RABBITMQ

RabbitMQ is an open-source message broker software that implements the Advanced Message Queuing Protocol(AMQP). This middleware is a popular, reliable, and scalable platform for building realtime applications.

RabbitMQ acts as a mediator between producers of messages and consumers who receive and process those messages. It facilitates the communication between different components of an application by enabling asynchronous messaging and decoupling the sender and the receiver. Thus, RabbitMQ receives messages from producers (senders) and deliver them to consumers (receivers). It uses a messaging queue, a buffer that holds messages until the consumers can process them.

Our application is composed by two micro services: Product and Shop, which communicate with each other. Product will be the producer and Shop will be the Consumer. Plus we have a graphical interface, which will trigger the communication between these two.

**Producer:**

A computer screen shot of a program code

Description automatically generated

Whenever a user, wants to find a product by its name, the Producer will send a message on the RabbitMQ, with the title of the product. Let’s analyse the code: first, we need to open a connection, by setting the host and the port, and to create a channel. Then, we define the queue “product\_queue”. Afterwards, the message is sent on the queue and the connection is closed.

A screenshot of a computer

Description automatically generated

We can also visualise the queue by accessing localhost:15672 with the credentials: username = guest, password = guest.

**Consumer**

A screen shot of a computer program

Description automatically generated

The consumer continuously listen for messages sent by the producer. Again, we need to open a connection by setting the hostname and the port, and to create a channel. Then, we define the consumer which will handle the deliveries by printing the message on the console. Afterwards, the message is consumed from the queue.

**Example:**

From the graphical user interface, a user introduce the name of the product, in order to retrieve its details.

A screenshot of a computer

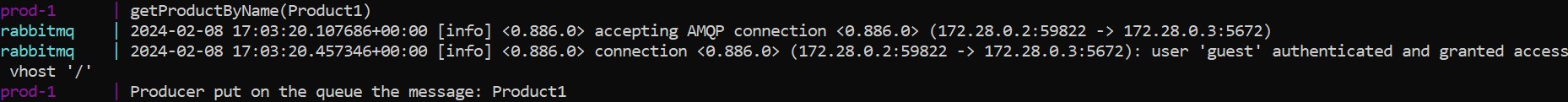
Description automatically generated

If we access localhost:15672, we can observe that a message appeared on the queue.

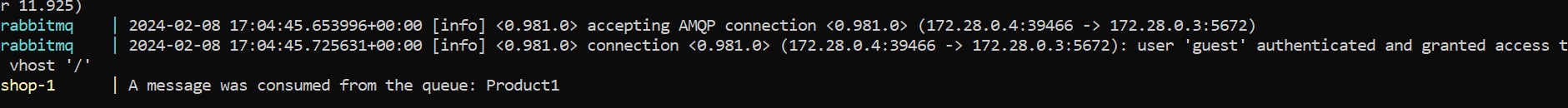
A screenshot of a computer

Description automatically generated

Also, we can observe in the logs, that the producer connected to the queue, and the message that was sent:



Next, the consumer will take the message from the queue and print it on the console.



If we look again on localhost:15672, we can observe that the message is no longer on the queue.

A screenshot of a computer

Description automatically generated