**วิธีการทำ Real time web application**

คู่มือนี้สามารถนำไปใช้งานได้ทั้ง React, angular, vue

\*\* หากต้องการที่จะพัฒนาเว็บแอพลิเคชั่น ให้มีความ Real time โดยที่ใช้เครื่องมือ Front-end + Back-end

สำหรับ Package ที่ใช้งาน คือ

**- Front-end** คือ @microsoft/signalr <support javascript library เท่านั้น>

yarn add @microsoft/signalr

\*\* หากติดตั้ง package แล้วติด error ว่า **The engine "node" is incompatible with this module**

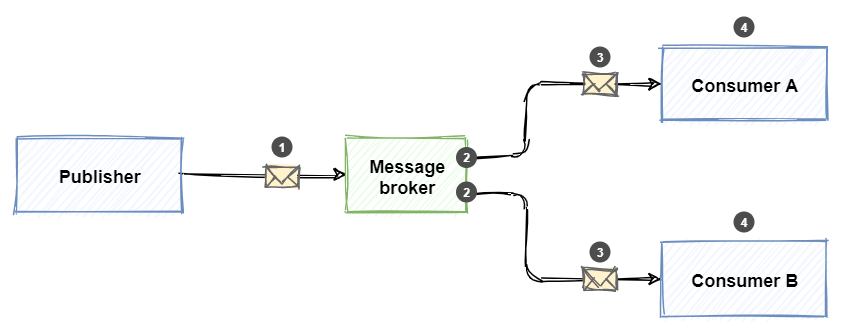
การแก้ไข คือ ติดตั้งใหม่โดยใช้คำสั่ง **yarn add @microsoft/signalr --ignore-engines**

**- Back-end** ใช้ .net core สร้างโปรเจ็คเป็น webapi 🡪 dotnet new webapi

ติดตั้ง package นี้ลงไปด้วย

https://www.nuget.org/packages/Microsoft.AspNet.SignalR/

## ภาพรวมการทำงาน



Client 1 (Front end 1)

Client 2 (Front end 2)

Client N (Front end N)

.net core API

เราจะทำการสร้าง C# .net api ให้เป็น Message broker สำหรับส่งต่อข้อมูลให้กับ front-end เมื่อมี action ใด ๆ เกิดขึ้น

- โดยตัว API จะต้องประกาศช่องทางในการเข้าใช้งาน Message broker (ประกาศ API URL ขึ้นมาเป็นช่องทางพิเศษ)

- Font-end ทำการเปิด connection มายัง API URL ที่ Back-end ประกาศขึ้น

## วิธีการตั้งค่า Back-end เป็น Message broker (.net core api)

ณ ตอนที่ทำเอกสารใช้ .net SDK version 5.0.102

1. ทำการสร้าง API โปรเจคด้วยคำสั่ง

dotnet new webapi

1. ติดตั้ง package signalR เข้าไปในโปรเจค โดยใช้คำสั่ง (การเลือกเวอร์ขึ้นอ้างอิงตามเว็บ nuget)

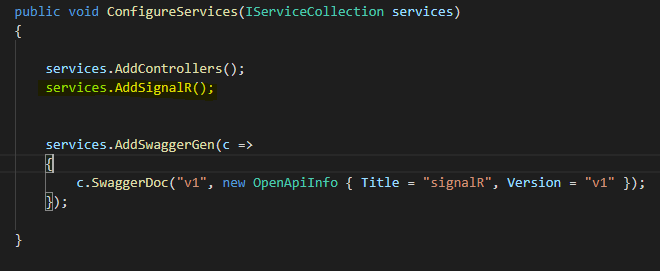
dotnet add package Microsoft.AspNet.SignalR --version 2.4.2

1. ทำการตั้งค่า cors policy ของ API ไว้ภายในไฟล์ startup.cs

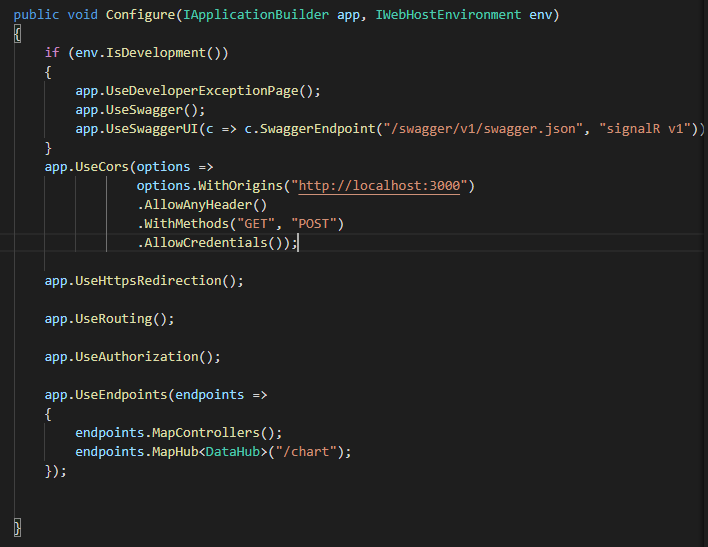
\*\* การตั้ง cors policy จะไม่อนุญาตให้ตั้งเป็น Allow any origin เพราะจะทำการเปิด

Allow credentials ด้วย

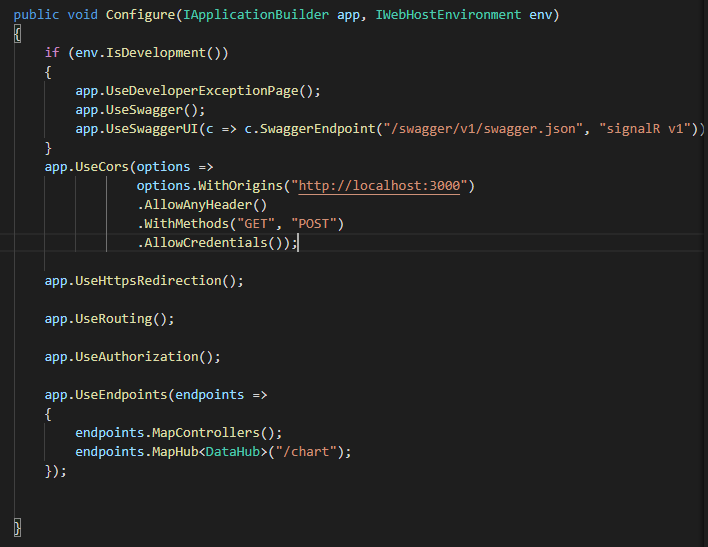
แก้ไขไฟล์ startup.cs ดังนี้



* 1. เรียกใช้งาน signalR ในฟังก์ชัน ConfigureServices

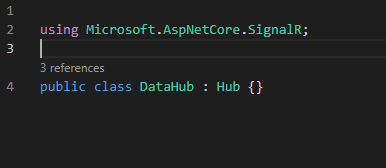


* 1. เปิดการอนุญาต cors policy เน้นการตั้งค่าที่ WithOrigins และ AllowCredentials



* 1. ประกาศช่องทางการเข้าใช้งาน Message broker

\*\*\* DataHub คือ class name ที่ทำการสร้างขึ้นมาอ้างอิงการสร้าง Message broker มีโครงสร้างดังนี้



จากขั้นตอนข้างต้น Client จะทำการเชื่อมต่อเข้ามายัง Message broker ได้ทาง http://<HOST\_NAME>/chart

\*\*\* เช่น <https://localhost:5001/chart>

Figure URL connection

1. สร้าง API controller ตามปกติ

HttpGET, HttpPOST ,,,.etc.

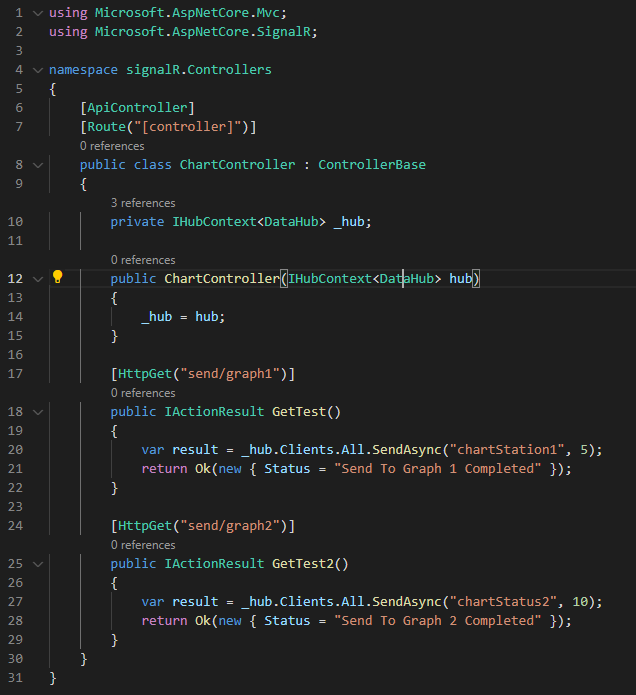


Figure API controller

เมื่อเรียก HTTP GET send/graph1 จะเข้ามาทำงานที่ฟังก์ชันด้านล่างตาม เงื่อนไขของโปรแกรมโดยทั่วไป

โค้ดโปรแกรมบรรทัดที่ 20 คือ

ให้ message broker ส่งค่าออกไปที่ channel ชื่อว่า **chartStation1** โดยส่ง 5 ออกไป

โค้ดโปรแกรมบรรทัดที่ 27 คือ

ให้ message broker ส่งค่าออกไปที่ channel ชื่อว่า **chartStation2** โดยส่ง 10 ออกไป

\*\*\*\* 2 คำสั่ง นี้จะต้องสอดคล้องกับ Front-end ซึ่งจะเปิด connection ไปตามชื่อ **channel** ที่ตั้ง

\*\*\* Front end ที่เชื่อต่อมายัง Channel เหล่านั้น จะได้รับข้อมูลกลับไปทันทีโดยที่ไม่ต้องรีเฟรชหน้า

## วิธีการให้หน้าบ้านเชื่อมต่อกับ message broker



1

2

3

1. Import package ที่ติดตั้งจากหน้าที่ 1 มาที่ Component
2. เปิดการเชื่อมต่อไปยัง Message broker ที่ตั้งค่าไว้ใน API (Figure 1 URL connection)
3. เปิดการติดตาม ไปยัง channel ชื่อว่า chartStation1 (Figure 2 API controller) ผลลัพธ์ที่ API ตอบกลับมาจะอยู่ในตัวแปลชื่อ data

โค้ดตัวอย่างอยู่ที่นี่

https://github.com/suwannason/signalR-example.git