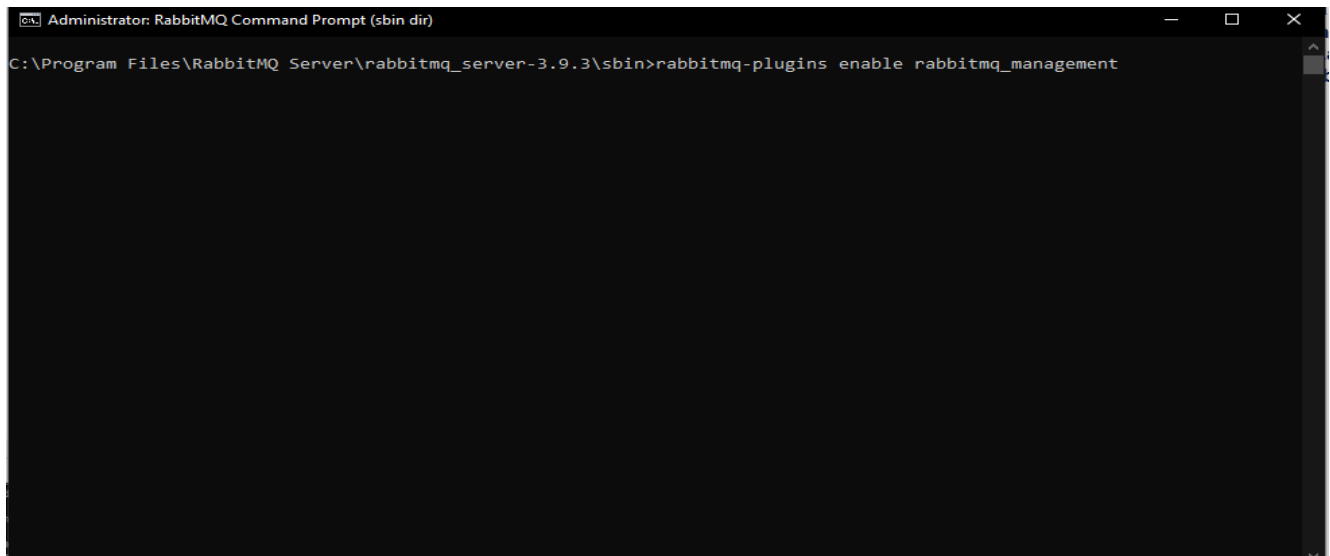
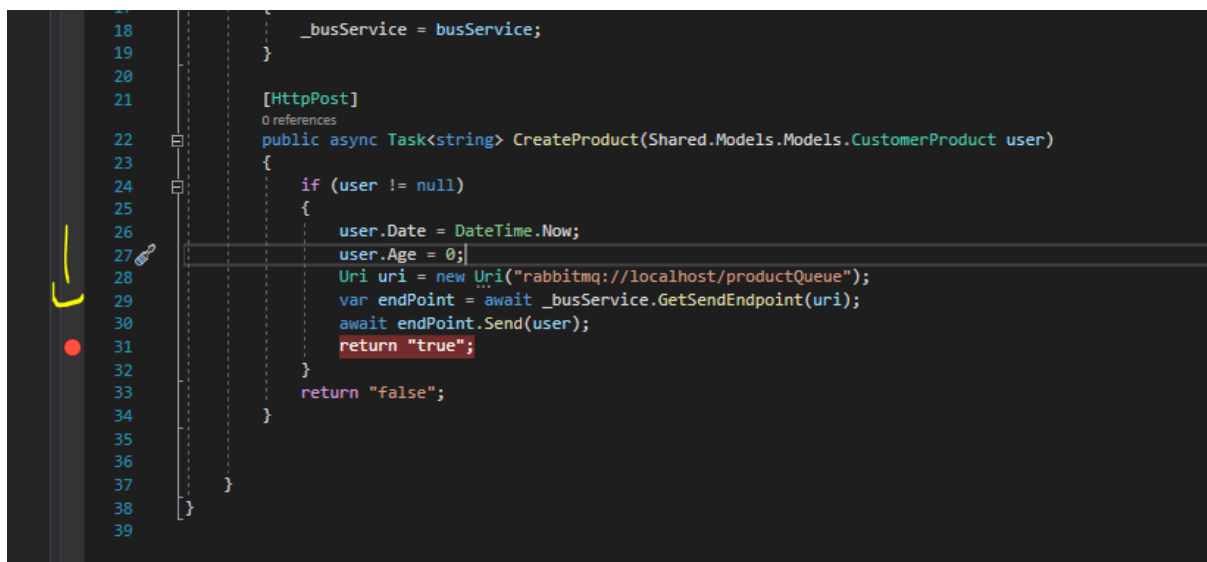


השלבים שצריך לבצע על מנת לראות את המידע עובר בין Microservices שונים :

- לבצע התקנה של Erlang כי היא שפת תכנות ו-RabbitMQ Server בנוי עליה.
- לבצע התקנה של RabbitMQ.
- לאחר התקנת שרת RabbitMQ, עלינו להפעיל את פורטל הניהול. יש לפתוח את שורת הפקודה עם זכויות מנהל (RabbitMQ Command Prompt) ולכתוב את הפקודה הבאה: `rabbitmq-plugins enable rabbitmq_management`

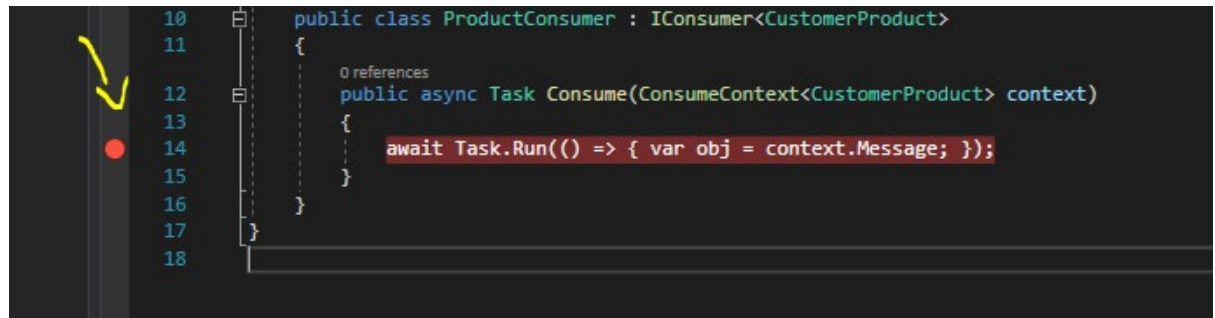


- כעת, עבור לכתובת זו (<http://localhost:15672>), ואז תראה שפורטל הניהול פועל.
משתמש : guest
סיסמא : guest
- כעת ניתן לפתוח את הפרויקט הנמצא בתיקיית `Microservices.WebApi`.
- בפרויקט נלחץ קליק ימני על Solution <- Properties <- Multiple startup project
-> נעביר לפעולה של Start על `Product.Microservices` ועל `Consomer.Microservices` <- Apply <- ok.
- תחת `Product.Microservices` <- Controllers <- בתוך `CustomerProductController.cs` נגדיר BreakPoint בדיוק לפי התמונה הבאה:



```
18     _busService = busService;
19 }
20
21 [HttpPost]
22 0 references
23 public async Task<string> CreateProduct(Shared.Models.Models.CustomerProduct user)
24 {
25     if (user != null)
26     {
27         user.Date = DateTime.Now;
27         user.Age = 0;
28         Uri uri = new Uri("rabbitmq://localhost/productQueue");
29         var endPoint = await _busService.GetSendEndpoint(uri);
30         await endPoint.Send(user);
31         return "true";
32     }
33     return "false";
34 }
35
36
37
38
39 }
```

- תחת ProductConsumer.cs נגדיר BreakPoing בדיוק לפי התמונה הבא:

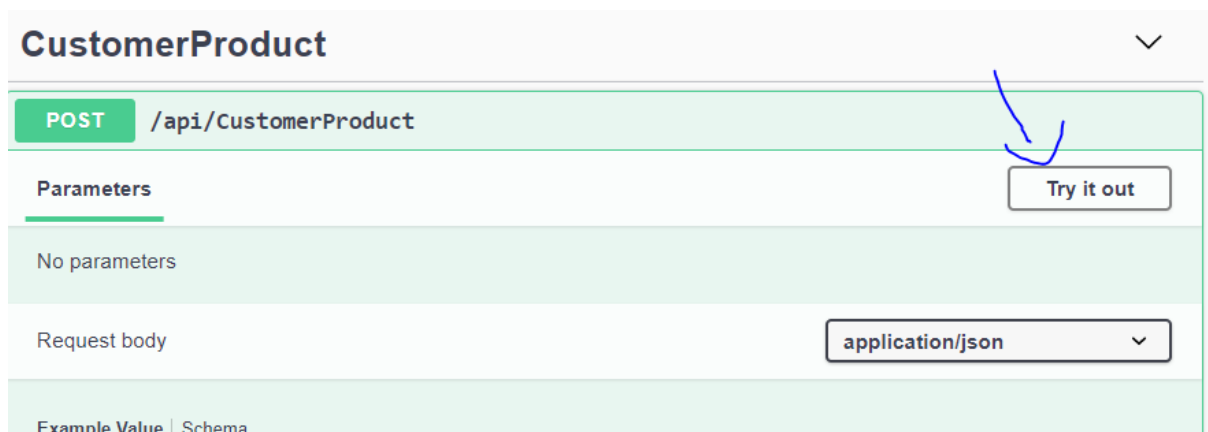
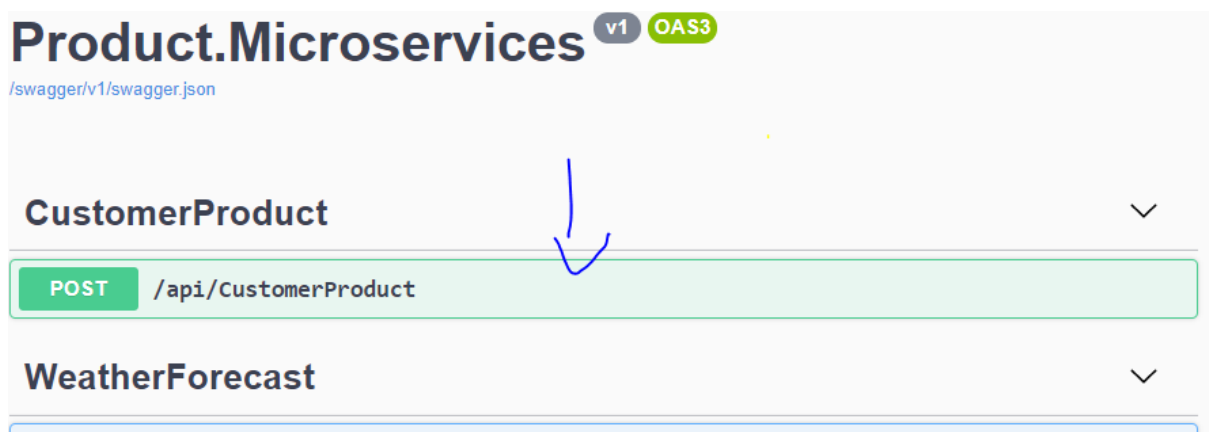


```

10 public class ProductConsumer : IConsumer<CustomerProduct>
11 {
12     0 references
13     public async Task Consume(ConsumeContext<CustomerProduct> context)
14     {
15         await Task.Run(() => { var obj = context.Message; });
16     }
17 }
18

```

- כעת ננלחץ על Play ב Visual studio ויפתחו לנו שתי חלונות אחד של Product.Microservices והשני של Consomer.Microservices.
 - כאשר ניכנס לפורטל הניהול של RabbitMq - <http://localhost:15672> , נוכל לראות שיש לנו שתי Connections.
- בחלון של Product.Microservices נבצע את הפעולות לפי התמונות הבאות:



CustomerProduct

POST

/api/CustomerProduct

Parameters

Cancel

Reset

No parameters

Request body

application/json

```
{  "firstName": "Natia",  "date": "2021-08-18T17:07:42.751Z",  "age": 30,  "profession": "IT Specialist"}}
```

Execute

Responses

RabbitMQ™

RabbitMQ 3.9.3Erlang 24.0.5

Overview

Connections

Channels

Exchanges

Queues

Admin

Queues

All queues (1)

Pagination

Page 1 of 1 - Filter: ☐ Regex ?

Overview				Messages			Message rates			+/-
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
productQueue	classic	D	idle	1	0	1	0.00/s	0.00/s	0.00/s	

Add a new queue

HTTP API

Server Docs

Tutorials

Community Support

Community Slack

Commercial Support

Plugins

GitHub

Changelog

- ▶ Consumers
- ▶ Bindings
- ▶ Publish message

▼ Get messages

Warning: getting messages from a queue is a destructive action. ?

Ack Mode:

Encoding: ?

Messages:

Get Message(s)

- ▶ Move messages
- ▶ Delete
- ▶ Purge
- ▶ Runtime Metrics (Advanced)



RabbitMQ 3.9.3

Erlang 24.0.5

Overview

Connections

Channels

Exchanges

Queues

Admin

```
],
  "message": {
    "firstName": "Natia",
    "date": "2021-08-18T20:10:57.0539534+03:00",
    "profession": "IT Spesialist"
  },
  "sentTime": "2021-08-18T17:10:57.0547557Z",
  "headers": {
    "MT-Activity-Id": "00-6975f9e8eeacfb499aaee0d368e4cc30-f6174c4e6242b947-00"
  },
  "host": {
```

Overview

Connections

Channels

Exchanges

Queues

Admin

▶ Publish message

▼ Get messages

Warning: getting message

Queue is empty

Close

Ack Mode: Nack message requeue true ▼

Encoding: Auto string / base64 ▼ ?

Messages: 1

Get Message(s)

Message 1

The server reported 0 messages remaining.

Exchange | productQueue

Routing
Key

Redelivered