0v5
  openjdk-11-jre-headless
Suggested packages:
  default-jre erlang-manpages erlang-doc xsltproc fop odbc-postgresql tdsodbc lksctp-tools fonts-dejavu-extra
  fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei | fonts-wqy-zenhei
```

Installing RabbitMQ:

In this step I will install rabbitmq-server in my machine. To do that, I have to run this command on terminal.

```
sudo apt update
wget -O- https://github.com/rabbitmq/signing-keys/releases/download/2.0/rabbitmq-relea
se-signing-key.asc | sudo gpg --dearmor -o /usr/share/keyrings/rabbitmq-archive-keyrin
g.gpg
echo "deb [signed-by=/usr/share/keyrings/rabbitmq-archive-keyring.gpg] https://package
cloud.io/rabbitmq/rabbitmq-server/ubuntu/ $(lsb_release -cs) main" | sudo tee /etc/ap
t/sources.list.d/rabbitmq.list
sudo apt install rabbitmq-server
```



```
sabid@mahmud-22301172: ~
sabid@mahmud-22301172:~$ # Now insatlling RabbitMQ
sabid@mahmud-22301172:~$ wget -O- https://github.com/rabbitmq/signing-keys/releases/download/2.0/rabbitmq-release-signing-key.asc
 | sudo gpg --dearmor -o /usr/share/keyrings/rabbitmq-archive-keyring.gpg
echo "deb [signed-by=/usr/share/keyrings/rabbitmq-archive-keyring.gpg] https://packagecloud.io/rabbitmq/rabbitmq-server/ubuntu/ $
(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/rabbitmq.list
sudo apt install rabbitmq-server
--2023-12-01 20:21:53--  https://github.com/rabbitmq/signing-keys/releases/download/2.0/rabbitmq-release-signing-key.asc
Resolving github.com (github.com)... 20.205.243.166
Connecting to github.com (github.com)|20.205.243.166|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/151086393/5f9f3800-c591-11e8-9fb0-451ca602
c7dd?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20231201%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date
=20231201T142039Z&X-Amz-Expires=300&X-Amz-Signature=4ab6d8ef43922db1923612eb3fe2084404050ab976bd3f8ff7cbf9fdc57906b3&X-Amz-Signed
Headers=host&actor_id=0&key_id=0&repo_id=151086393&response-content-disposition=attachment%3B%20filename%3Drabbitmq-release-signi
ng-key.asc&response-content-type=application%2Foctet-stream [following]
--2023-12-01 20:21:53--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/151086393/5f9f3800-c591-11e
8-9fb0-451ca602c7dd?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20231201%2Fus-east-1%2Fs3%2Faws4_req
uest&X-Amz-Date=20231201T142039Z&X-Amz-Expires=300&X-Amz-Signature=4ab6d8ef43922db1923612eb3fe2084404050ab976bd3f8ff7cbf9fdc57906
b3&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=151086393&response-content-disposition=attachment%3B%20filename%3Drabbitm
q-release-signing-key.asc&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3187 (3.1K) [application/octet-stream]
Saving to: 'STDOUT'

-                                   100%[=====================================================>]   3.11K  --.-KB/s    in 0s

2023-12-01 20:21:54 (42.9 MB/s) - written to stdout [3187/3187]

deb [signed-by=/usr/share/keyrings/rabbitmq-archive-keyring.gpg] https://packagecloud.io/rabbitmq/rabbitmq-server/ubuntu/ jammy m
ain
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  socat
The following NEW packages will be installed:
```

Enabling rabbitmq:

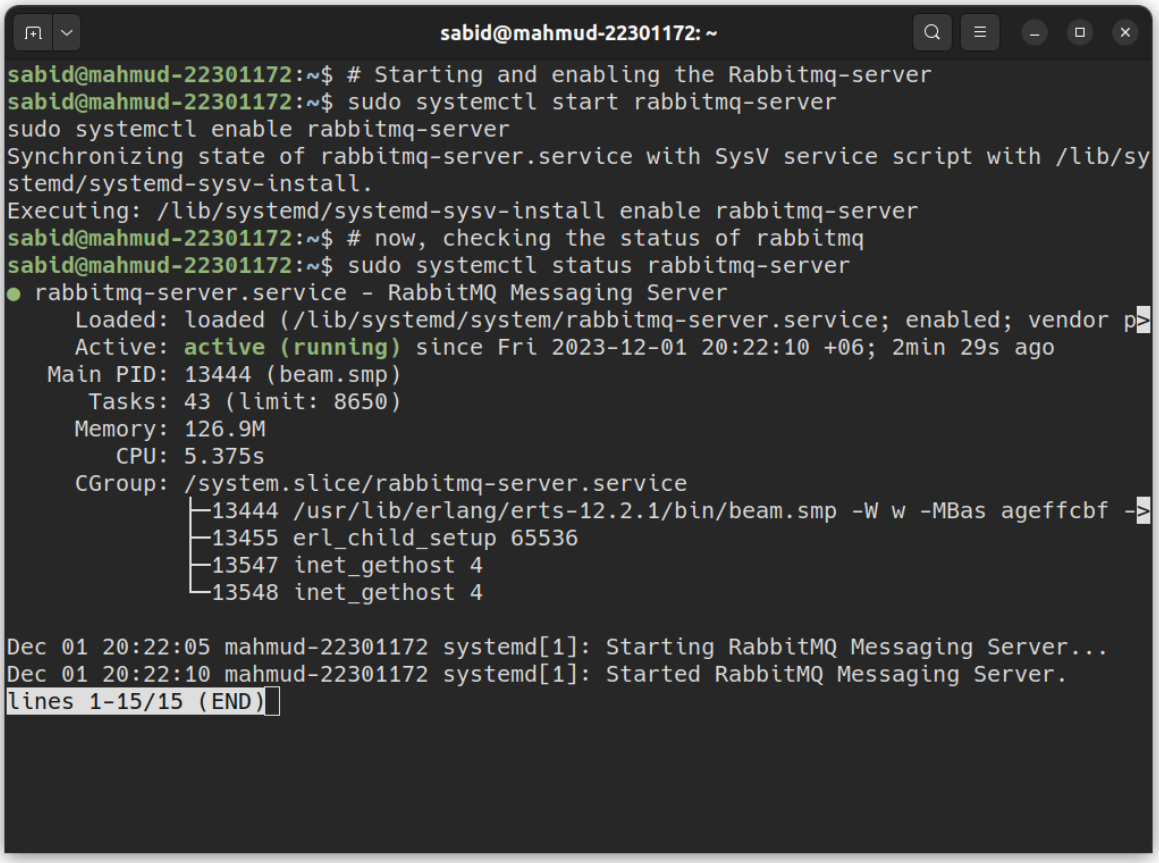
once installed, by running this command, I can start and enable rabbitmq-server in my machine.

```
sudo systemctl start rabbitmq-server
sudo systemctl enable rabbitmq-server
```

Checking the status:

We can check the status of the rabbitmq-server by running this command.

```
sudo systemctl status rabbitmq-server
```



Install Pika

Pika requires python and pip. So, If we do not have python and pip installed, we need to install it first. In this case, I am using python3.

So, let's install python3 and pip3.

```
sudo apt install python3 python3-pip
```

Then, we can use the `pip` to install pika.

```
pip3 install pika
```

```
sabid@mahmud-22301172: ~
sabid@mahmud-22301172:~$ sudo apt-get install python3-dev python3-pip
[sudo] password for sabid:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3-dev is already the newest version (3.10.6-1~22.04).
python3-dev set to manually installed.
python3-pip is already the newest version (22.0.2+dfsg-1ubuntu0.4).
0 upgraded, 0 newly installed, 0 to remove and 8 not upgraded.
sabid@mahmud-22301172:~$ pip install pika
Defaulting to user installation because normal site-packages is not writeable
Collecting pika
  Downloading pika-1.3.2-py3-none-any.whl (155 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 155.4/155.4 KB 1.2 MB/s eta 0:00:00
Installing collected packages: pika
Successfully installed pika-1.3.2
sabid@mahmud-22301172:~$
```

Let’s verify that pika is installed

```
pip3 show pika
```

```
sabid@mahmud-22301172: ~
sabid@mahmud-22301172:~$ pip3 show pika
Name: pika
Version: 1.3.2
Summary: Pika Python AMQP Client Library
Home-page:
Author:
Author-email:
License: BSD-3-Clause
Location: /home/sabid/.local/lib/python3.10/site-packages
Requires:
Required-by:
sabid@mahmud-22301172:~$
```

TASK1: Hello World

Creating a producer and a consumer script using pika:

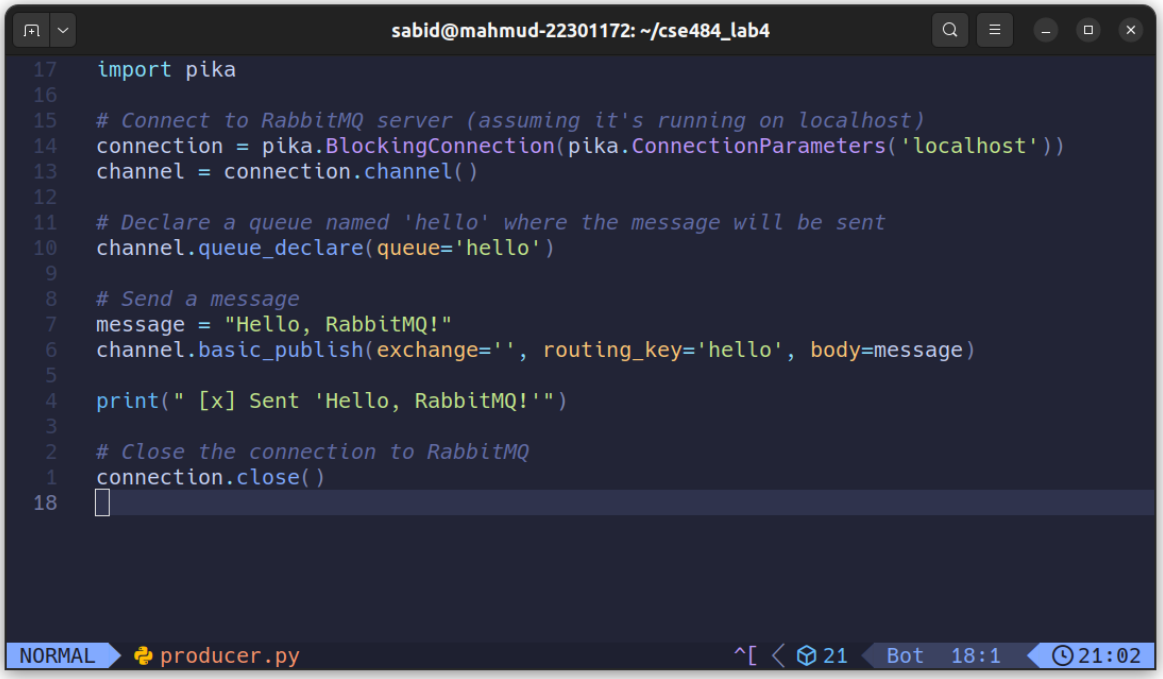
Create a file `producer.py` using a text editor. In this case I am using my default text editor neovim.

```
nvim producer.py
```

Then write this code in the file:

```
import pika

# Connect to RabbitMQ server (assuming it's running on localhost)
connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
channel = connection.channel()

# Declare a queue named 'hello' where the message will be sent
channel.queue_declare(queue='hello')

# Send a message
message = "Hello, RabbitMQ!"
channel.basic_publish(exchange='', routing_key='hello', body=message)

print(" [x] Sent 'Hello, RabbitMQ!'")

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

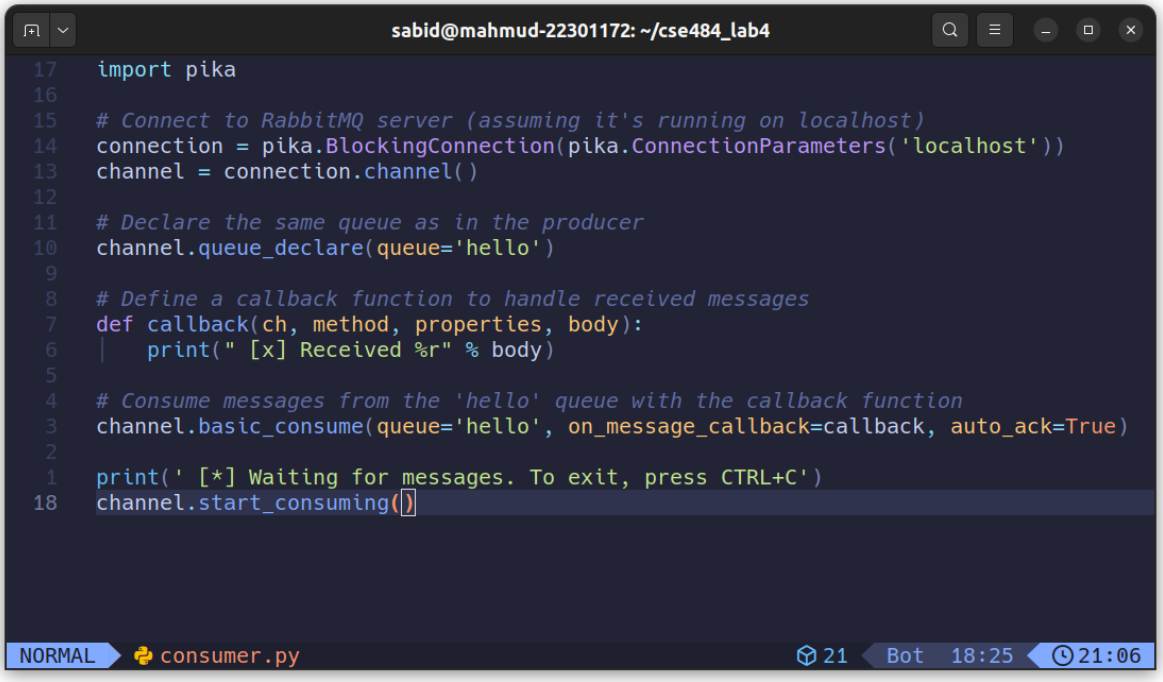
save the the file and exit.



Now, let us create another file `consumer.py` using the same method:

```
nvim consumer.py
```

```python
import pika

# Connect to RabbitMQ server (assuming it's running on localhost)
connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
channel = connection.channel()

# Declare the same queue as in the producer
channel.queue_declare(queue='hello')

# Define a callback function to handle received messages
def callback(ch, method, properties, body):
    print(" [x] Received %r" % body)

# Consume messages from the 'hello' queue with the callback function
channel.basic_consume(queue='hello', on_message_callback=callback, auto_ack=True)

print(' [*] Waiting for messages. To exit, press CTRL+C')
channel.start_consuming()
```



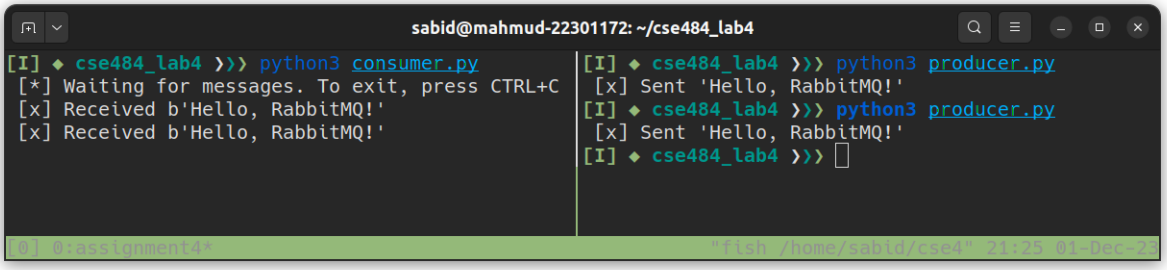
Now let us run the `consumer.py` file

```
python3 consumer.py
```

Now, Run the `producer.py` file

```
python3 producer.py
```

If everything runs perfectly, the output should seems like this:



TAKS2: Work queues

To complete Task 2, "Work Queues," using RabbitMQ, we will create a task sender script and a worker script that processes tasks with varying complexities.

Create the Task Sender Script:

Lets write a python script `new_task.py` that will send messages simulating tasks with various processing time.

```
nvim new_task.py
```

Now write this code in the file.

```
import pika
import sys

connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
channel = connection.channel()

channel.queue_declare(queue='task_queue', durable=True)

message = ' '.join(sys.argv[1:]) or "Hello World!"
channel.basic_publish(
    exchange='',
    routing_key='task_queue',
    body=message,
    properties=pika.BasicProperties(
        delivery_mode=2,  # make message persistent
    ))
```

```
print(f" [x] Sent {message}")
connection.close()
```

Save and exit.

Create the Worker Script

Let us write another python script `worker.py` which will pick up tasks from the queue and simulate the processing of the tasks.

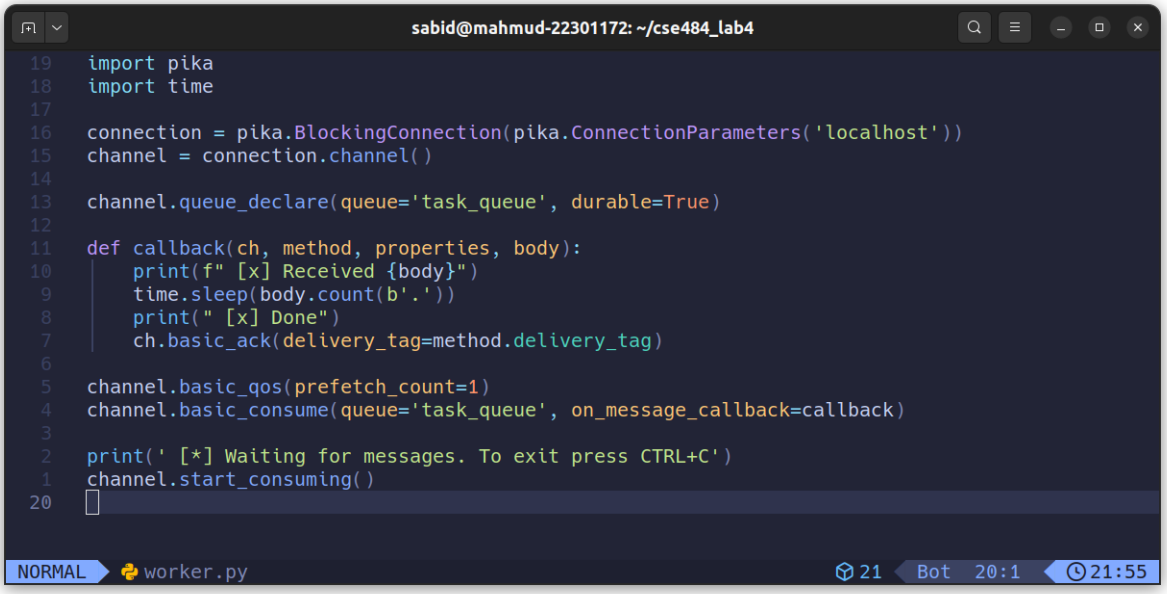
```
nvim worker.py
```

Now let us write this scrip in the file.

```
import pika
import time

connection = pika.BlockingConnection(pika.ConnectionParameters('localhost'))
channel = connection.channel()

channel.queue_declare(queue='task_queue', durable=True)

def callback(ch, method, properties, body):
    print(f" [x] Received {body}")
    time.sleep(body.count(b'.'))
    print(" [x] Done")
    ch.basic_ack(delivery_tag=method.delivery_tag)

channel.basic_qos(prefetch_count=1)
```

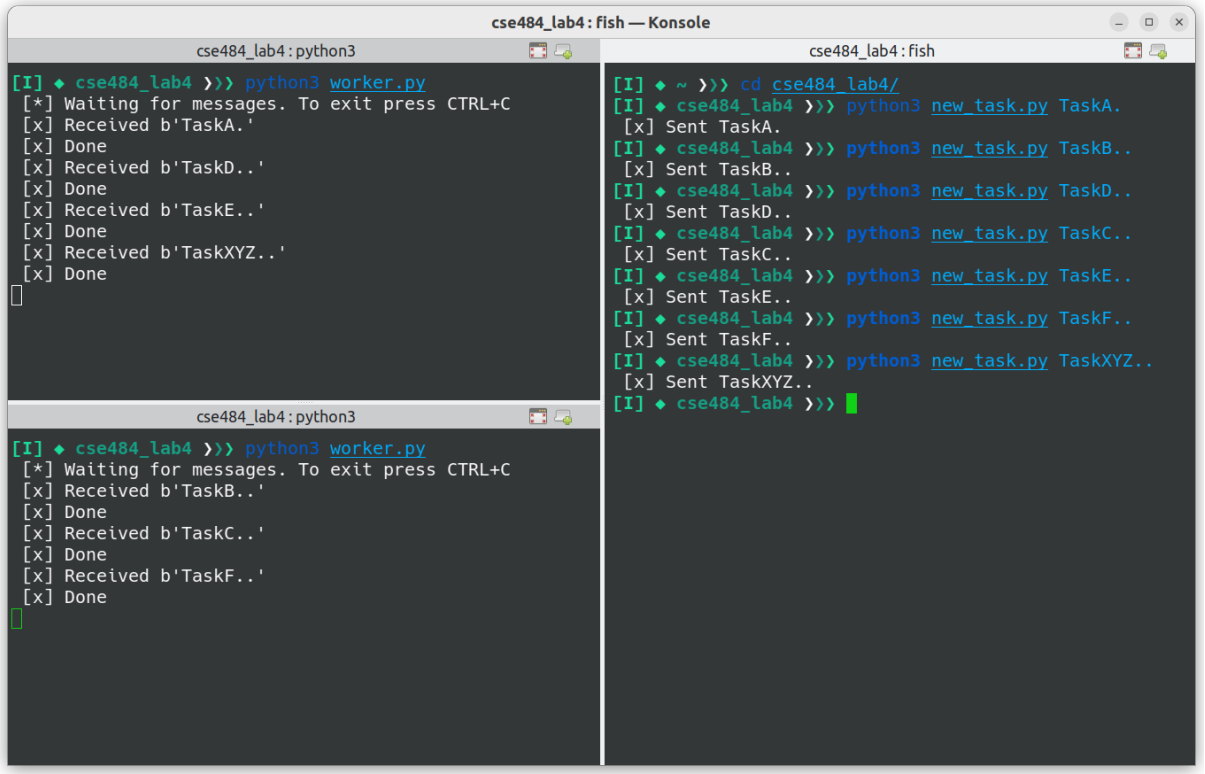
```
channel.basic_consume(queue='task_queue', on_message_callback=callback)

print(' [*] Waiting for messages. To exit press CTRL+C')
channel.start_consuming()
```

Let us save and exit.



Run the `worker.py` and Observe:

Now, let us run the `worker.py` and the `new_task.py` in a separate terminal.



My observation:

To illustrate RabbitMQ's work queue distribution, I set up three instances of `worker.py` in separate terminal windows, ordered from top to bottom based on their launch sequence. This arrangement aimed to showcase how RabbitMQ allocates tasks across different workers.

In a separate terminal acting as the task sender, I ran the `new_task.py` script using various commands to simulate tasks of differing lengths. For instance, commands like python3 `new_task.py` TaskA. and python3 `new_task.py` TaskB.. created tasks labeled from TaskA to TaskF, with increasing numbers of dots indicating longer processing times.

By monitoring the worker terminals, I observed the task distribution. The first worker (top terminal) handled '`TaskA.`' and '`TaskD..`' and "`TaskE..`", the second worker (bottom terminal) processed '`TaskB..`' and '`TaskC..`' and `TaskF..`. This demonstrated RabbitMQ's ability to evenly distribute tasks among available workers, showcasing its load-balancing capabilities.