Coding Challenge – GoGET

Strictly Confidential

.NET C# Coding Challenge

Background

Devices in the Goget system can be communicated with over a cellular network. Connections to these devices can be initiated from the server by phoning their SIM card numbers.

Task

Using C# and .NET Framework 4.0 or greater but not .NET Core, create a way of initiating a connection to these devices by consuming a RabbitMQ AMQP message (this message can be published from anywhere, the details of that don't matter to the consumer), make an API call to get the data required to initiate the connection and then using the required data call a dummy 3rd party API to make the connection.

There are three parts to this problem

1. Creation of a consumer to process an AMQP message:

Using a RabbitMQ client (Nuget package RabbitMQ.Client) create a consumer that will consume a json message like this DeviceNetworkConnectJob{"deviceId":6}. This consumer should run inside a Console app that can be started from the command line.

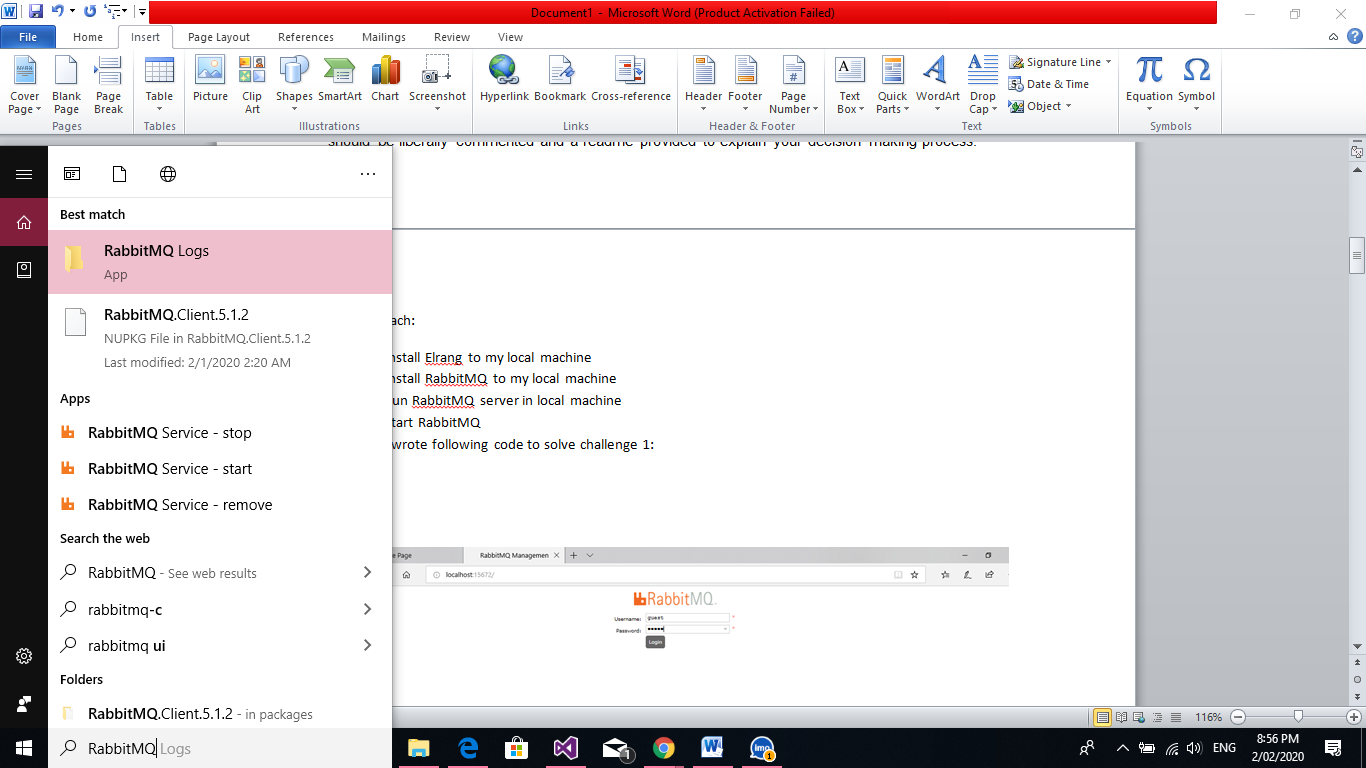
2. Creation of a REST API endpoint to return required data:

Using .NET Web API create an endpoint like this http://localhost:50010/DeviceDetails/5 that will return Details for the requested Device from the attached simple text file. This simple file is provided to make implementation and testing easier but the solution provided should require few changes to also work with extracting this data from a database table. Text file attached - SimpleDB.csv

3. Inside the consumer, make a call to the endpoint created in 2 and with the returned data, call the fake 3rd party API MakeConnection(string Details) to complete the DeviceNetworkConnectJob consumption.

General Expectations

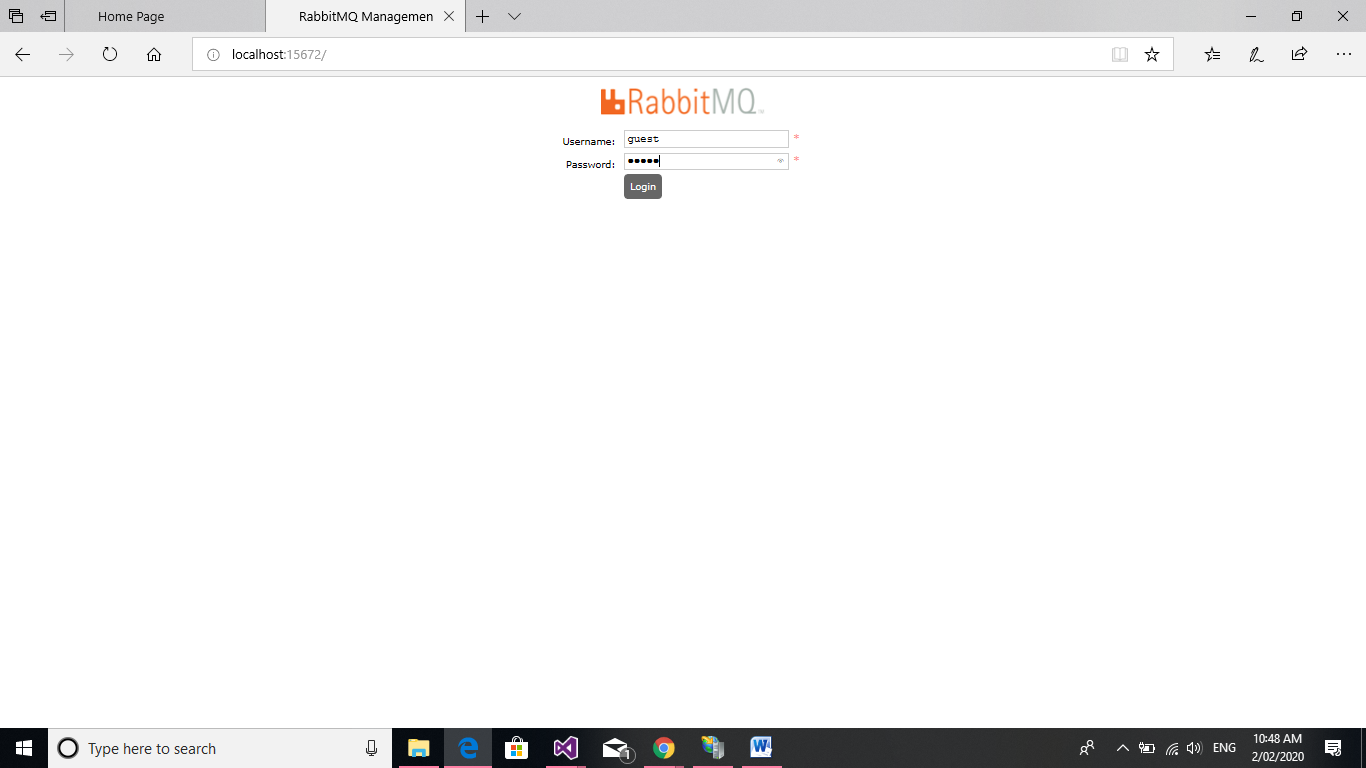
The Visual Studio projects created for this challenge should be contained in a single Visual Studio solution file. This can be uploaded to a private github account or zipped and emailed. The code should be liberally commented and a readme provided to explain your decision making process.



My Approach:

1. Install Elrang to my local machine
2. Install RabbitMQ to my local machine
3. Run RabbitMQ server in local machine
4. Start RabbitMQ
5. Enter RabbitMQManagement : Username – guest, Password – guest
6. Write code to consume msg for RabbitMQ
7. Write code for API to collect data from CSV
8. Setup API into IIS server and run from IIS.
9. Write code for collect data from local host(<http://localhost/api/DeviceDetails/{deviceID}>) to pass on RabbitMQ by using Consumer.
10. Write method to show in console that message has transferred successfully
11. Check in RabbitMQ management for test.

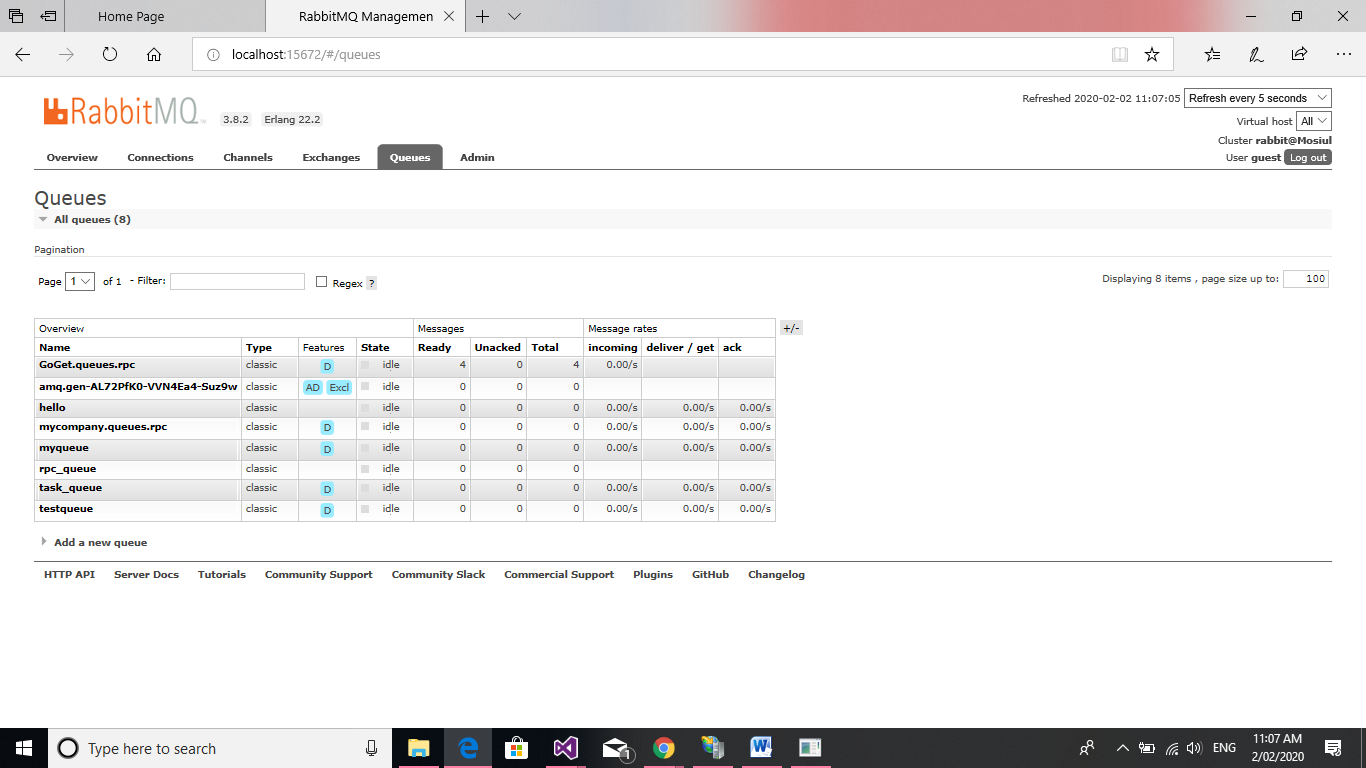
[Important Note: Please publish your API in IIS server first. Put you file in C:/inetpub/wwwroot ]



In RabbitMQ to check if Queues are working Click on

-> Queues : you will find all the Queues which has created and their status..

SCREENSHOTS:



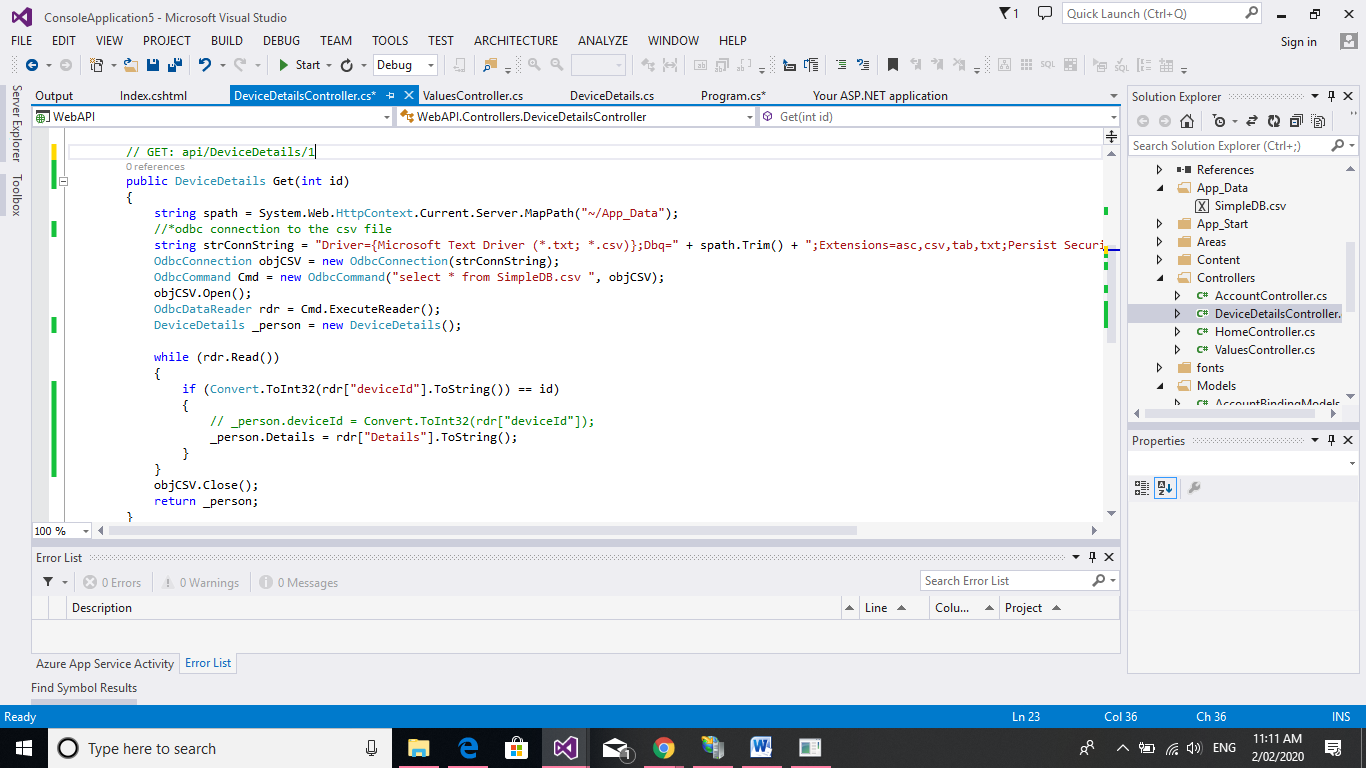
Creating API:

Database is CSVfile which for this program I stored in App\_data Folder in local directory.

To convert into json response I have added this code in WebApiConfig.cs file

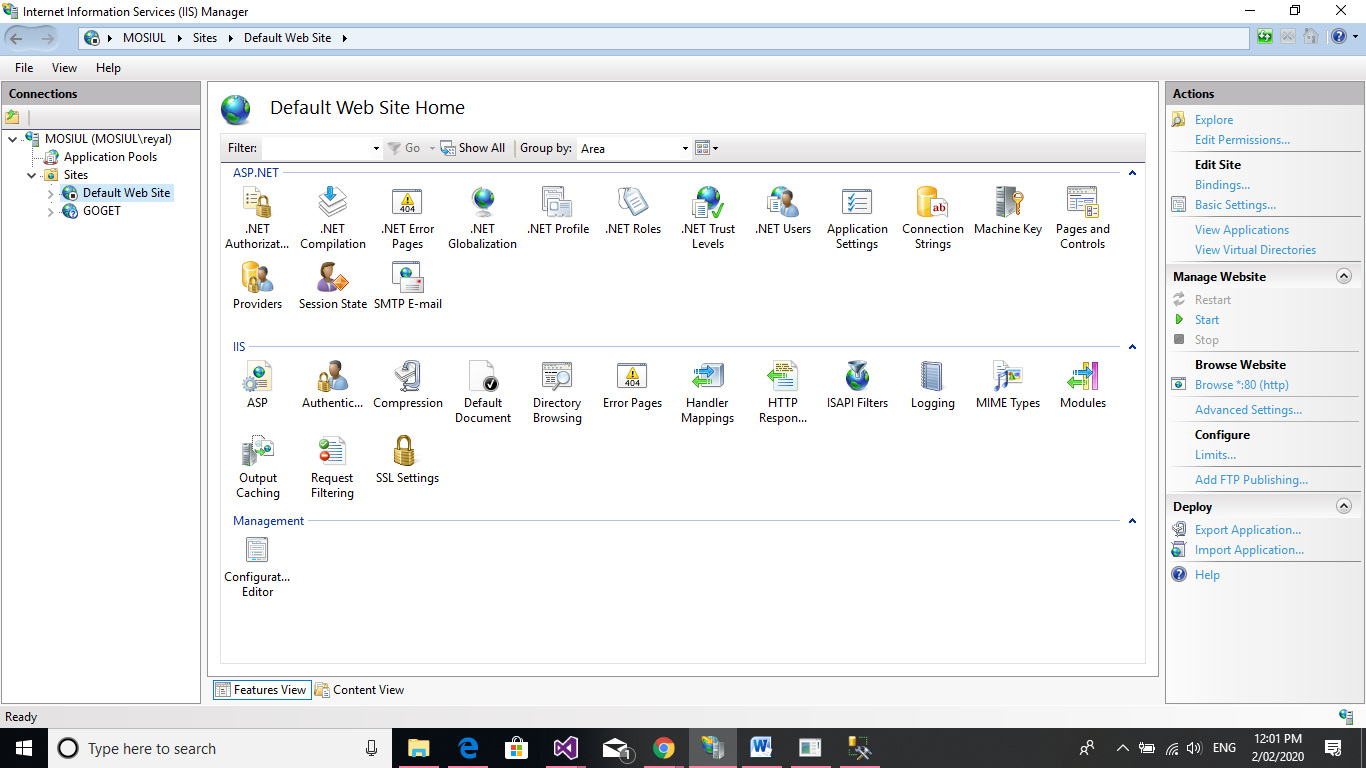
//Configure Json response

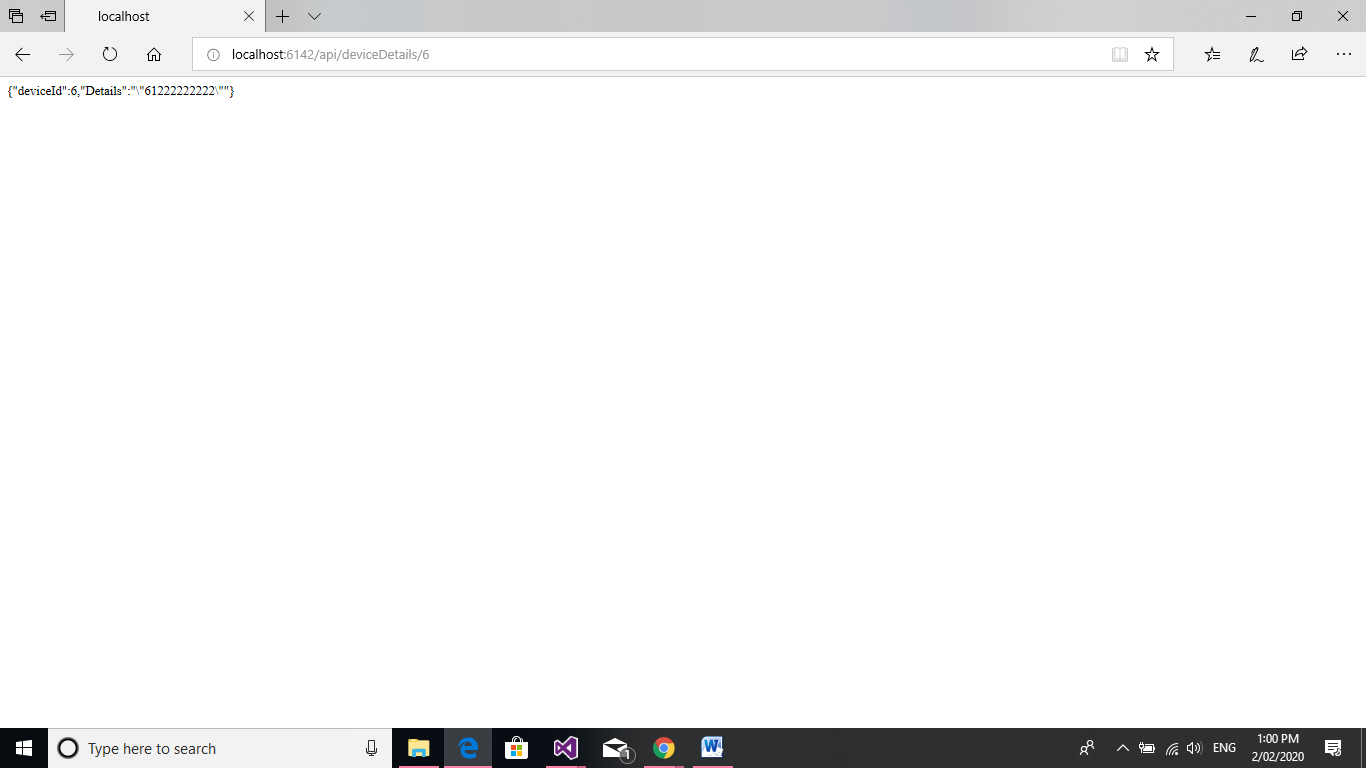
config.Formatters.JsonFormatter.SupportedMediaTypes.Add(new MediaTypeHeaderValue("text/html"));



Step 3:

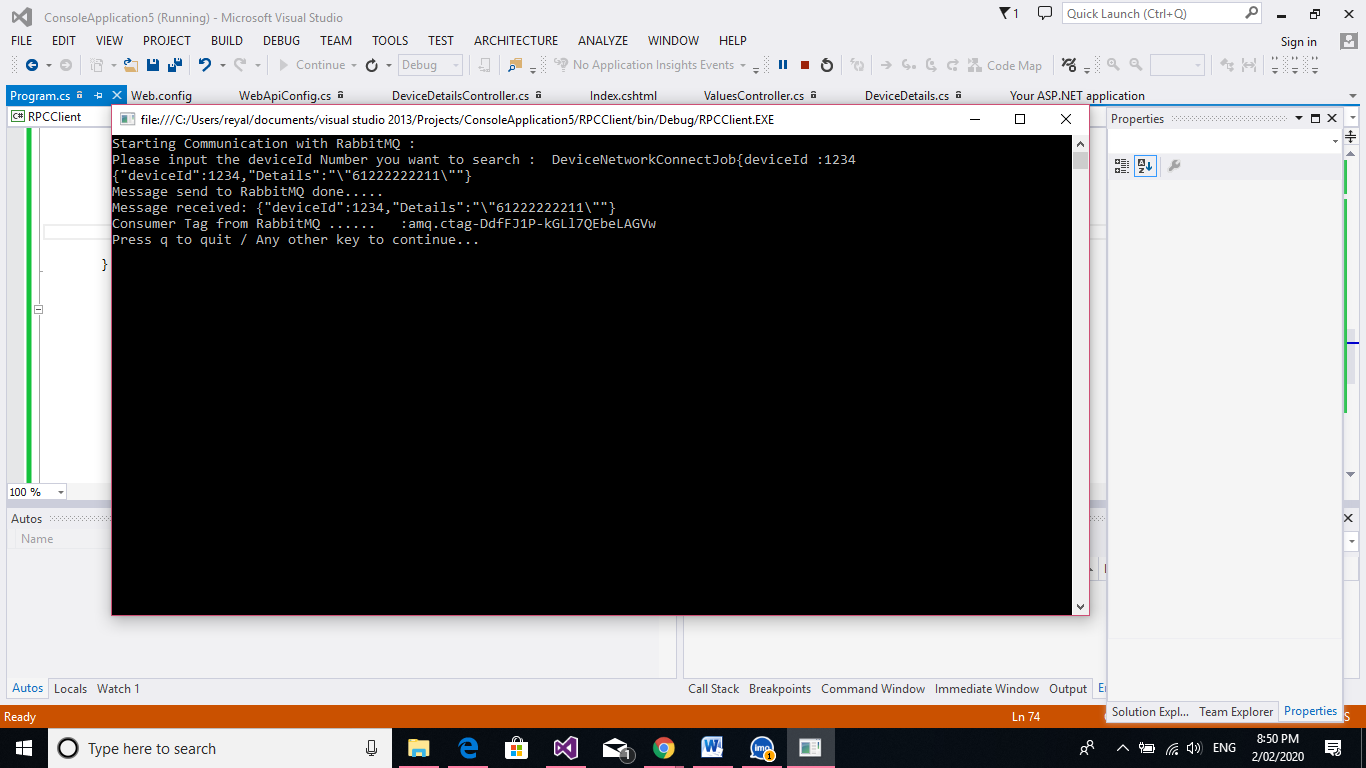
Publish API in windows IIS server.

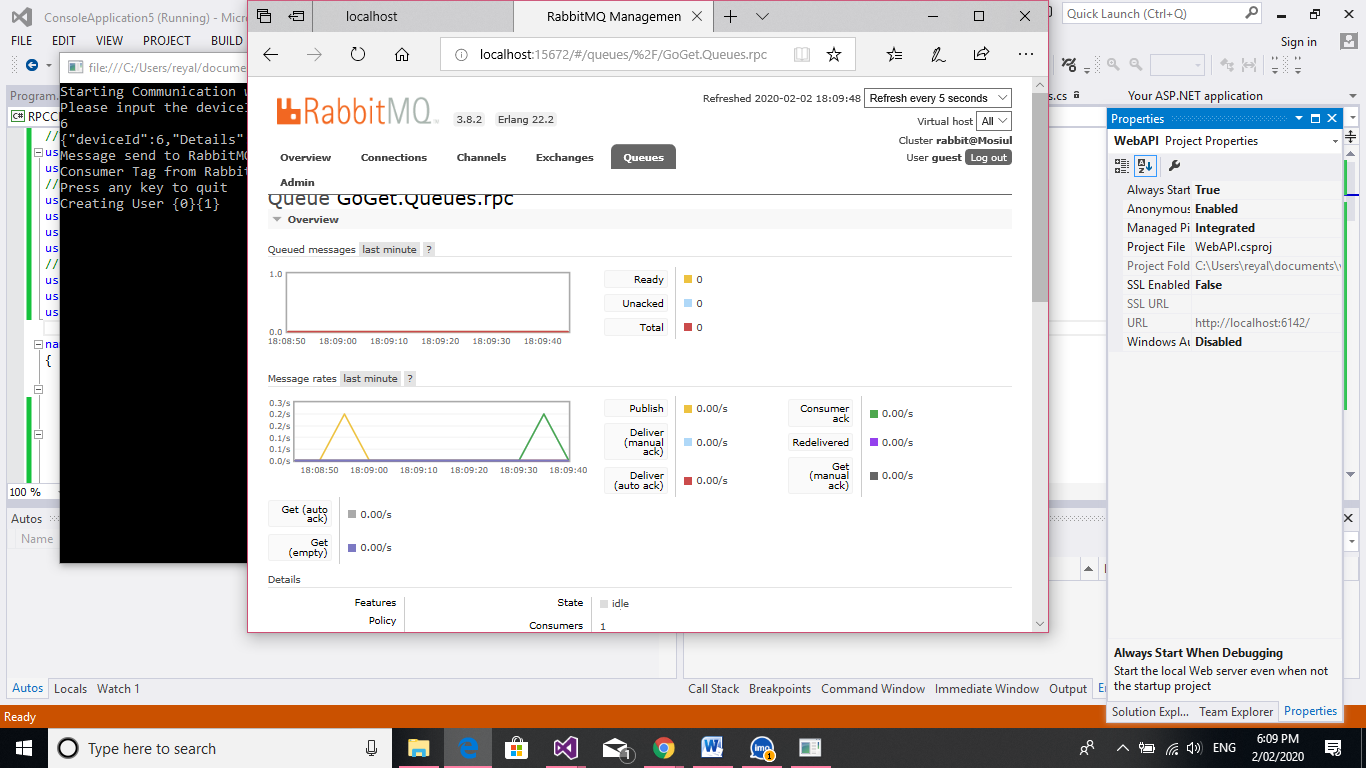




Browser is trying to connect through API

Final Output:





Code for Console Application:

//To use RabbitMQ

using RabbitMQ.Client;

using RabbitMQ.Client.Events;

//Default C# application

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

//For multithreading using API..

using System.Net;

using System.Text;

using System.Threading.Tasks;

namespace RPCClient

{

class Program

{

static void Main(string[] args)

{

while (true)

{

Console.Clear();

// \*For connecting RabbitMQ

Console.WriteLine("Starting Communication with RabbitMQ :");

var factory = new ConnectionFactory()

{

HostName = "localhost",

Port = 5672,

UserName = "guest",

Password = "guest"

};

using (var connection = factory.CreateConnection())

using (var channel = connection.CreateModel())

{

// \*.............................. For declaring Queues ...................................................\*

channel.QueueDeclare(queue: "GoGet.Queues.rpc",

durable: false,

exclusive: false,

autoDelete: false,

arguments: null);

//\*...............................Sending Message.to Queues.....................................................\*

var methodCall = new Program();

//\* APIMSG() method is calling fake API which has stored in IIS server in local machine...

string message = methodCall.ApIMSG();

Console.WriteLine(message);

var body = Encoding.UTF8.GetBytes(message);

channel.BasicPublish(exchange: "",

routingKey: "GoGet.Queues.rpc",

basicProperties: null,

body: body);

Console.WriteLine("Message send to RabbitMQ done.....");

//\*.......................................Msg Delivering from Queues...........................................\*

channel.BasicQos(0, 1, false); //basic quality of service

QueueingBasicConsumer consumer = new QueueingBasicConsumer(channel);

channel.BasicConsume("GoGet.Queues.rpc", false, consumer);

BasicDeliverEventArgs deliveryArguments = consumer.Queue.Dequeue() as BasicDeliverEventArgs;

String message1 = Encoding.UTF8.GetString(deliveryArguments.Body);

Console.WriteLine("Message received: {0}", message1);

channel.BasicAck(deliveryArguments.DeliveryTag, false);

String consumerTag = channel.BasicConsume("GoGet.Queues.rpc", false, consumer);

//\*.................... Printing Tag..........................................................................\*

Console.WriteLine("Consumer Tag from RabbitMQ ...... :" + consumerTag);

connection.Close();

// Console.ReadKey();

}

Console.WriteLine("Press q to quit / Any other key to continue... ");

string message2 = Console.ReadLine();

if (message2.ToLower() == "q") break;

}

}

private string ApIMSG()

{

//\*.................... Reading From API..........................................................................\*

Console.Write("Please input the deviceId Number you want to search : DeviceNetworkConnectJob{"+"deviceId :"+"");

string deviceIds = Console.ReadLine();

string listn = String.Format("Http://localhost:6142/api/deviceDetails" + "/" + deviceIds);

WebRequest requestObj = WebRequest.Create(listn);

requestObj.Method = "GET";

HttpWebResponse response = null;

response = (HttpWebResponse)requestObj.GetResponse();

string strResult = null;

using (Stream stream = response.GetResponseStream())

{

StreamReader sr = new StreamReader(stream);

strResult = sr.ReadToEnd();

sr.Close();

}

return strResult;

}

}

}

Code for API(IIS server/Local Machine/Networked machine)

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Odbc;

using System.Data.OleDb;

using System.Linq;

using System.Net;

using System.Net.Http;

using System.Web.Http;

using System.Web.Script.Serialization;

namespace WebAPI.Controllers

{

public class DeviceDetailsController : ApiController

{

// GET: api/GoGet

public IEnumerable<string> Get()

{

return new string[] { "value1", "value2" };

}

// GET: api/DeviceDetails/1

public DeviceDetails Get(int id)

{

string spath = System.Web.HttpContext.Current.Server.MapPath("~/App\_Data");

//\*odbc connection to the csv file

string strConnString = "Driver={Microsoft Text Driver (\*.txt; \*.csv)};Dbq=" + spath.Trim() + ";Extensions=asc,csv,tab,txt;Persist Security Info=False";

OdbcConnection objCSV = new OdbcConnection(strConnString);

OdbcCommand Cmd = new OdbcCommand("select \* from SimpleDB.csv ", objCSV);

objCSV.Open();

OdbcDataReader rdr = Cmd.ExecuteReader();

DeviceDetails \_person = new DeviceDetails();

JavaScriptSerializer json = new JavaScriptSerializer();

while (rdr.Read())

{

if (Convert.ToInt32(rdr["deviceId"].ToString()) == id)

{

\_person.deviceId = Convert.ToInt32(rdr["deviceId"]);

\_person.Details = json.Serialize(rdr["Details"].ToString());

}

}

objCSV.Close();

return \_person;

}

// POST: api/GoGet

public void Post([FromBody]string value)

{

}

// PUT: api/GoGet/5

public void Put(int id, [FromBody]string value)

{

}

// DELETE: api/GoGet/5

public void Delete(int id)

{

}

}

}

Mosiul Haque Khan

Thank you