**Approach -1**

Install Erlang

<https://github.com/erlang/otp/releases/tag/OTP-26.0.2>

Download rabbitmq-server-3.12.1.exe

<https://www.rabbitmq.com/install-windows.html>

Go to the rabitMq CLI tool and enable it

rabbitmq-plugins enable rabbitmq\_management

npm init -y

Then lunch rabitMq using <http://localhost:15672/>

Login: guest

Password: guest

Approach -2

Instruction from

<https://www.youtube.com/watch?v=gntSKlXedvE&ab_channel=BigTechTalk>

Install using docker

1. Install docker software
2. Go to hub.docker.com and search for rabbitmq
3. Select first option: Rabbit official image and click
4. Copy the name 3.11.19-management
5. Now pull this image using docker
6. Go to cmd prompt and type docker pull rabbitmq:3.11.19-management.
7. Now run this image and configure ports using command

docker run -d --hostname rabbit --name rabbit-server -p 15672:15672 -p 5672:5672 rabbitmq:3.11.19-management

15672 is for the web UI port

5672 is for rabbit broker port

1. Now open browser and type <http://localhost:15672/>. Clear brower catch memory if you can’t see the login page

Now create a project folder

The go to cmd

Npm init -y

Then code .

Coding tutorials

<https://www.youtube.com/watch?v=Qbs53khPoQs&ab_channel=ItsCodingTime>

<https://www.youtube.com/watch?v=vMFcayVfFvM&t=301s&ab_channel=TechnicalBabaji>

<https://www.rabbitmq.com/tutorials/tutorial-one-javascript.html>

Coding:

1. Initialize the package manager using npm

npm init -y

This will generate the package json file

1. Install AMQP library to the project.

npm install amqplib

1. Create sendor.js file. This will act as a producer to send message to the queue
2. Now import the library

const amqp = require('amqplib/callback\_api');

1. Connect to RabbitMQ management and the following code for sensor.js.

const amqp = require('amqplib/callback\_api');

//Step 1: Create Connection

amqp.connect('amqp://localhost', (connError, connection)=> {

if(connError){

throw connError;

}

//Step 2: Create Channel

connection.createChannel((channelError, channel) => {

if(channelError){

throw channelError;

}

//Step 3: Assert Queue

const QUEUE = 'codingtest'

channel.assertQueue(QUEUE);

//Step 4: Send message to queue

channel.sendToQueue(QUEUE,Buffer.from('hello from Yuba'));

console.log('Message send ${QUEUE}');

})

})

1. Create receiver.js file

const amqp = require('amqplib/callback\_api');

// Step 1: Create Connection

amqp.connect('amqp://localhost', (connError, connection) => {

if (connError) {

throw connError;

}

// Step 2: Create Channel

connection.createChannel((channelError, channel) => {

if (channelError) {

throw channelError;

}

// Step 3: Assert Queue

const QUEUE = 'codingtest'

channel.assertQueue(QUEUE);

// Step 4: Send message to queue

channel.sendToQueue(QUEUE, Buffer.from('hello from its coding time'));

console.log(`Message send ${QUEUE}`);

})

})

Program -2

Publisher.js

const amqp = require('amqplib/callback\_api');

amqp.connect('amqp://localhost', (err, connection) => {

if(err){

throw err;

}

connection.createChannel((err,channel) => {

if(err){

throw err;

}

let queueName = "chat";

let message = " This is a chat program";

channel.assertQueue(queueName, {

durable: false

});

channel.sendToQueue(queueName, Buffer.from(message));

console.log(`Message: ${message}`);

setTimeout(() => {

connection.close();

}, 1000)

})

})

Subscriber.js

const amqp = require('amqplib/callback\_api');

amqp.connect('amqp://localhost', (err, connection) => {

if(err){

throw err;

}

connection.createChannel((err,channel) => {

if(err){

throw err;

}

let queueName = "chat";

channel.assertQueue(queueName, {

durable: false

});

channel.consume(queueName,(msg) => {

console.log(`Received : ${msg.content.toString()}`);

channel.ack(msg);

})

})

})