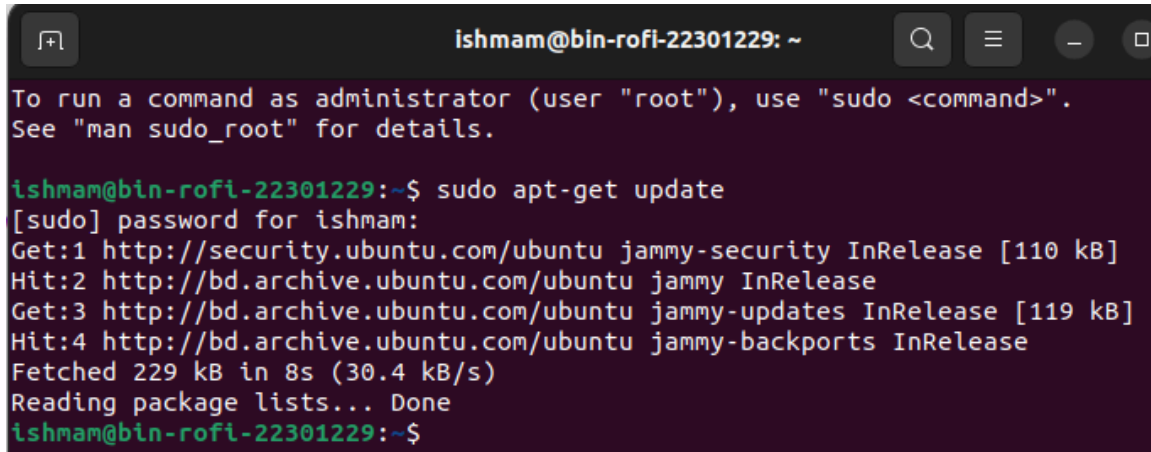


Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

`sudo apt-get update`

First in this step we are going to update our system by running this command on our terminal.

A terminal window with a dark background and light-colored text. The window title is 'ishmam@bin-rofi-22301229: ~'. It contains a message about running commands as administrator, followed by the execution of 'sudo apt-get update'. The output shows the system fetching updates from various sources, including security updates and backports, and reading package lists.

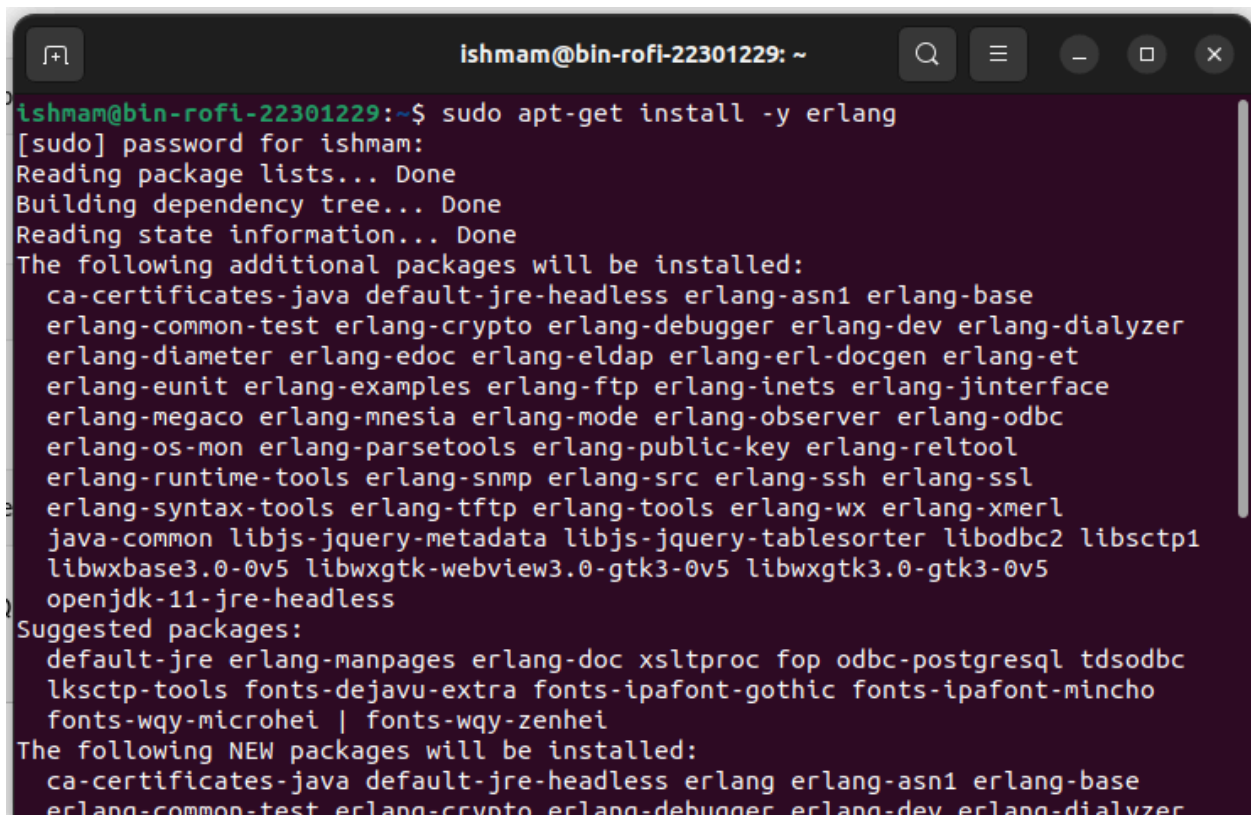
```
ishmam@bin-rofi-22301229: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ishmam@bin-rofi-22301229:~$ sudo apt-get update  
[sudo] password for ishmam:  
Get:1 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]  
Hit:2 http://bd.archive.ubuntu.com/ubuntu jammy InRelease  
Get:3 http://bd.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]  
Hit:4 http://bd.archive.ubuntu.com/ubuntu jammy-backports InRelease  
Fetched 229 kB in 8s (30.4 kB/s)  
Reading package lists... Done  
ishmam@bin-rofi-22301229:~$
```

Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

Sudo apt-get install -y erlang

The command `sudo apt-get install -y erlang` is used here to install the Erlang programming language on a Linux system.

A terminal window with a dark background and light-colored text. The window title is 'ishmam@bin-rofi-22301229: ~'. The command 'ishmam@bin-rofi-22301229:~\$ sudo apt-get install -y erlang' has been entered. The output shows the password prompt, progress bars for reading package lists, building the dependency tree, and reading state information. It then lists additional packages to be installed, including various Erlang modules and Java-related packages. Suggested packages are also listed, and finally, the NEW packages to be installed are listed.

```
ishmam@bin-rofi-22301229:~$ sudo apt-get install -y erlang
[sudo] password for ishmam:
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
The following NEW packages will be installed:
  ca-certificates-java default-jre-headless erlang erlang-asn1 erlang-base
  erlang-common-test erlang-crypto erlang-debugger erlang-dev erlang-dialyzer
```

Add the RabbitMQ repository to your system:

```
echo 'deb http://www.rabbitmq.com/debian/ testing main' | sudo tee /etc/apt/sources.list.d/rabbitmq.list
```

The command `echo 'deb http://www.rabbitmq.com/debian/ testing main' | sudo tee /etc/apt/sources.list.d/rabbitmq.list` is used to add the RabbitMQ repository to my system's software sources. It does this by echoing a string that represents the repository information, then using the tee command to write this string to the file `/etc/apt/sources.list.d/rabbitmq.list`.

Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

```
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
done.
ishmam@bin-rofi-22301229:~$ echo 'deb http://www.rabbitmq.com/debian/ testing main' | sudo tee /etc/apt/sources.list.d/rabbitmq.list
deb http://www.rabbitmq.com/debian/ testing main
ishmam@bin-rofi-22301229:~$
```

Install Rabbit-mq


sudo apt-get install -y rabbitmq-server

Here we will be installing the RabbitMQ server on our device.

Using the command sudo apt-get install -y rabbitmq-server.

```
ishmam@bin-rofi-22301229: ~
N: See apt-secure(8) manpage for repository creation and user configuration details.
ishmam@bin-rofi-22301229:~$ sudo apt-get install -y rabbitmq-server
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:
  rabbitmq-server socat
0 upgraded, 2 newly installed, 0 to remove and 96 not upgraded.
Need to get 15.5 MB of archives.
After this operation, 24.3 MB of additional disk space will be used.
Get:1 http://bd.archive.ubuntu.com/ubuntu jammy/main amd64 socat amd64 1.7.4.1-3ubuntu4 [349 kB]
Get:2 http://bd.archive.ubuntu.com/ubuntu jammy-updates/main amd64 rabbitmq-server all 3.9.13-1ubuntu0.22.04.2 [15.2 MB]
Fetched 15.5 MB in 4s (3,453 kB/s)
Selecting previously unselected package socat.
(Reading database ... 202101 files and directories currently installed.)
Preparing to unpack .../socat_1.7.4.1-3ubuntu4_amd64.deb ...
Unpacking socat (1.7.4.1-3ubuntu4) ...
Selecting previously unselected package rabbitmq-server.
Preparing to unpack .../rabbitmq-server_3.9.13-1ubuntu0.22.04.2_all.deb ...
```

Ishmam Bin Rofi - 22301229
CSE484 - assignment 4

Then we will check that Rabbitmq is running. The green dot  beside rabbitmq indicates that the server is running and been online.

```
ishmam@bin-rofi-22301229: ~  
Not creating home directory '/var/lib/rabbitmq'.  
Created symlink /etc/systemd/system/multi-user.target.wants/rabbitmq-server.service → /lib/systemd/system/rabbitmq-server.service.  
Processing triggers for man-db (2.10.2-1) ...  
ishmam@bin-rofi-22301229:~$ sudo systemctl status rabbitmq-server  
Unit rabbitmq-server.service could not be found.  
ishmam@bin-rofi-22301229:~$ sudo systemctl status rabbitmq-server  
● rabbitmq-server.service - RabbitMQ Messaging Server  
   Loaded: loaded (/lib/systemd/system/rabbitmq-server.service; enabled; vendor preset: enabled)  
   Active: active (running) since Sun 2024-03-31 18:29:11 +06; 1min 51s ago  
     Main PID: 7591 (beam.smp)  
       Tasks: 43 (limit: 18804)  
      Memory: 105.0M  
         CPU: 2.709s  
    CGroup: /system.slice/rabbitmq-server.service  
            └─7591 /usr/lib/erlang/erts-12.2.1/bin/beam.smp -W w -MBas ageffcb  
              └─7602 erl_child_setup 65536  
                └─7693 inet_gethost 4  
                  └─7694 inet_gethost 4  
  
মার্চ 31 18:29:06 bin-rofi-22301229 systemd[1]: Starting RabbitMQ Messaging Server:.  
মার্চ 31 18:29:11 bin-rofi-22301229 systemd[1]: Started RabbitMQ Messaging Server:.  
ishmam@bin-rofi-22301229:~$
```

Task1: Hello World

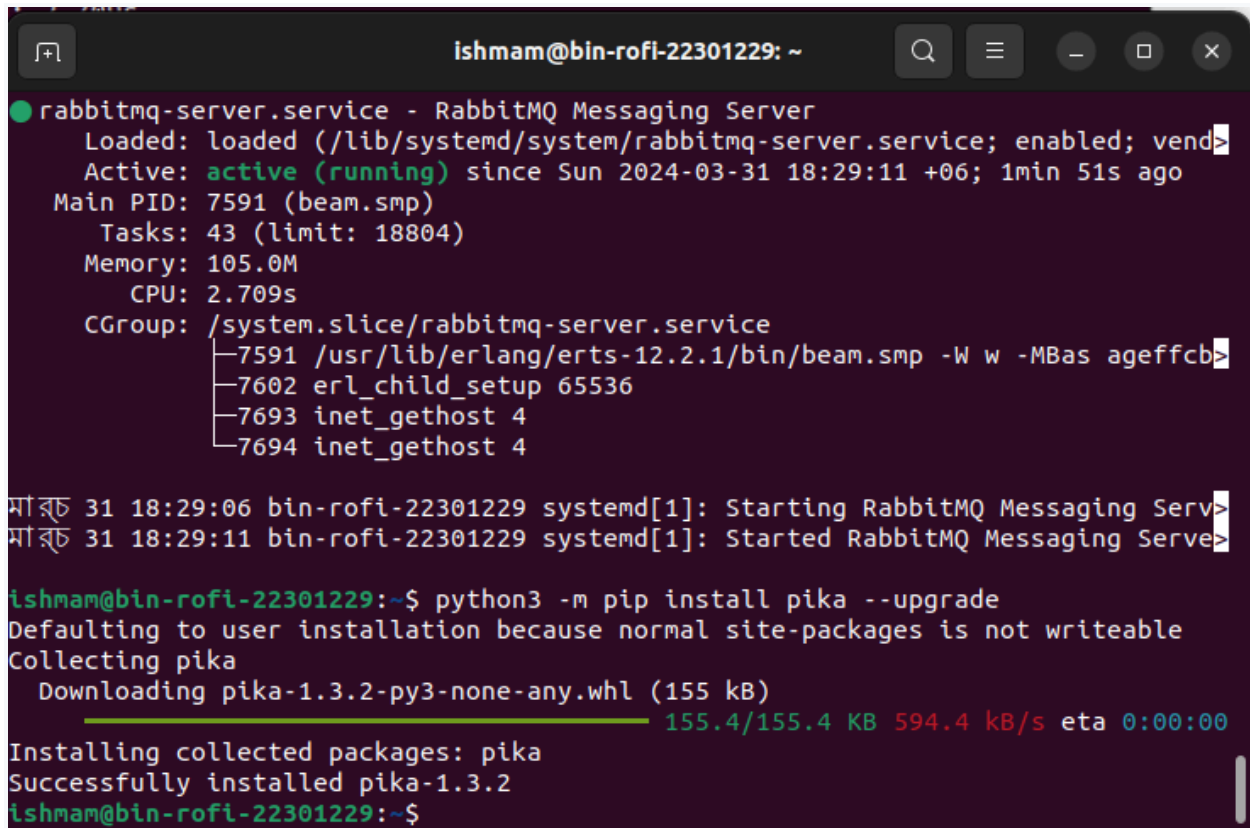
Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

First install pika

```
python3 -m pip install pika --upgrade
```

Here we are installing a python 3.3 library called Pika on our system's python server.

A terminal window titled 'ishmam@bin-rofi-22301229: ~' with standard window controls. It displays the status of the 'rabbitmq-server.service' as 'active (running)' with various system metrics. Below this, it shows the execution of 'python3 -m pip install pika --upgrade', which successfully installs 'pika-1.3.2' from a wheel file. The terminal text is as follows:

```
● rabbitmq-server.service - RabbitMQ Messaging Server
   Loaded: loaded (/lib/systemd/system/rabbitmq-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-03-31 18:29:11 +06; 1min 51s ago
     Main PID: 7591 (beam.smp)
        Tasks: 43 (limit: 18804)
      Memory: 105.0M
         CPU: 2.709s
       CGroup: /system.slice/rabbitmq-server.service
              └─7591 /usr/lib/erlang/erts-12.2.1/bin/beam.smp -W w -MBas ageffcb>
                └─7602 erl_child_setup 65536
                  └─7693 inet_gethost 4
                    └─7694 inet_gethost 4

শনি ৩১ ১৮:২৯:০৬ bin-rofi-22301229 systemd[1]: Starting RabbitMQ Messaging Serv>
শনি ৩১ ১৮:২৯:১১ bin-rofi-22301229 systemd[1]: Started RabbitMQ Messaging Serve>

ishmam@bin-rofi-22301229:~$ python3 -m pip install pika --upgrade
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 594.4 kB/s eta 0:00:00
Installing collected packages: pika
Successfully installed pika-1.3.2
ishmam@bin-rofi-22301229:~$
```

1. Writing the producer.py file

Now we are going to open a python file called producer.py on our system using gedit and copy paste this code from the RabbitMQ website here. Note that this code uses the channel connection to connect with pika.

Here is the python code that is for producer.py

```
'import pika

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

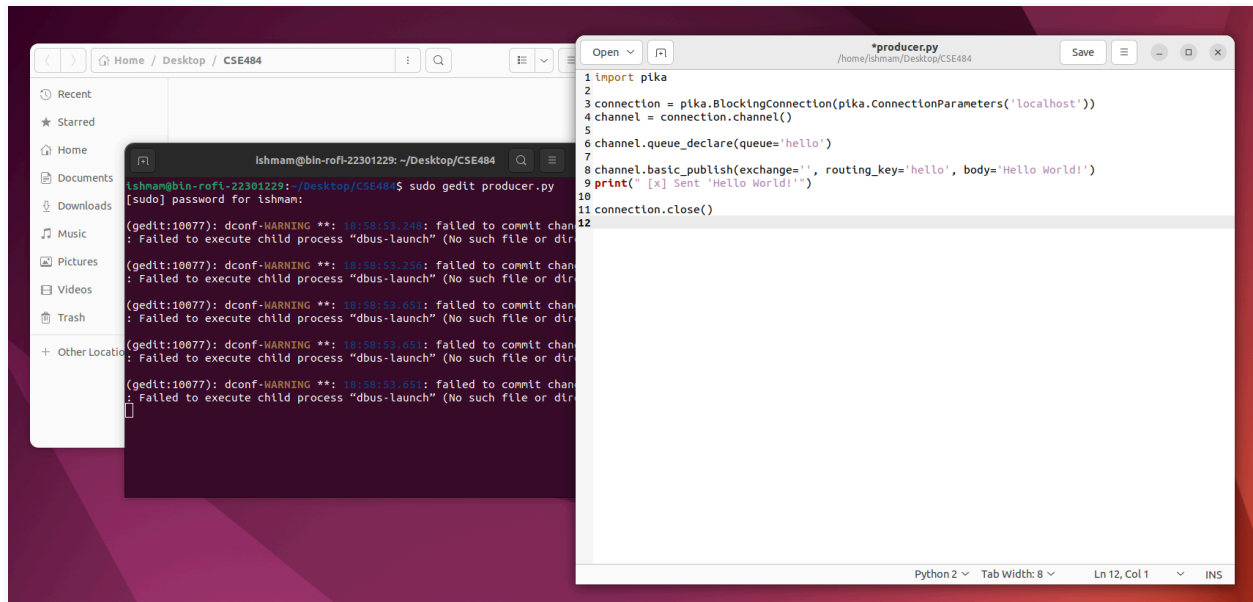
channel.queue_declare(queue='hello')
```

Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

```
channel.basic_publish(exchange='', routing_key='hello',
body='Hello World!')
print(" [x] Sent 'Hello World!'")

connection.close()'
```



2. Now lets open consumer.py

Here I will open a python script consumer.py in my same folder.

Here is the python script for the script.

Now we will open a python script on our system and then there we will copy paste our python script to the file and save it.

```
'import pika
```

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

```
channel.queue_declare(queue='hello')
```

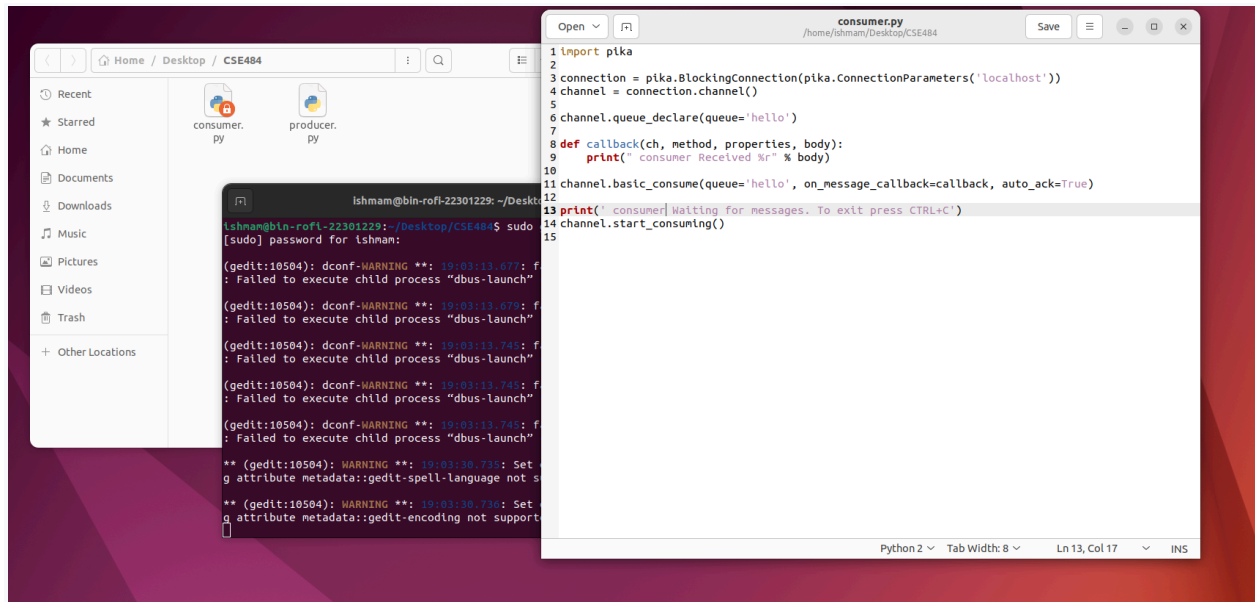
```
def callback(ch, method, properties, body):
    print(" consumer Received %r" % body)
```

Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

```
channel.basic_consume(queue='hello',  
on_message_callback=callback, auto_ack=True)
```

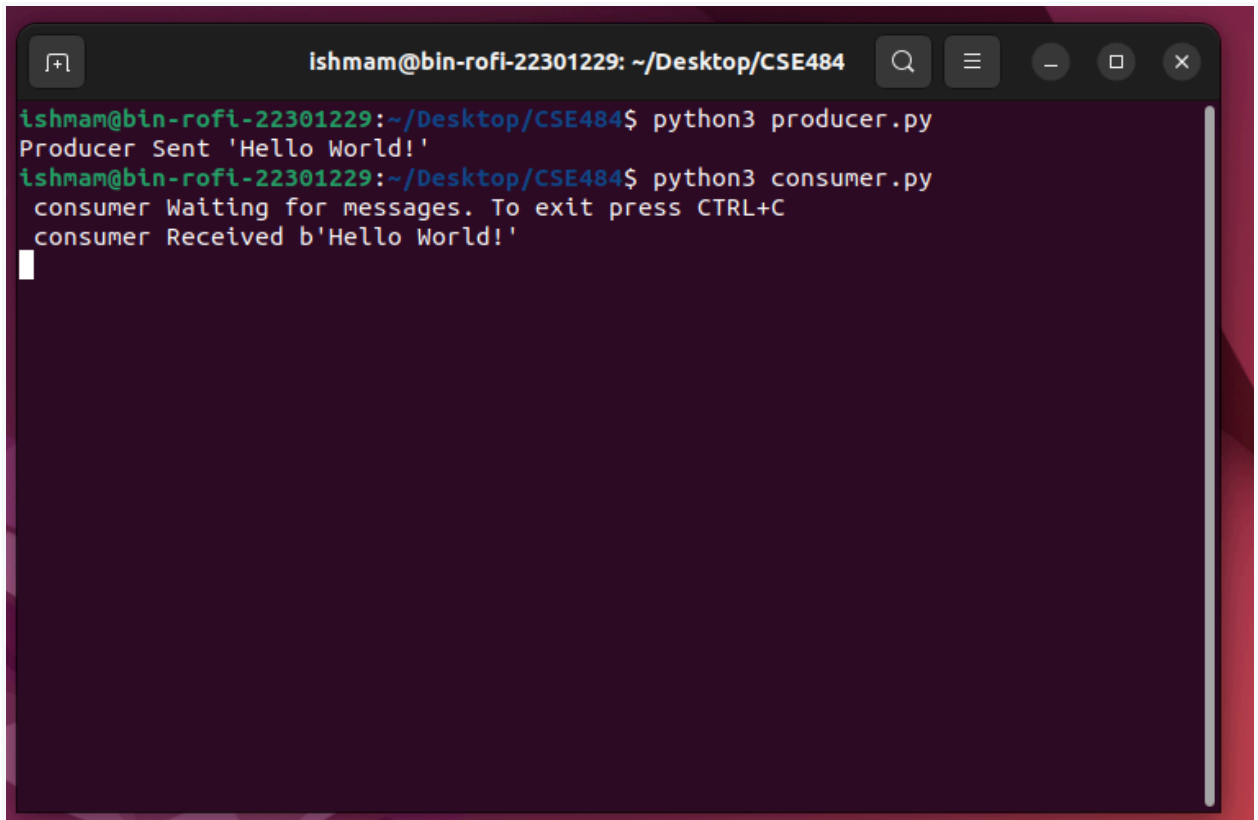
```
print(' consumer Waiting for messages. To exit press CTRL+C')  
channel.start_consuming()
```



3. Now running the python script

Now lets run run the python scripts on our terminal. It works ! So, we have successfully said hello world with our broker !

Ishmam Bin Rofi - 22301229
CSE484 - assignment 4

A terminal window with a dark background and light-colored text. The window title is "ishmam@bin-rofi-22301229: ~/Desktop/CSE484". The terminal shows the following commands and output:

```
ishmam@bin-rofi-22301229:~/Desktop/CSE484$ python3 producer.py
Producer Sent 'Hello World!'
ishmam@bin-rofi-22301229:~/Desktop/CSE484$ python3 consumer.py
consumer Waiting for messages. To exit press CTRL+C
consumer Received b'Hello World!'
```

Task 2: "Work queues"- Distributing tasks among workers
Following the same steps from the previous task,
Firstly, lets write our producer code python script on our
console. Here is the python code

```
'import sys
import pika

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

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

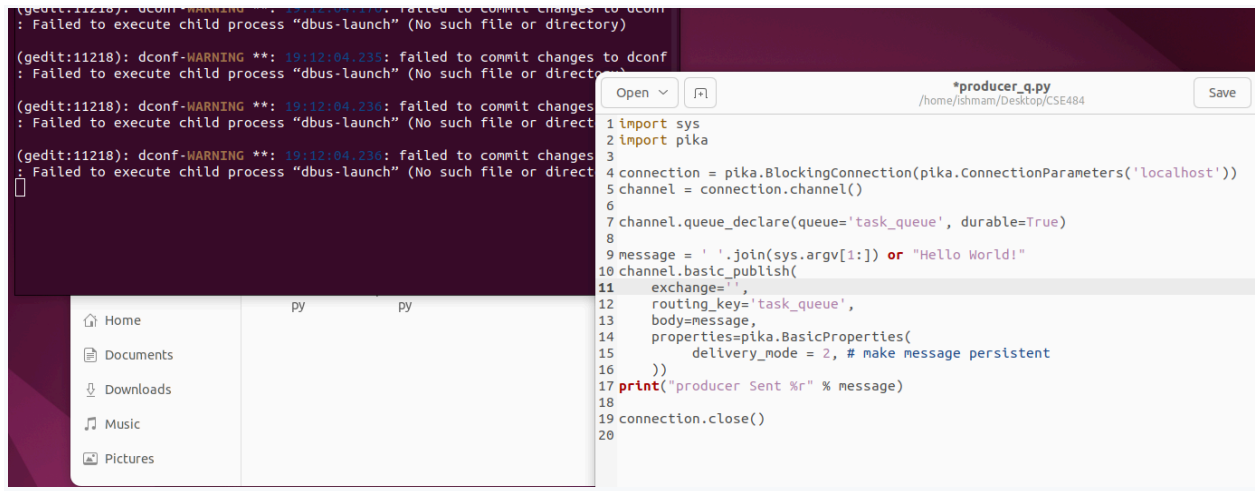
message = ' '.join(sys.argv[1:]) or "Hello World!"
channel.basic_publish(
```


Ishmam Bin Rofi - 22301229

CSE484 - assignment 4

```
        exchange='',
        routing_key='task_queue',
        body=message,
        properties=pika.BasicProperties(
            delivery_mode = 2, # make message persistent
        ))
print("producer Sent %r" % message)

connection.close()'
```



Now write the worker python script

```
'import time
import pika

def callback(ch, method, properties, body):
    print(" worker Received %r" % body)
    time.sleep(body.count(b'.'))
    print(" worker Done")
    ch.basic_ack(delivery_tag = method.delivery_tag)

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

channel.queue_declare(queue='task_queue', durable=True)
print('worker Waiting for messages. To exit press CTRL+C')
```

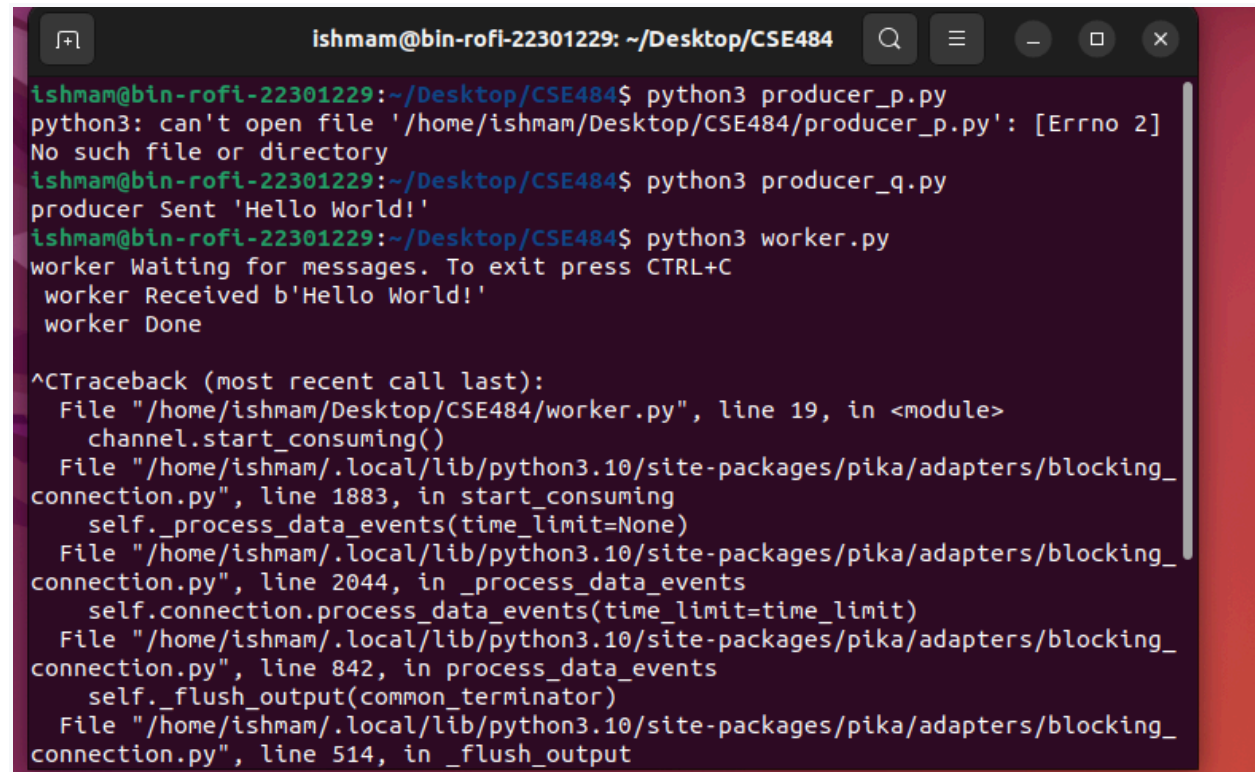
Ishmam Bin Rofi - 22301229
CSE484 - assignment 4

```
channel.basic_qos(prefetch_count=1)
channel.basic_consume(queue='task_queue',
on_message_callback=callback)

channel.start_consuming()'
```

Now lets run the python script

Here we can see that our python script is running properly !



```
ishmam@bin-rofi-22301229: ~/Desktop/CSE484
ishmam@bin-rofi-22301229:~/Desktop/CSE484$ python3 producer_p.py
python3: can't open file '/home/ishmam/Desktop/CSE484/producer_p.py': [Errno 2]
No such file or directory
ishmam@bin-rofi-22301229:~/Desktop/CSE484$ python3 producer_q.py
producer Sent 'Hello World!'
ishmam@bin-rofi-22301229:~/Desktop/CSE484$ python3 worker.py
worker Waiting for messages. To exit press CTRL+C
worker Received b'Hello World!'
worker Done

^CTraceback (most recent call last):
  File "/home/ishmam/Desktop/CSE484/worker.py", line 19, in <module>
    channel.start_consuming()
  File "/home/ishmam/.local/lib/python3.10/site-packages/pika/adapters/blocking_
connection.py", line 1883, in start_consuming
    self._process_data_events(time_limit=None)
  File "/home/ishmam/.local/lib/python3.10/site-packages/pika/adapters/blocking_
connection.py", line 2044, in _process_data_events
    self.connection.process_data_events(time_limit=time_limit)
  File "/home/ishmam/.local/lib/python3.10/site-packages/pika/adapters/blocking_
connection.py", line 842, in process_data_events
    self._flush_output(common_terminator)
  File "/home/ishmam/.local/lib/python3.10/site-packages/pika/adapters/blocking_
connection.py", line 514, in _flush_output
```