**Documentation for Micro Service integration with RabbitMQ**:

**Tools And Technologies :**

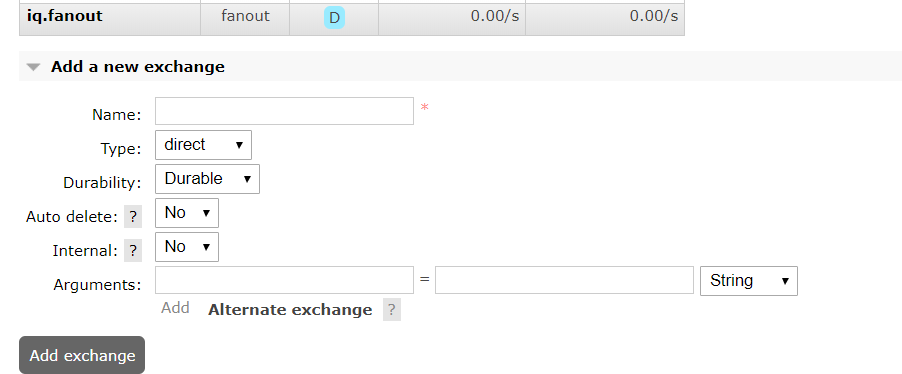
* Java 8 and above.
* Spring Boot
* Spring MVC,Rest webservices
* Rabbit MQ
* Junit
* Docker
* Log4j
* GitHub
* Postman

**Pre-Requisites:**

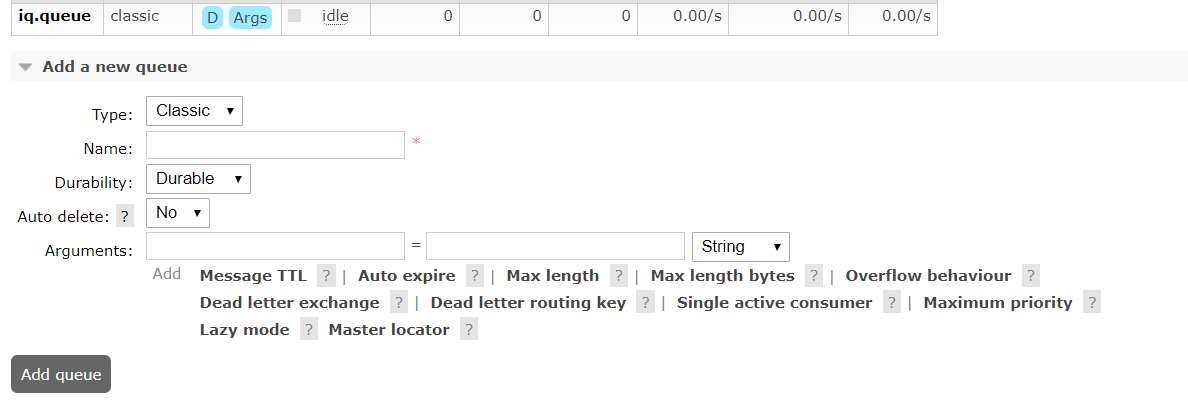
Download and Install the ERLANG, and set the ERLANG\_HOME path,

Download the RabbitMQ and install the RabbitMQ,

After Installation of RabbitMQ, Create the exchange in the RabbitMQ,



After creating the exchange create the Queue as below

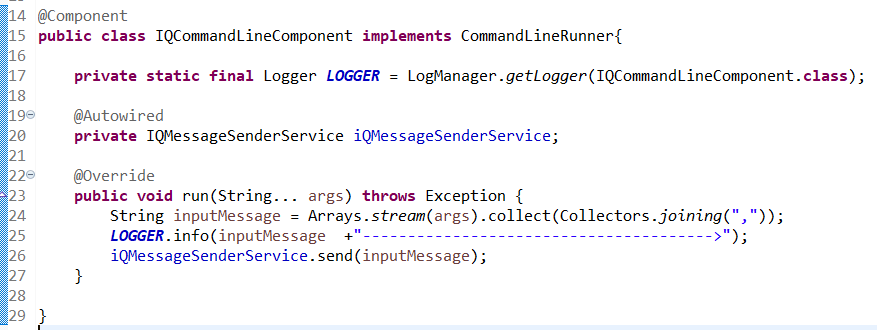


Download and install the Docker and setup the docker to run the two micro services applications.

Create two Spring boot applications (Micro services) for Producer and Consumer

**Message Producer Implementation:**

Create the command line IQCommandLineComponent class to take the arguments of messages from the command line



Create the IQMessageController to support the message using REST API



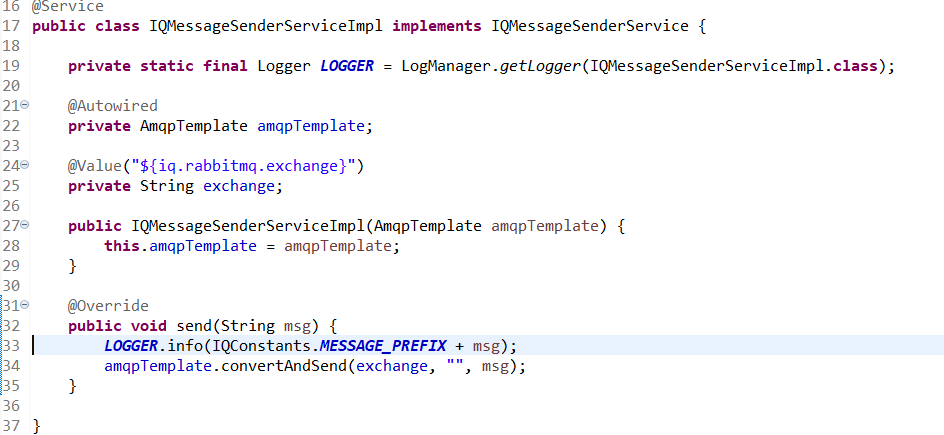
**URL Pattern:**

[http://localhost:8080/iq-rabbitmq/send?message={Name}](http://localhost:8080/iq-rabbitmq/send?message=%7bName%7d)

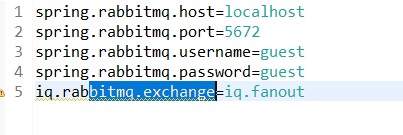
Sample URL: <http://localhost:8080/iq-rabbitmq/send?message=Andy>

Create the service and configuration properties to send the message to RabbitMQ,

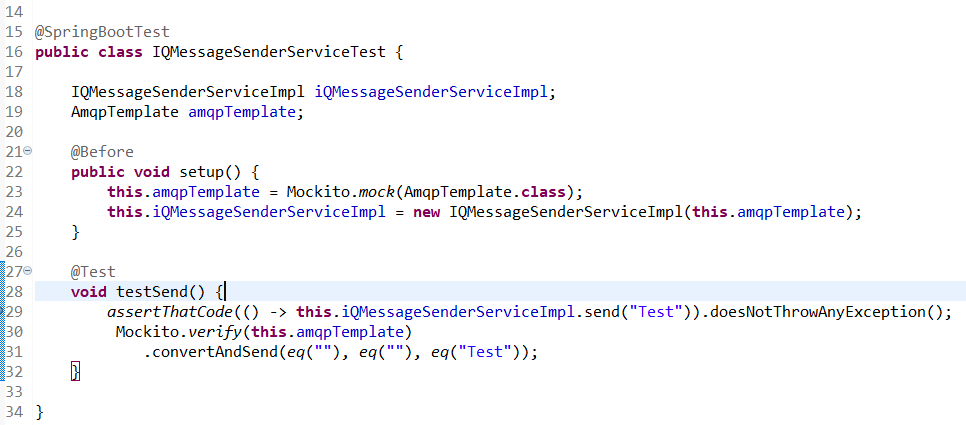
IQMessageSenderServiceImpl Service Class



Properties :

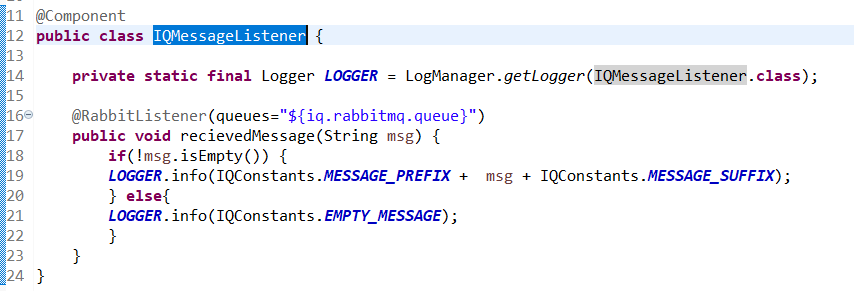


Create Unit Test for the Service Class :

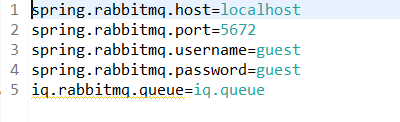


**Message Consumer Implementation:**

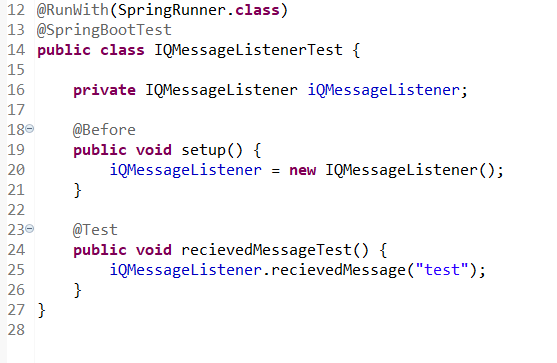
Create the IQMessageListener to listen the massage and configuration for RabbitMQ



Configuration proper:



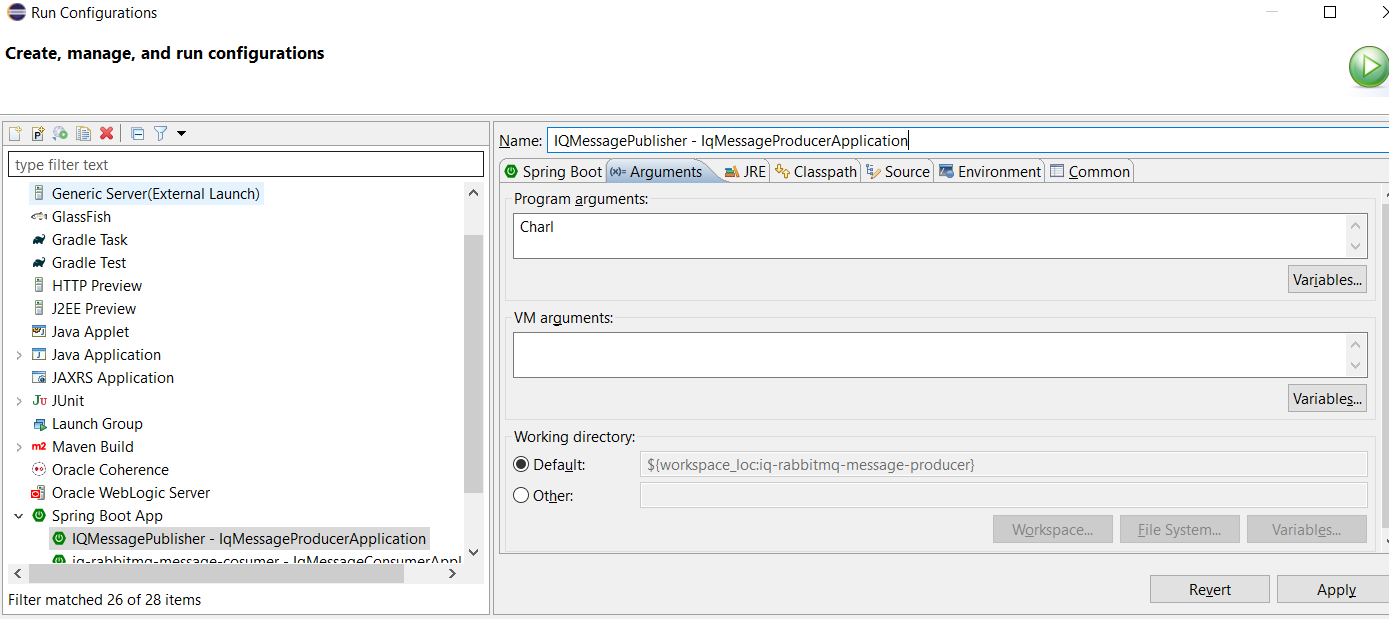
Create the unit test for the listener,



**Testing :**

Testing with command line arguments:

Providing the Charl name in the arguments and start the applications

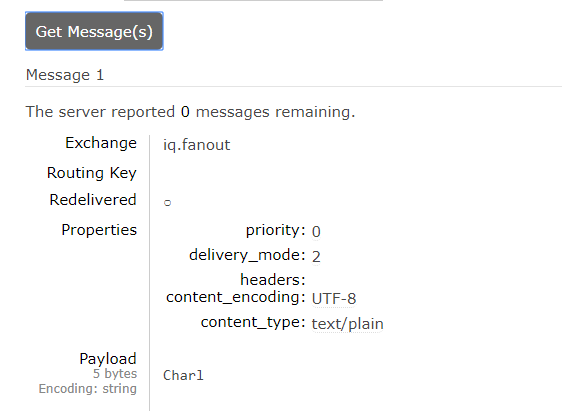


While starting of the application we can see the logs as below:

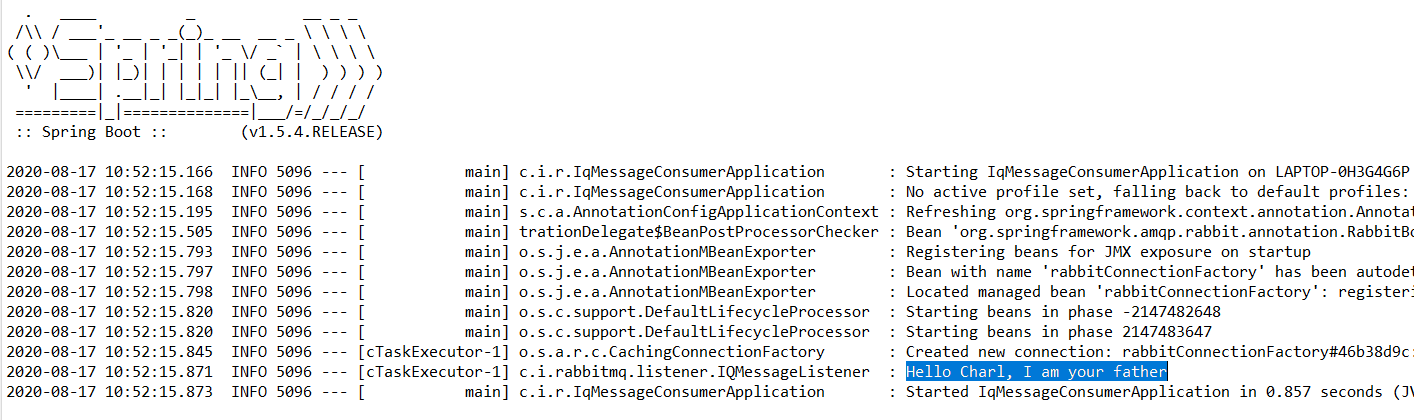


Now go to RabbitMQ login with credentials guest/guest, we can see the one message in the queue





Now lets start the consumer application and see the logs ,

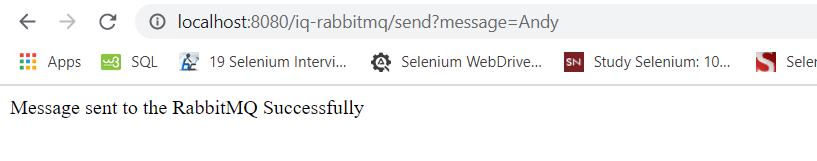


After consuming the message, if we verify the queue, it will be empty,

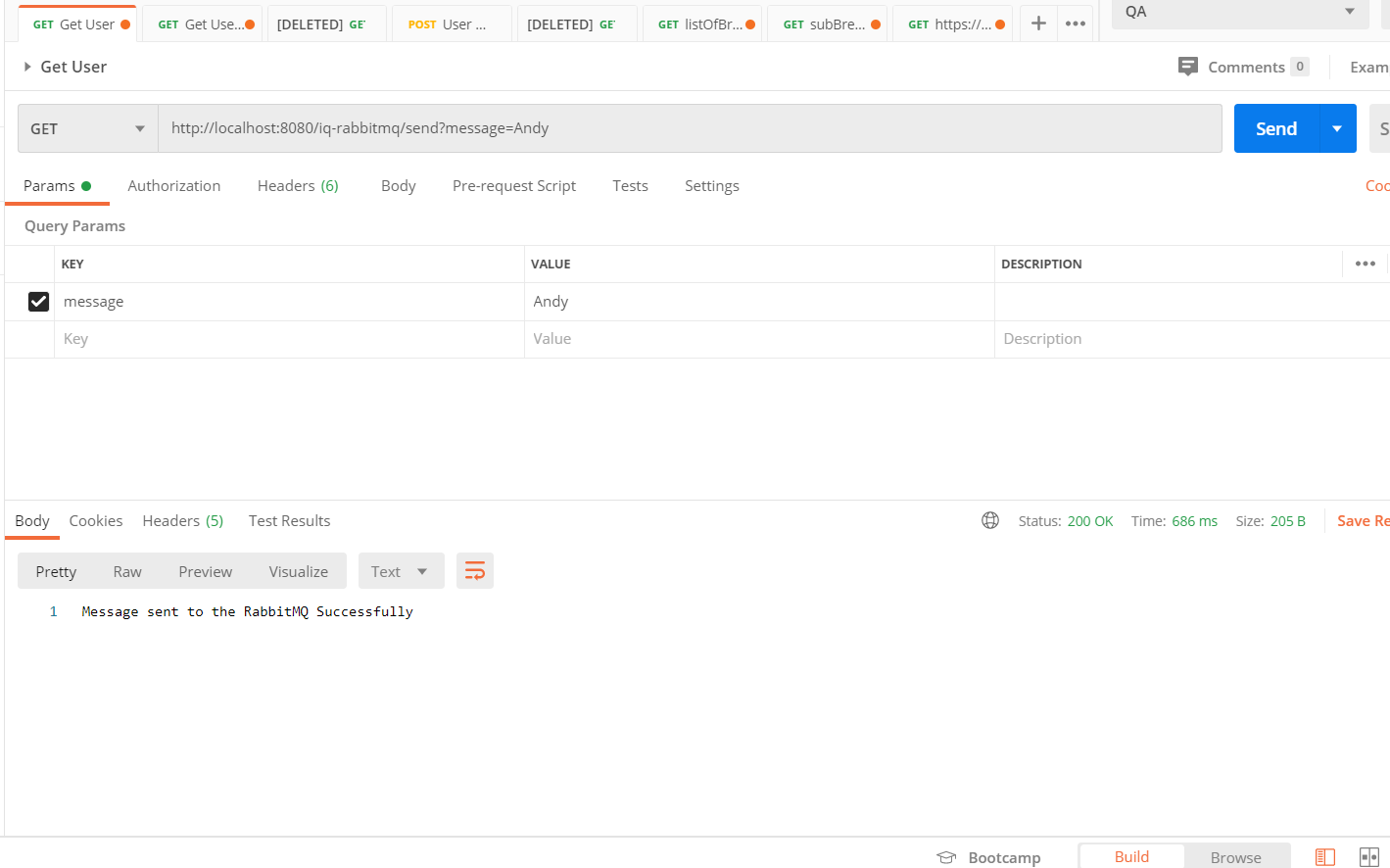


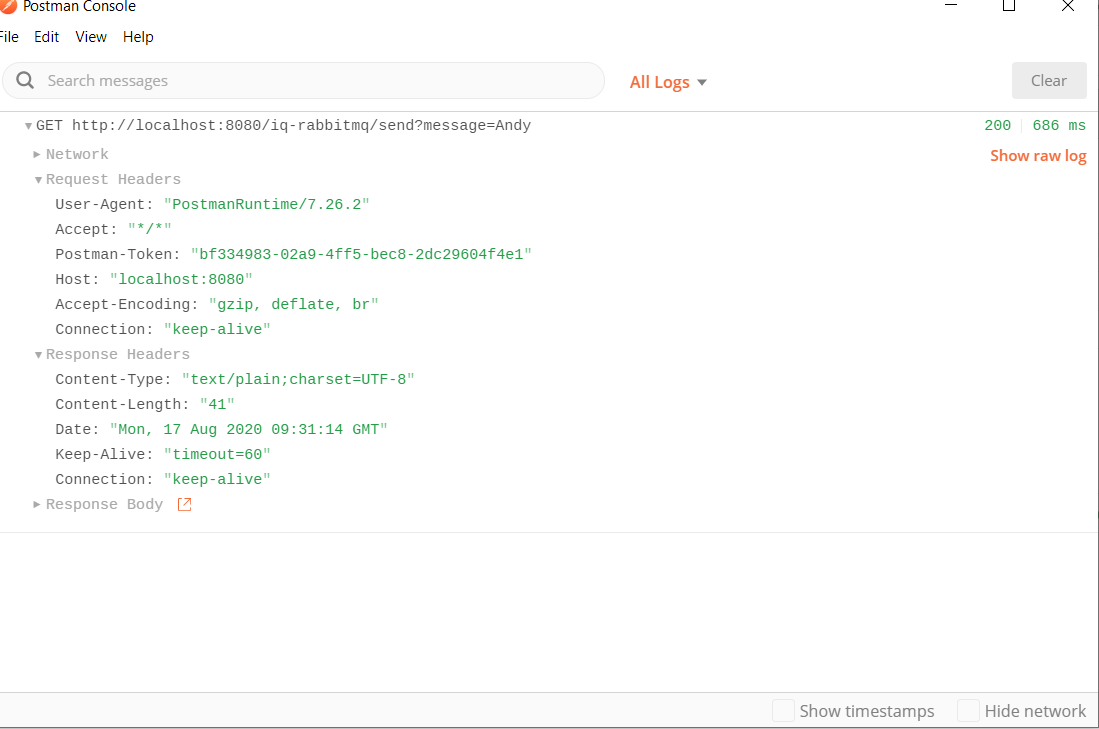
**Now let’s test it with Rest webservices:**

**Using Browser:**

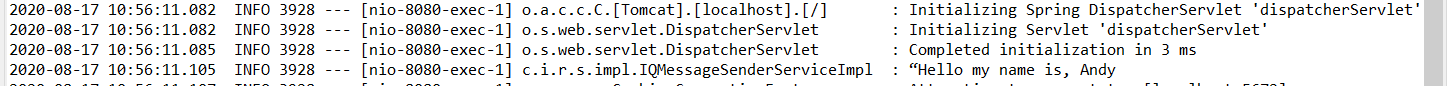


**Using Postman:**



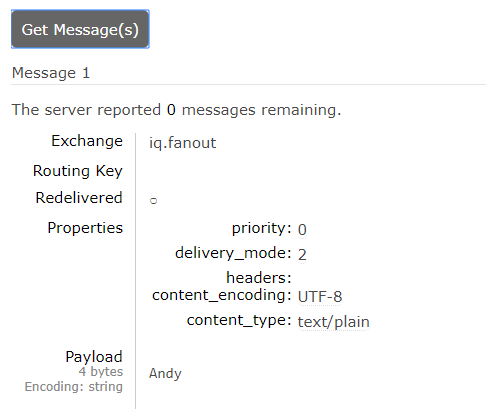


Verify the logs :

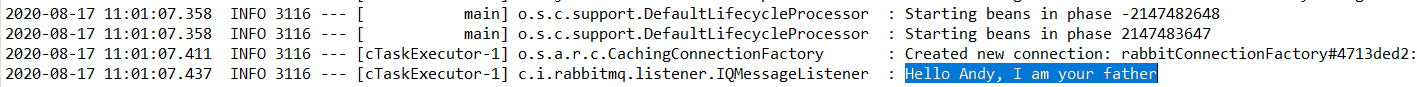


Verify the RabbitMQ queue,



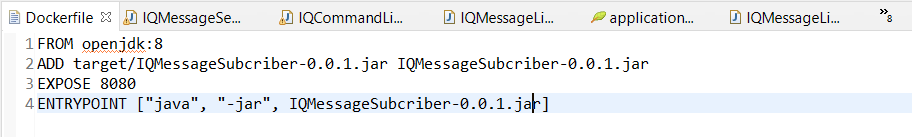


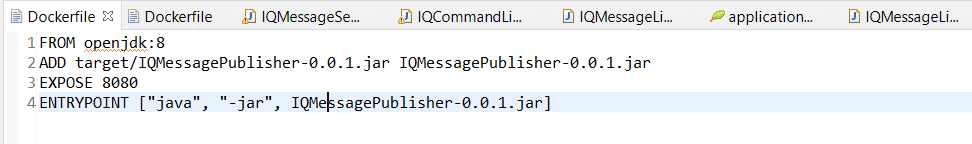
Now let’s start the consumer application and verify the logs and Queue,





**Created the Docker files for both applications:**





**Got to project folder and create the image inside the docker.**

docker build -f Dockerfile -t iqmessagepublisher .

docker build -f Dockerfile -t iqmessagecosumer .

**Run the image by using below command:**

docker run -p 8080:8080 iqmessagepublisher

docker run -p 8080:8080 iqmessagepublisher