**Steps for Micro-backend**

**Step 1:**Open Visual Studio 2022

**Step 2:**Create Project with a blank solution (named Evochat in our case)

**Step 3:**Create folder named src under blank solution then create Services folder under src

**Step 4:** Right click on services folder and Add New Asp.net core 6 webapi project Like Products so name enter like Evochat.Services.Products and so on

**Step 5:**Add empty class library project named as Evochat.Shared i.e Mainproject.Shared for shared things and add shared library into your projects created in **step 4**

**Step 6:**For Gateway add folder named as APIGateway under main project then add asp.net core 6 empty web application

The common nuget package needed for our application are

* AutoMapper
* AutoMapper.Extensions.Microsoft.DependencyInjection
* Microsoft.AspNetCore.Authentication.JwtBearer
* Microsoft.EntityFrameworkCore
* Microsoft.EntityFrameworkCore.SqlServer
* Microsoft.EntityFrameworkCore.Tools
* Swashbuckle.AspNetCore. Annotations
* Swashbuckle.AspNetCore.SwaggerUI

For other packages like RabbitMQ & masstransit follow the following steps

**How to set-up the environment?**

**Install Erlang:**So, first-of-all, we need to install Erlang according to our machine. Because Erlang is a programming language and the RabbitMQ server is built on it. So, click [here](https://www.erlang.org/downloads)to download the latest installer according to your machine.

**Note: – make sure you have installed it with admin rights.**

**Install RabbitMQ:**Now, in this step, we will install the RabbitMQ server. So, click [here](https://www.rabbitmq.com/install-windows.html) to download the latest installer according to your machine.

**Note: – Make sure you have installed it with admin rights.**

**Enable RabbitMQ Management Plugin Portal in windows:**Once the RabbitMQ server is installed, then we need to activate the Management portal. So, open the command prompt with admin rights and then go to the below location as you do see below in the screenshot.

C:\Program Files\RabbitMQ Server\rabbitmq\_server-3.8.11\sbin

Now, run the below command as you do see below in the screenshot. this command will actually run the **rabbitmq-plugins** batch file which is present inside the **sbin**folder.

rabbitmq-plugins enable rabbitmq\_management

Now, go to this URL (<http://localhost:15672/>), and then you will see the management portal is up and running.

**Username:**guest

**Password:**guest

below two commands to restart the RabbitMQ service.

net stop RabbitMQ

net start RabbitMQ

## ****What is MassTransit?****

It is a lightweight service bus for building distributed .Net applications. The main goal of MassTransit is to provide a consistent, .NET friendly abstraction over the message broker services (whether it is **RabbitMQ, Azure Service Bus,**etc). It is very helpful for routing messages over RabbitMQ, MSMQ, TIBCO, and ActiveMQ service busses with native support for **RabbitMQ**and MSMQ. It provides us lots of features like multicast, versioning, encryption, sagas, transactions, distributed systems, retries, and many other features.

So, we will use MassTransit to publish and receive the messages from the RabbitMQ server.

### **Install MassTransit Packages.**

So, now in this section we will install MassTransit in our both services (Customer Microservice, Order Microservice). So, go to **Manage NuGet Packages For Solution…**and then install these below packages as you see below in the screenshot.

MassTransit

MassTransit.RabbitMQ

MassTransit.AspNetCore