# ASP.NET Core 6 with OpenTelemetry

Hi there!

Today we are going to create an ASP.NET Core 6 WebAPI and we are going to integrate it with OpenTelemetry. First, we need to start Jaeger, Zipkin and ELK (Elasticsearch, Logstash and Kibana) Stack. They collect the logs and traces from our API and store them to present them in GUI format for the convenience of developers. They trace every request, their source, etc.

Let us start our docker and type:

## docker run -d -p 6831:6831/udp -p 6832:6832/udp -p 14250:14250 -p 14268:14268 -p 16686:16686 -p 5778:5778 --name jaeger jaegertracing/all-in-one

Now, we need to run Zipkin. Type the command:

## docker run -d -p 9411:9411 --name zipkin openzipkin/zipkin

```
C:\Windows\system32\cmd.exe

C:\Users\2048766>docker run -d -p 9411:9411 --name zipkin openzipkin/zipkin
Unable to find image 'openzipkin/zipkin:latest' locally
latest: Pulling from openzipkin/zipkin
b49a1cd62aab: Pull complete
ed98ae2aec4b: Pull complete
2fda6b53cfec: Pull complete
afcff3f8f4f7: Pull complete
aa81df1cac50: Pull complete
08ecf0b4a53e: Pull complete
dd27e27737a4: Pull complete
d305b82a4bd0: Pull complete
a4192e291340: Pull complete
Digest: sha256:9a7dbc81516a15348a250225fb79cd60d0d9938f1a5373b2a0118ba8c828ffdf
Status: Downloaded newer image for openzipkin/zipkin:latest
ce7b11932c5d92491635e52c6492e4b2f3a6534770094f7a9ed74642e49535e8

C:\Users\2048766>
```

Now, we need to compose this file:

```yaml
version: '3.6'

services:

  Elasticsearch:
    image: elasticsearch:7.16.2
    container_name: elasticsearch
    restart: always
    volumes:
      - elastic_data:/usr/share/elasticsearch/data/
    environment:
      ES_JAVA_OPTS: "-Xmx256m -Xms256m"
      discovery.type: single-node
    ports:
      - "9200:9200"
      - "9300:9300"
    networks:
      - elk


  Logstash:
    image: logstash:7.16.2
    container_name: logstash
    restart: always
    volumes:
```

```yaml
  Logstash:
    image: logstash:7.16.2
    container_name: logstash
    restart: always
    volumes:
      - ./logstash/:/logstash_dir
    depends_on:
      - Elasticsearch
    environment:
      LS_JAVA_OPTS: "-Xmx256m -Xms256m"
    ports:
      - "9600:9600"
    networks:
      - elk


  Kibana:
    image: kibana:7.16.2
    container_name: kibana
    restart: always
    ports:
      - "5601:5601"
    environment:
      - ELASTICSEARCH_URL=http://elasticsearch:9200
    depends_on:
      - Elasticsearch
    networks:
      - elk
```

```yaml
volumes:
  elastic_data: {}

networks:
  elk:
```

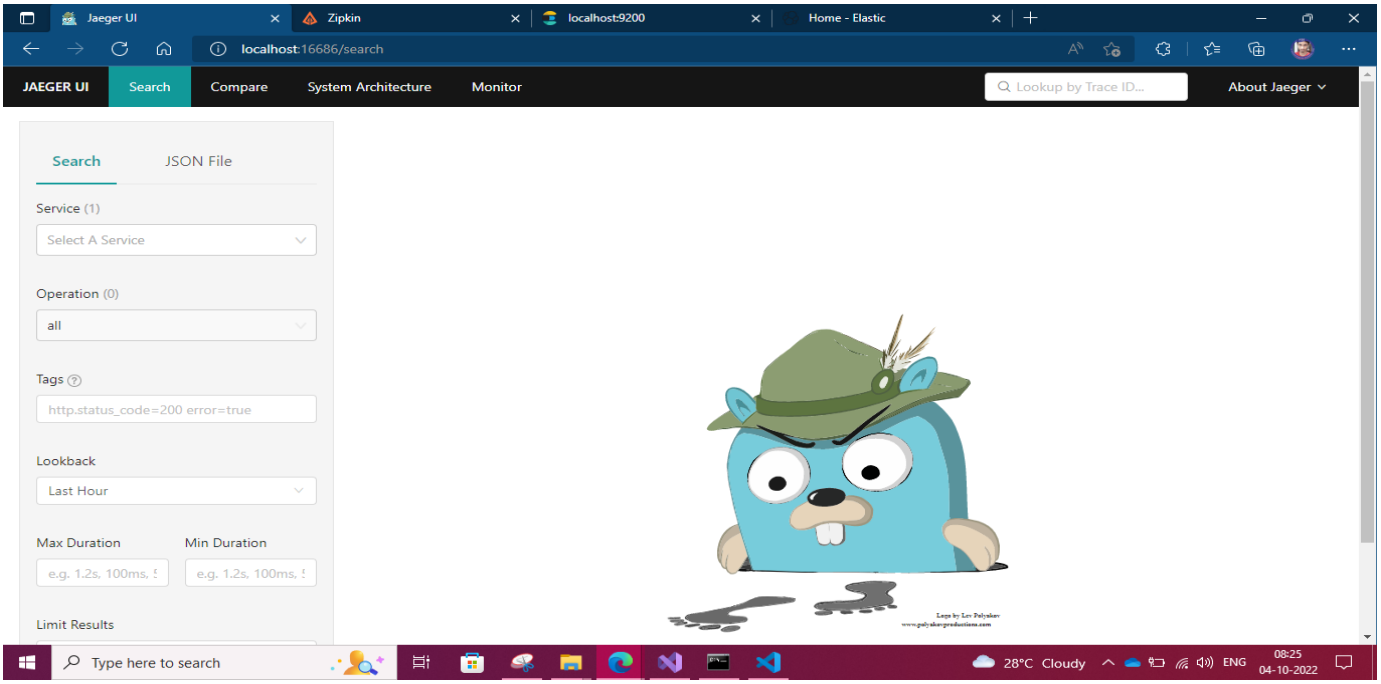
```
C:\Windows\system32\cmd.exe - docker-compose up -d

C:\Users\2048766>cd "C:\Users\2048766\source\repos\Net6Otel\ELK Docker"

C:\Users\2048766\source\repos\Net6Otel\ELK Docker>dir
 Volume in drive C has no label.
 Volume Serial Number is A243-3629

 Directory of C:\Users\2048766\source\repos\Net6Otel\ELK Docker

04-10-2022  08:15    <DIR>          .
04-10-2022  08:15    <DIR>          ..
04-10-2022  08:13             1,013 docker-compose.yml
               1 File(s)          1,013 bytes
               2 Dir(s)  358,163,148,800 bytes free

C:\Users\2048766\source\repos\Net6Otel\ELK Docker>docker-compose up -d
```

```
C:\Windows\system32\cmd.exe
 - Elasticsearch Pulled                                                    132.4s
   - 7b1a6ab2e44d Pull complete                                             78.6s
   - c0f9908af642 Pull complete                                             83.4s
   - 04928834f6a2 Pull complete                                             84.0s
   - 6f647d55d420 Pull complete                                            125.3s
   - da7afec4a2d7 Pull complete                                            125.7s
   - 146b4a7018b9 Pull complete                                            125.8s
   - dc193d8d7c71 Pull complete                                            126.0s
   - e3deb19b2c38 Pull complete                                            126.1s
   - 8b79893ce5a3 Pull complete                                            126.4s
 - Logstash Pulled                                                         137.6s
   - 2d473b07cdd5 Pull complete                                             67.4s
   - 86c3ca619800 Pull complete                                             87.9s
   - 6f0d86179cf0 Pull complete                                             88.6s
   - dab45c17c796 Pull complete                                            130.6s
   - 31f07a7dad20 Pull complete                                            130.8s
   - 688f479e228d Pull complete                                            130.9s
   - b54982bc6eed Pull complete                                            131.0s
   - 490e23f550b1 Pull complete                                            131.1s
   - e6ce1a615e06 Pull complete                                            131.2s
   - ecb11b460dfc Pull complete                                            131.4s
   - 165ca4b8efa5 Pull complete                                            131.6s
[+] Running 5/5
 - Network elkdocker_elk            Created                                  0.1s
 - Volume "elkdocker_elastic_data"  Created                                  0.0s
 - Container elasticsearch          Started                                 26.5s
 - Container kibana                 Started                                 26.9s
 - Container logstash               Started                                 27.2s

C:\Users\2048766\source\repos\Net6Otel\ELK Docker>
```

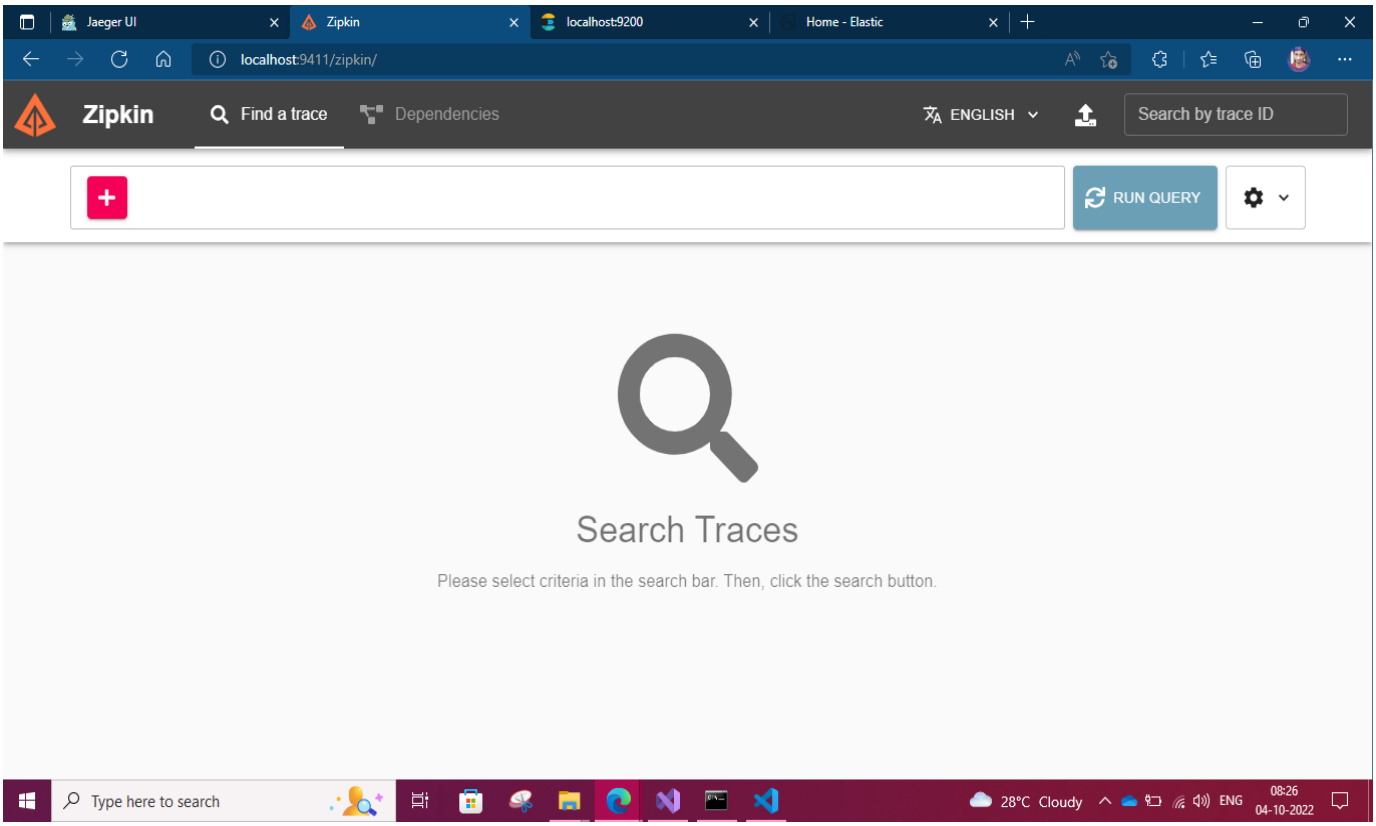
Now, check the URLs:

- http://localhost:16686/
- http://localhost:9411/
- http://localhost:9200/
- http://localhost:5601/

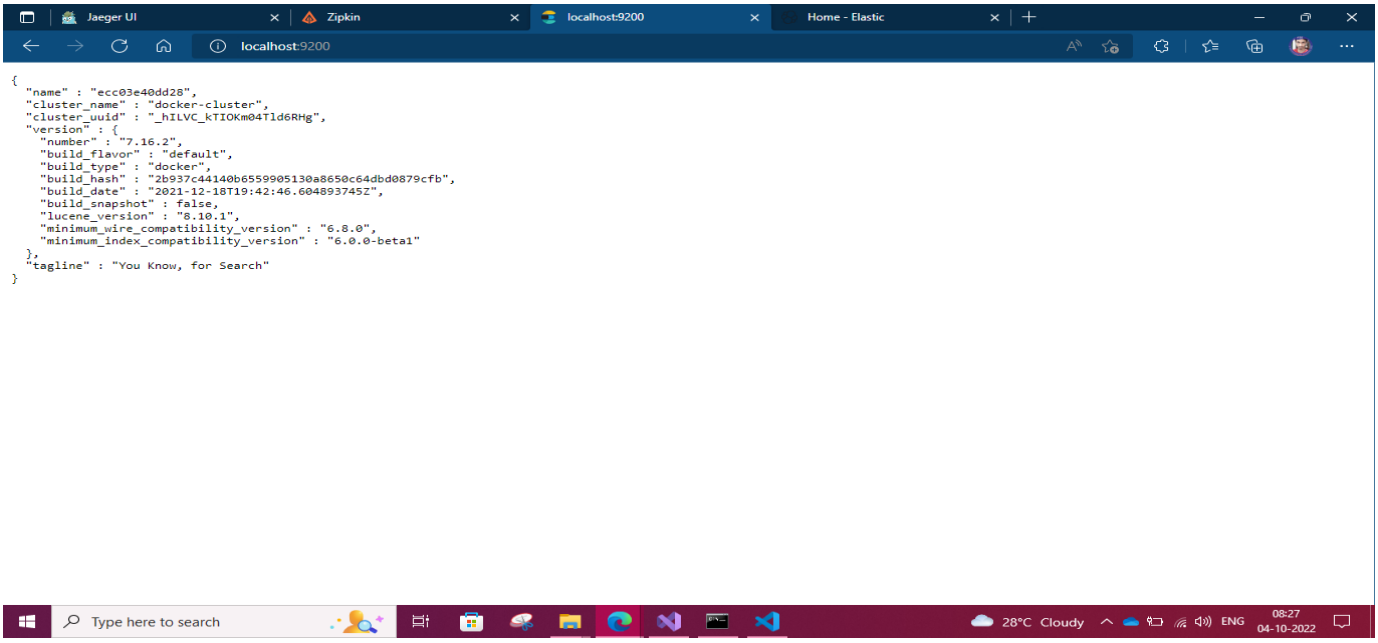
## Jaeger



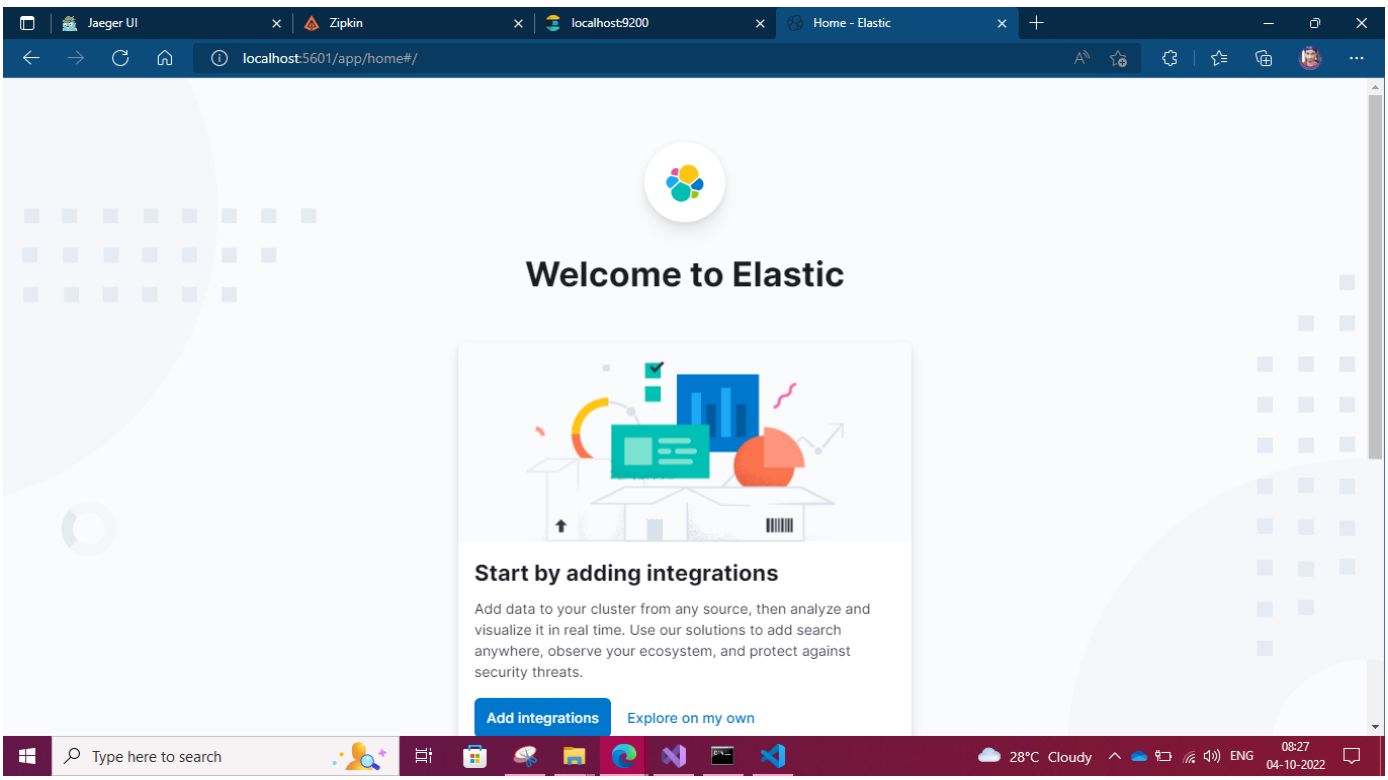

## Zipkin

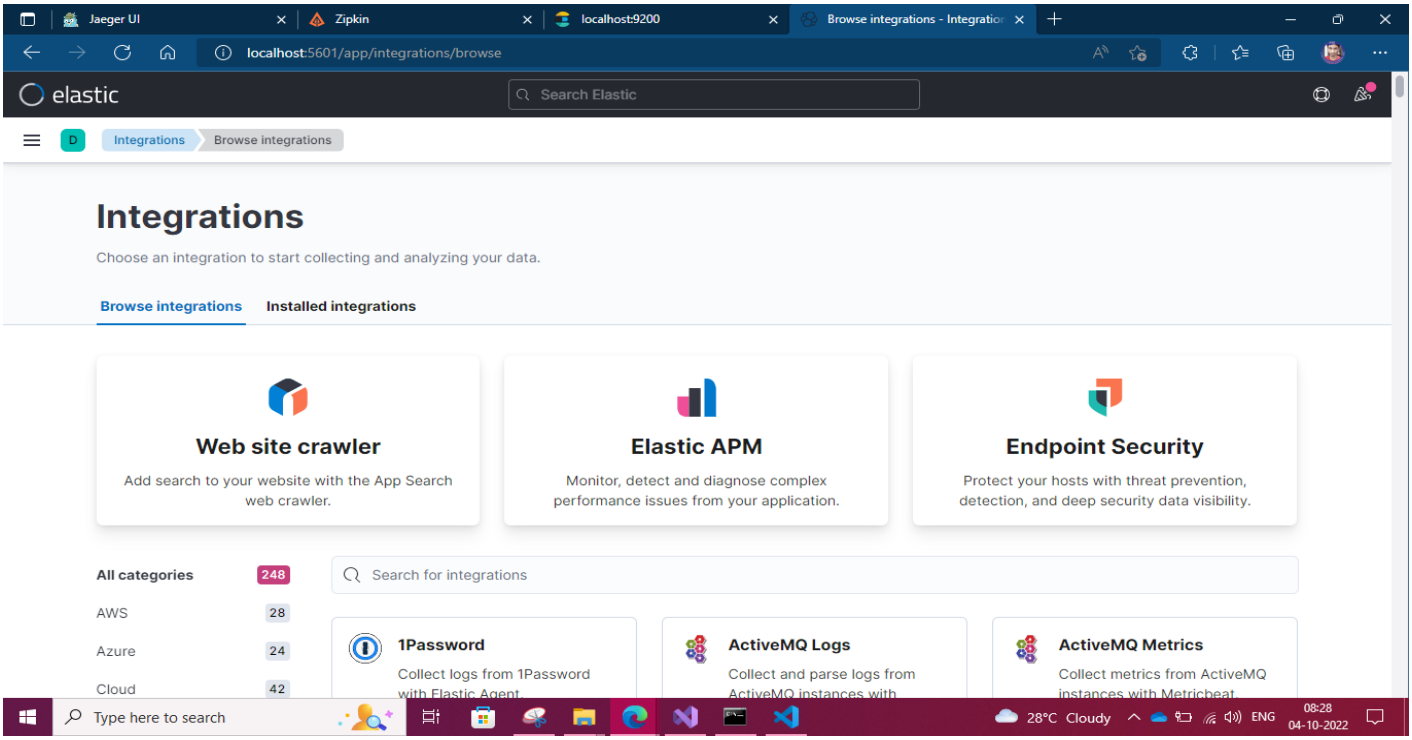



## Elasticsearch

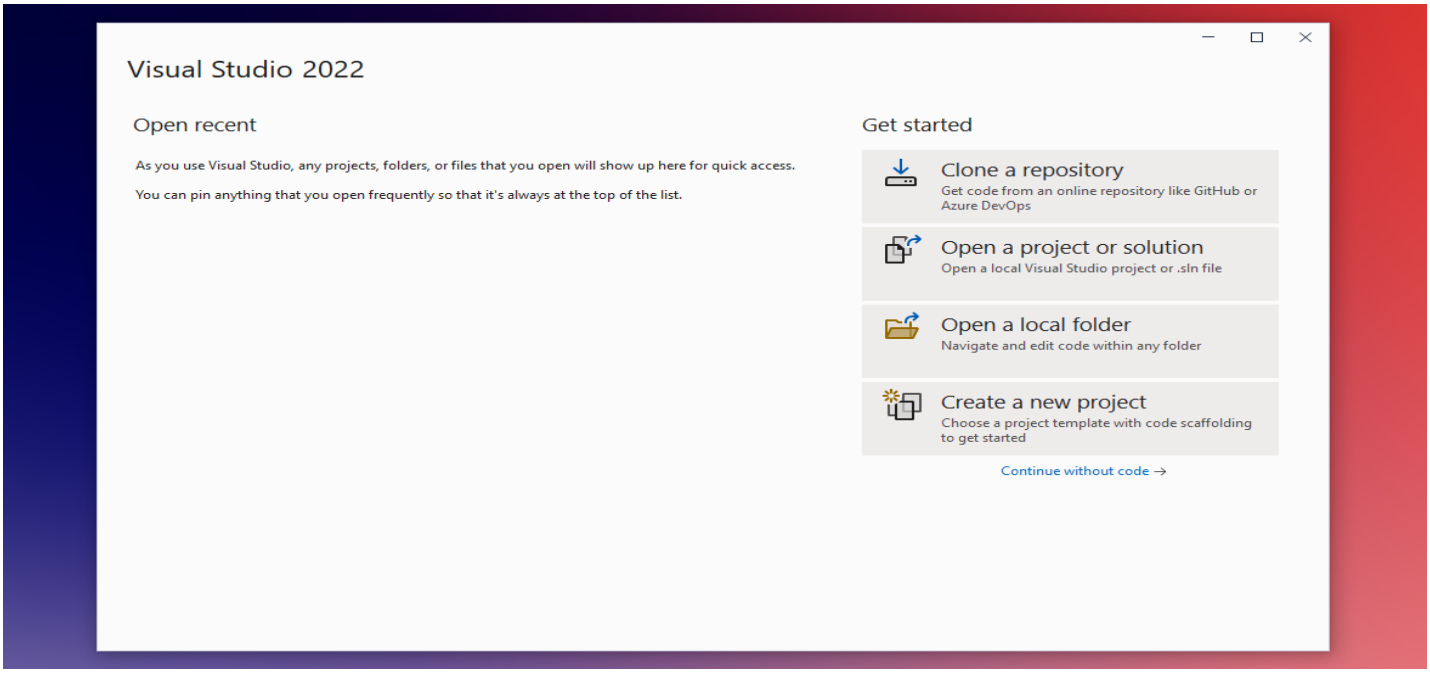

Jaeger UI
Zipkin
localhost:9200
Home - Elastic
localhost:9200
{
"name" : "ecc03e40dd28",
"cluster_name" : "docker-cluster",
"cluster_uuid" : "_hILVC_kTIOKm04Tld6RHg",
"version" : {
"number" : "7.16.2",
"build_flavor" : "default",
"build_type" : "docker",
"build_hash" : "2b937c44140b6559905130a8650c64dbd0879cfb",
"build_date" : "2021-12-18T19:42:46.604893745Z",
"build_snapshot" : false,
"lucene_version" : "8.10.1",
"minimum_wire_compatibility_version" : "6.8.0",
"minimum_index_compatibility_version" : "6.0.0-beta1"
},
"tagline" : "You Know, for Search"
}
Type here to search
28°C Cloudy
ENG
08:27
04-10-2022

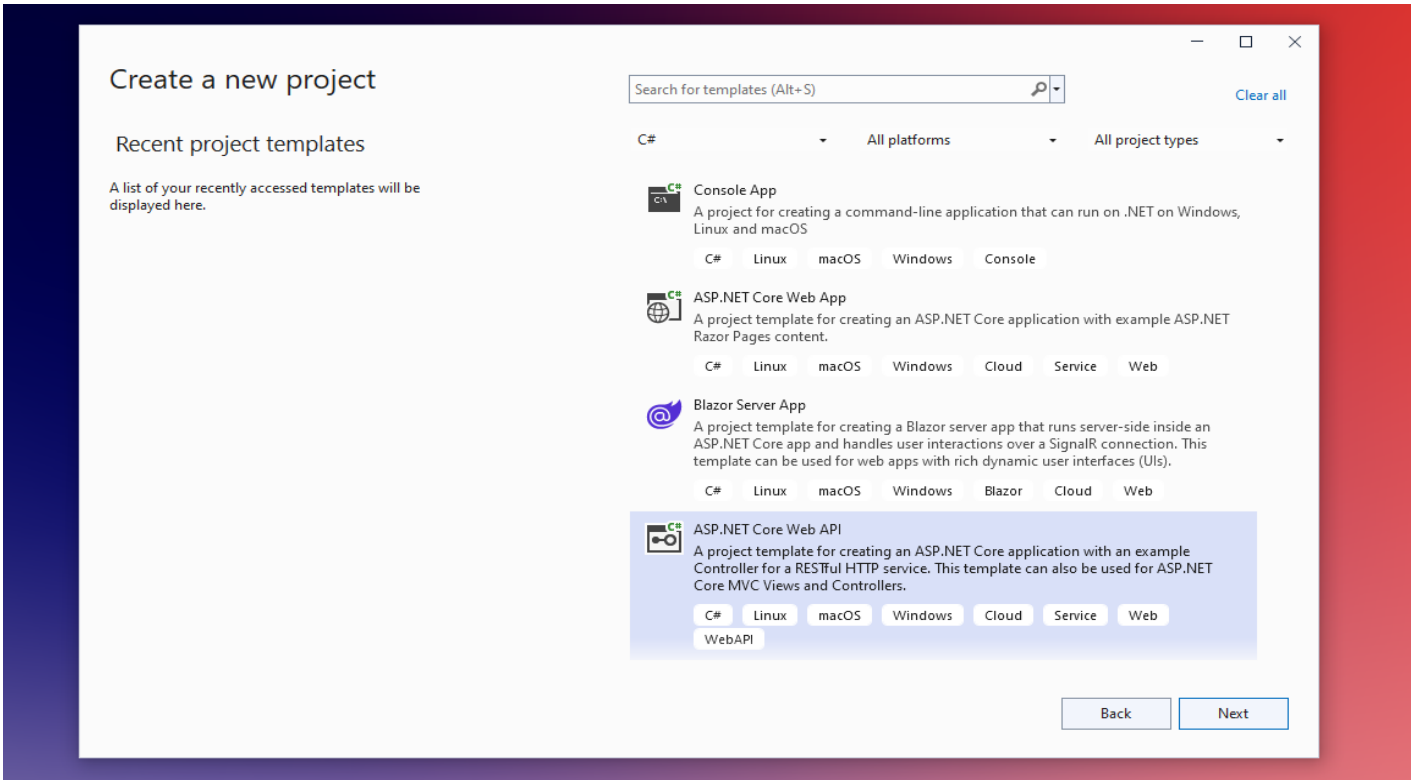
## Kibana

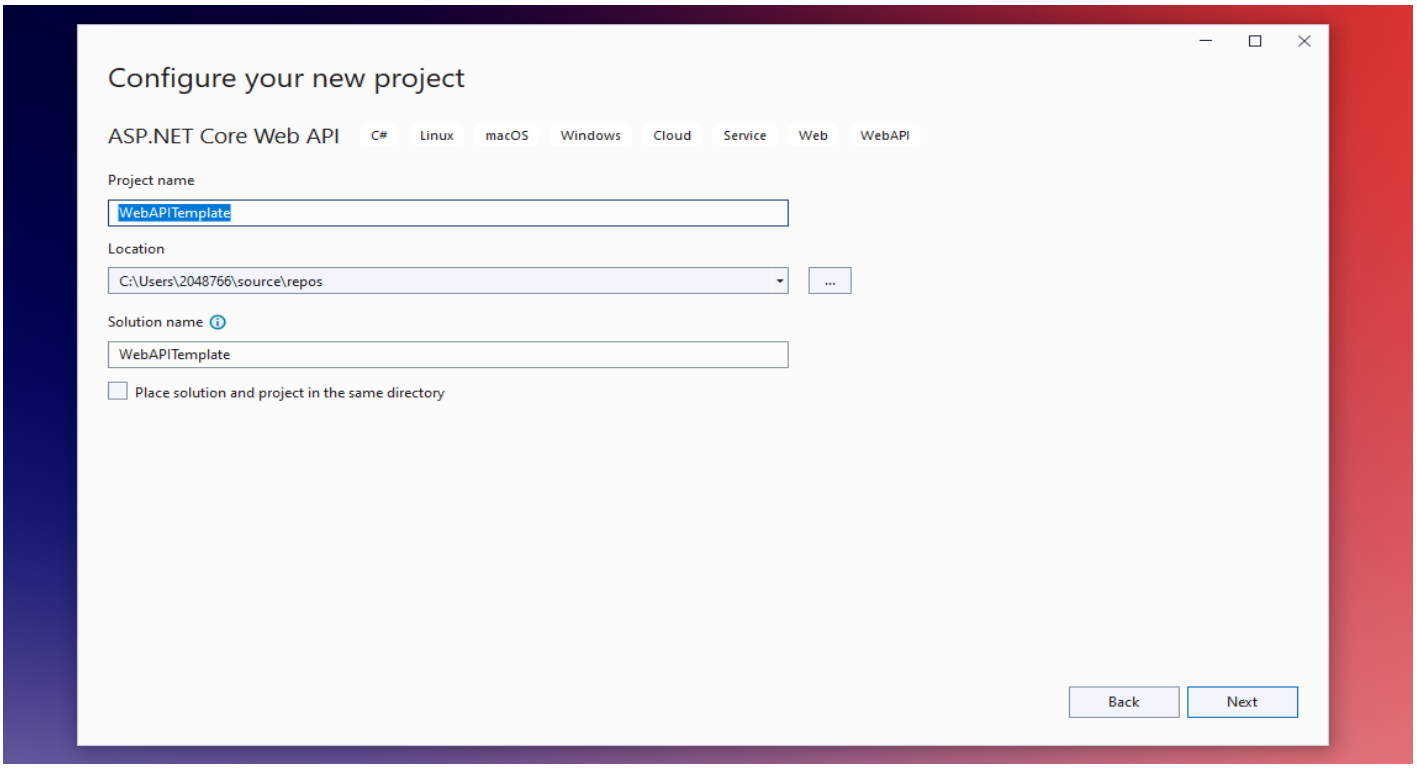

Jaeger UI
Zipkin
localhost:9200
Home - Elastic
localhost:5601/app/home#/
Welcome to Elastic
Start by adding integrations
Add data to your cluster from any source, then analyze and visualize it in real time. Use our solutions to add search anywhere, observe your ecosystem, and protect against security threats.
Add integrations
Explore on my own
Type here to search
28°C Cloudy
ENG
08:27
04-10-2022

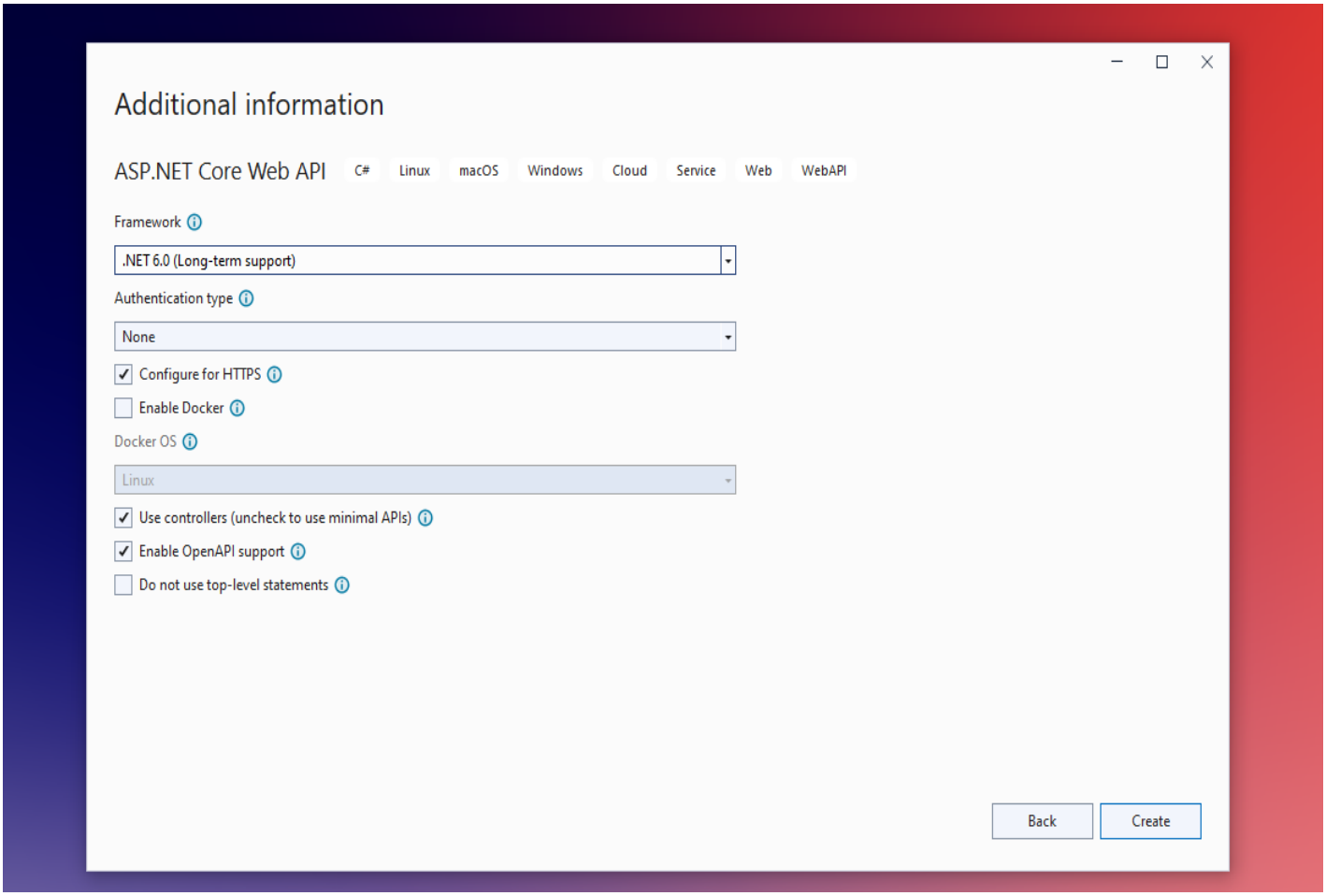


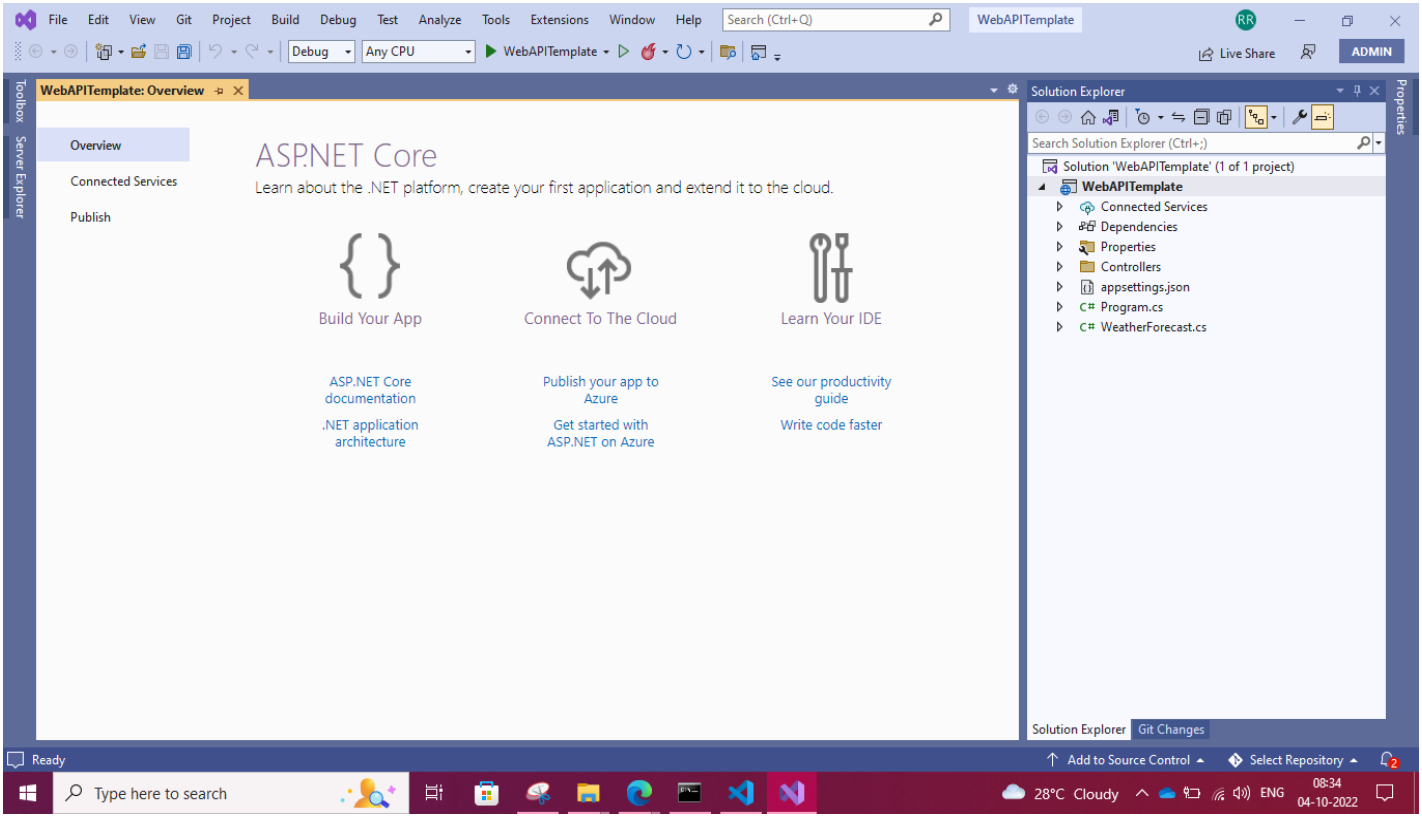

Now, let us create a webapi.



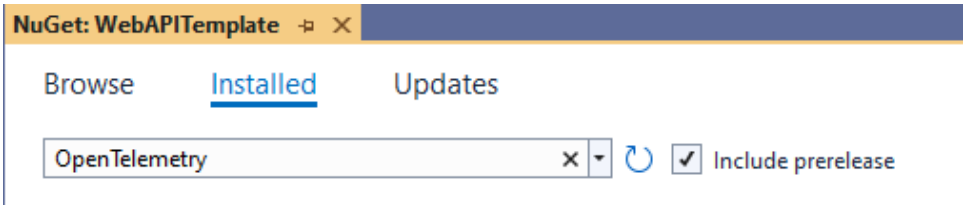

Create a new project
Search for templates (Alt+S)
Clear all
Recent project templates
A list of your recently accessed templates will be displayed here.
C#
All platforms
All project types
Console App
A project for creating a command-line application that can run on .NET on Windows, Linux and macOS
C# Linux macOS Windows Console
ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.
C# Linux macOS Windows Cloud Service Web
Blazor Server App
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Blazor Cloud Web
ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.
C# Linux macOS Windows Cloud Service Web WebAPI
Back
Next


Configure your new project
ASP.NET Core Web API
C# Linux macOS Windows Cloud Service Web WebAPI
Project name
WebAPITemplate
Location
C:\Users\2048766\source\repos
...
Solution name
WebAPITemplate
Place solution and project in the same directory
Back
Next

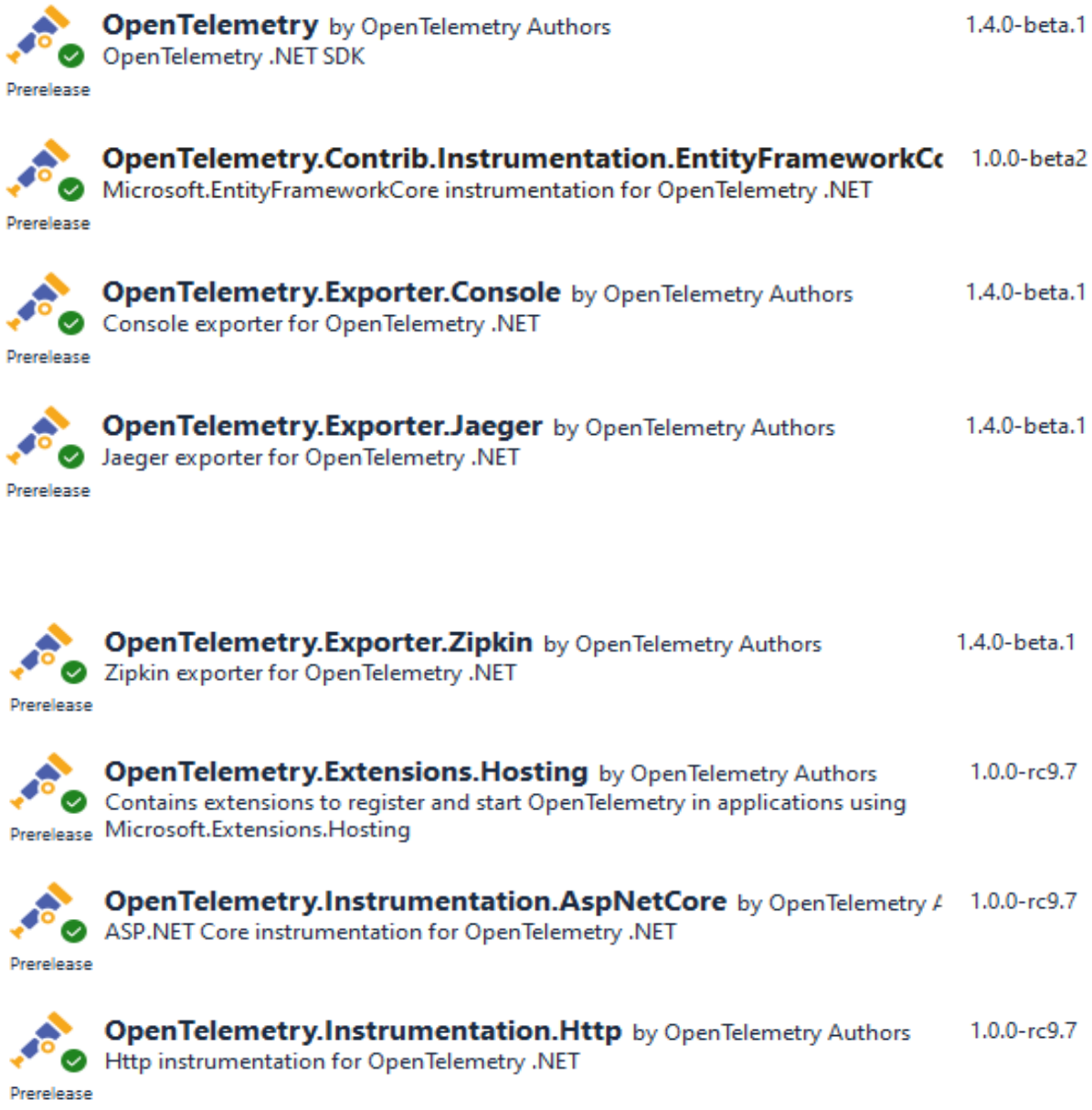

Additional information
ASP.NET Core Web API
C#
Linux
macOS
Windows
Cloud
Service
Web
WebAPI
Framework
.NET 6.0 (Long-term support)
Authentication type
None
Configure for HTTPS
Enable Docker
Docker OS
Linux
Use controllers (uncheck to use minimal APIs)
Enable OpenAPI support
Do not use top-level statements
Back
Create


File
Edit
View
Git
Project
Build
Debug
Test
Analyze
Tools
Extensions
Window
Help
Search (Ctrl+Q)
WebAPITemplate
Debug
Any CPU
WebAPITemplate
Live Share
ADMIN
WebAPITemplate: Overview
Overview
Connected Services
Publish
ASP.NET Core
Learn about the .NET platform, create your first application and extend it to the cloud.
Build Your App
Connect To The Cloud
Learn Your IDE
ASP.NET Core documentation
.NET application architecture
Publish your app to Azure
Get started with ASP.NET on Azure
See our productivity guide
Write code faster
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'WebAPITemplate' (1 of 1 project)
WebAPITemplate
Connected Services
Dependencies
Properties
Controllers
appsettings.json
Program.cs
WeatherForecast.cs
Solution Explorer
Git Changes
Ready
Add to Source Control
Select Repository
Type here to search
28°C Cloudy
ENG
08:34
04-10-2022

Now, we need to install some packages. Open NuGet Package Manager. Right-click the project and select Manage NuGet Package. Remember to check the Include prerelease check box as most of the OpenTelemetry were created in 2019 and are in prerelease phase.




Packages to be installed:




Now build the project. Right-click on the project and click Build.

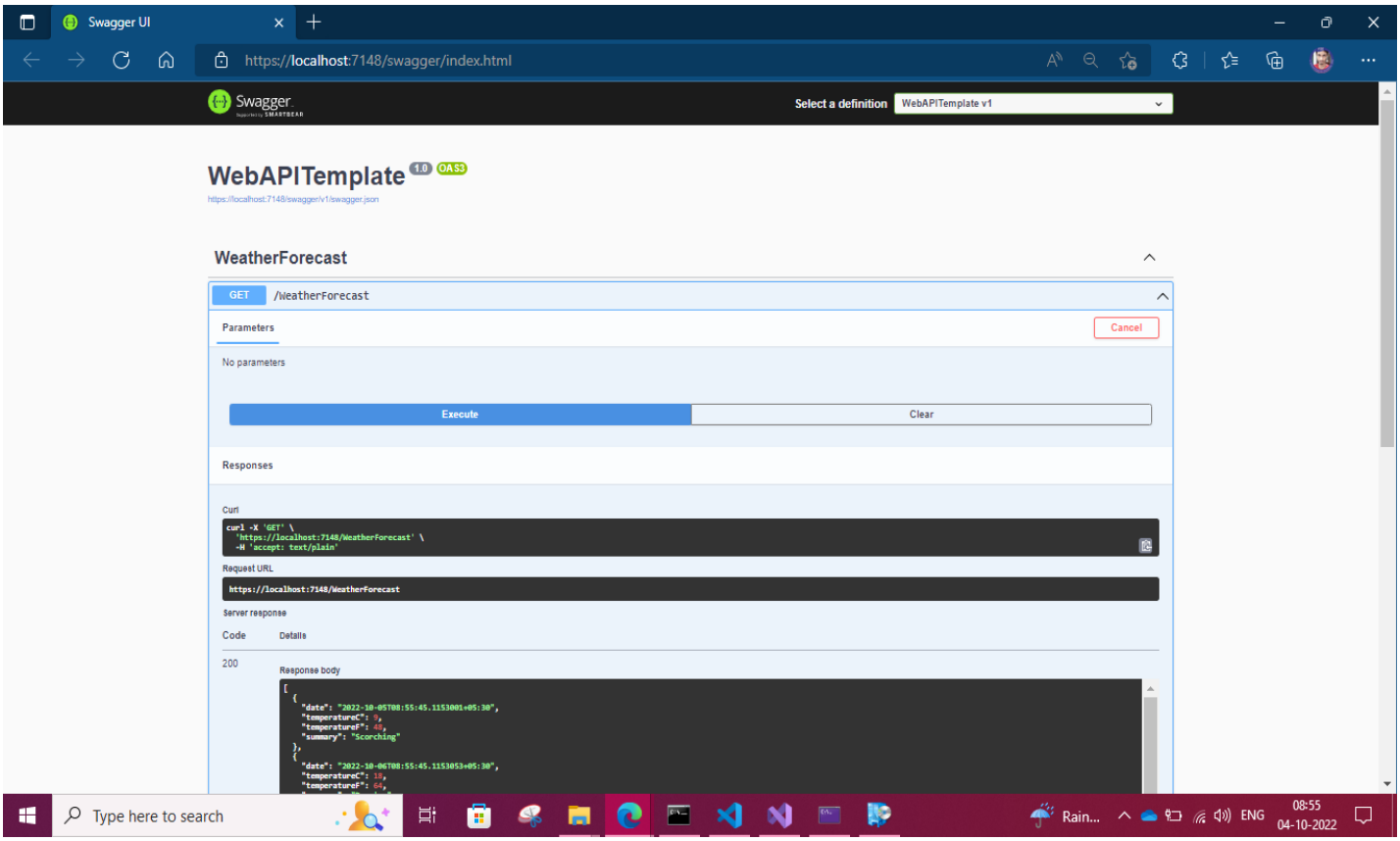
```
Output
Show output from: Build
Build started...
1>------ Build started: Project: WebAPITemplate, Configuration: Debug Any CPU ------
1>WebAPITemplate -> C:\Users\2048766\source\repos\WebAPITemplate\WebAPITemplate\bin\Debug\net6.0\WebAPITemplate.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

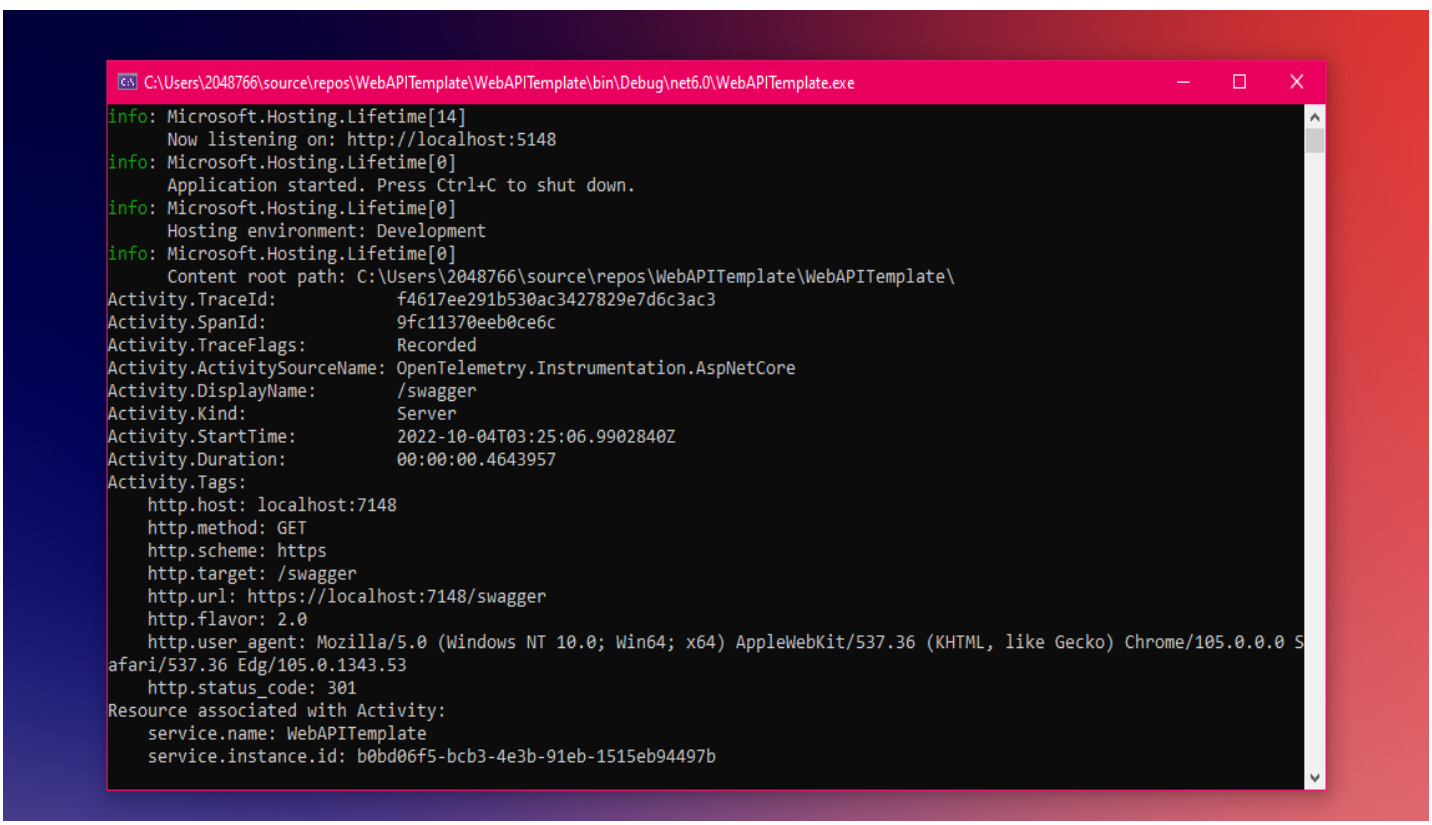
Add the following codes to Program.cs:

```
Program.cs*
WebAPITemplate
1  using OpenTelemetry.Trace;
2  using OpenTelemetry.Resources;
3
4  var builder = WebApplication.CreateBuilder(args);
5
6  // Add OpenTelemetry
7  builder.Services.AddOpenTelemetryTracing(x =>
8  {
9      x.SetResourceBuilder(ResourceBuilder.CreateDefault()
10         .AddService("WebAPITemplate"));
11
12     x.AddSource("TraceSource");
13
14     x.AddAspNetCoreInstrumentation(sam => sam.Filter = httpContext =>
15         !httpContext.Request.Path.Value?
16         .Contains("/_framework/aspnetcore-browser-refresh.js") ?? true);
17
18     x.AddHttpClientInstrumentation(sam => sam.Enrich =
19     (activity, eventName, rawObject) =>
20     {
21         if (eventName == "OnStartActivity"
22         && rawObject is HttpRequestMessage request
23         && request.Method == HttpMethod.Get)
24             activity.SetTag("WebAPITemplate", "This is a microservice.");
25     });
```

```
26
27     x.AddEntityFrameworkCoreInstrumentation(sam =>
28     {
29         sam.SetDbStatementForStoredProcedure = true;
30         sam.SetDbStatementForText = true;
31     });
32
33     x.AddConsoleExporter();
34
35     x.AddJaegerExporter(sam =>
36     {
37         sam.AgentHost = "localhost";
38         sam.AgentPort = 6831;
39     });
40
41     x.AddZipkinExporter(sam =>
42     {
43         sam.Endpoint = new Uri($"http://localhost:9411/api/v2/spans");
44     });
45 });
46
```

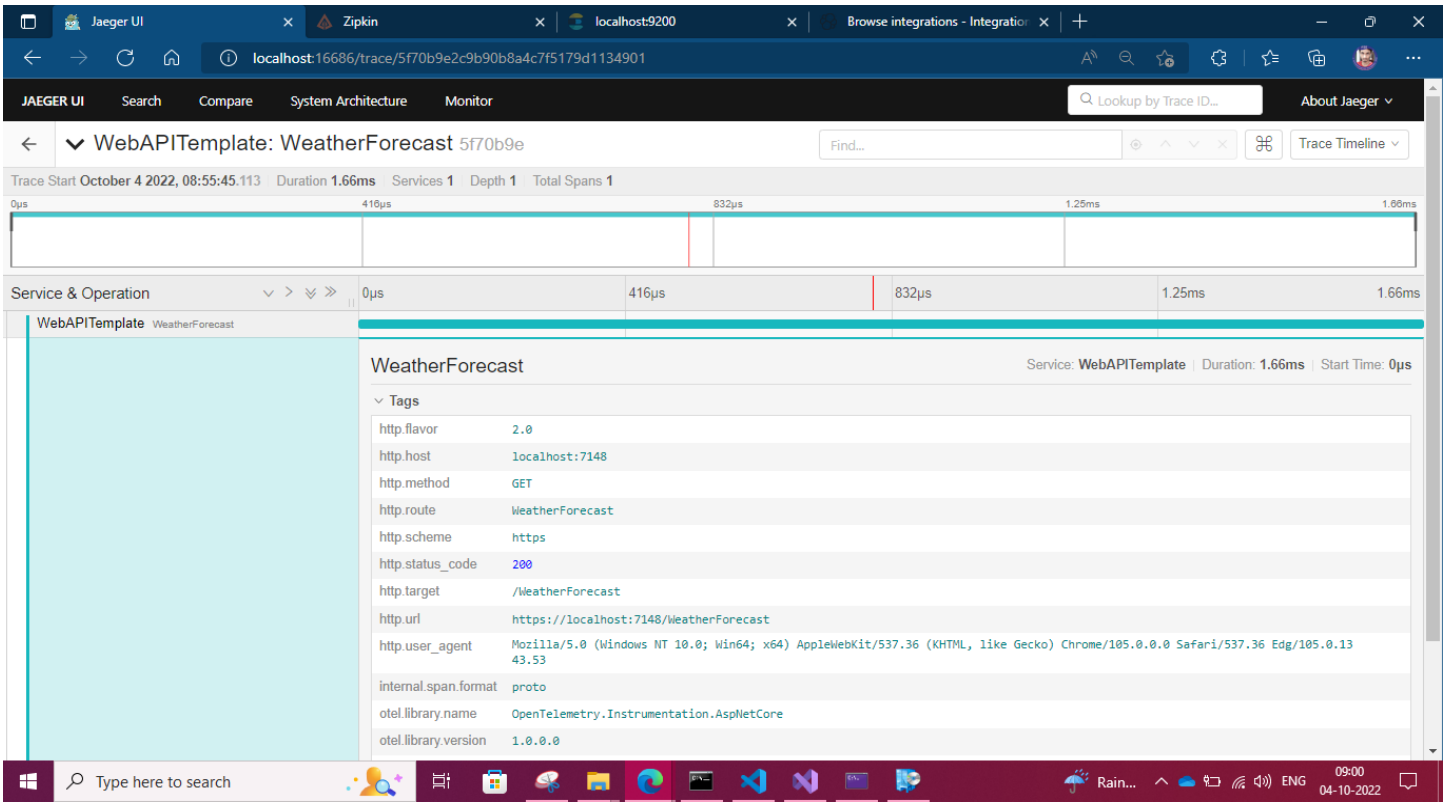
Run the API template.



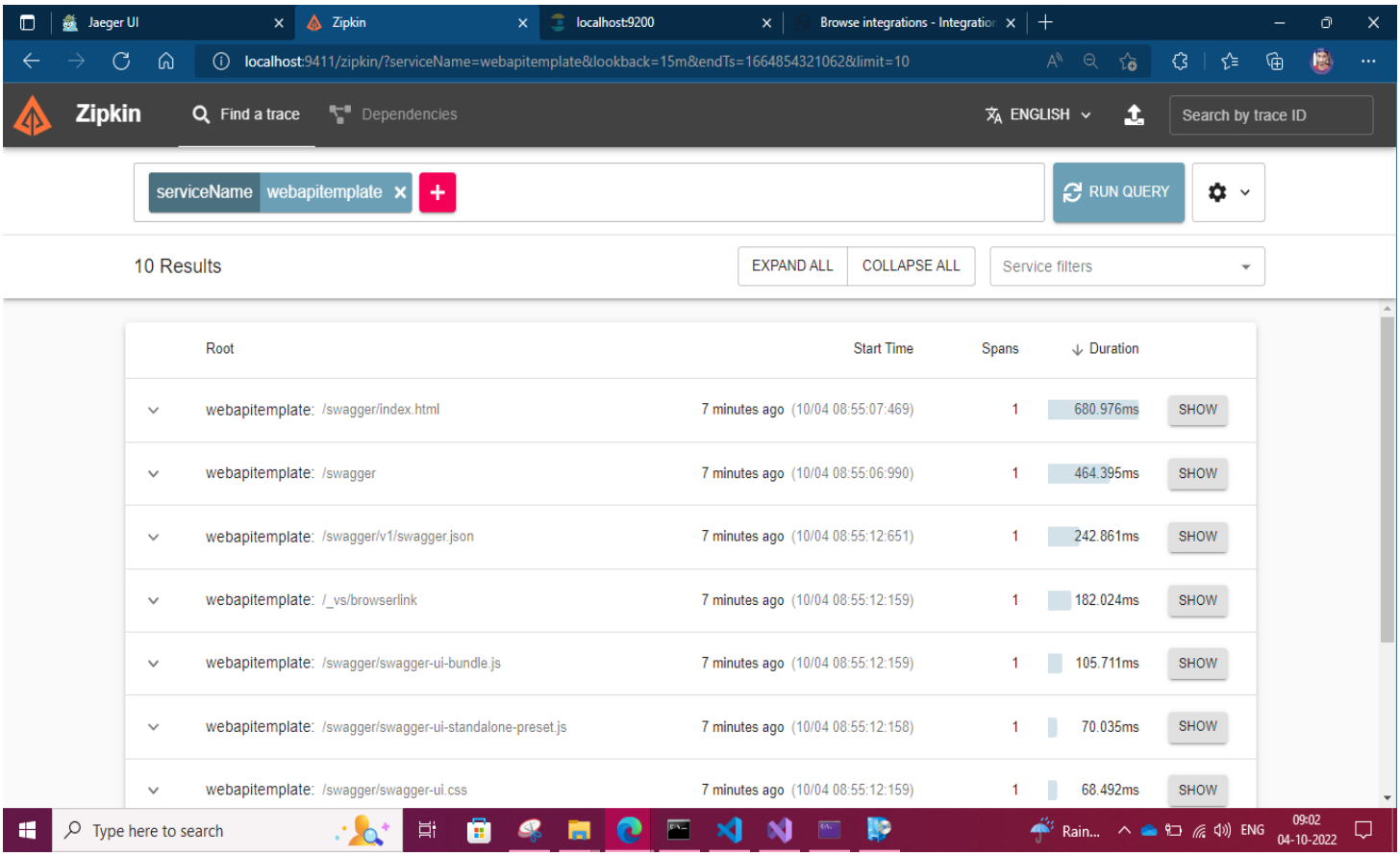




Now, refresh the Jaeger page and select the WebAPITemplate service. Click on Find Trace.

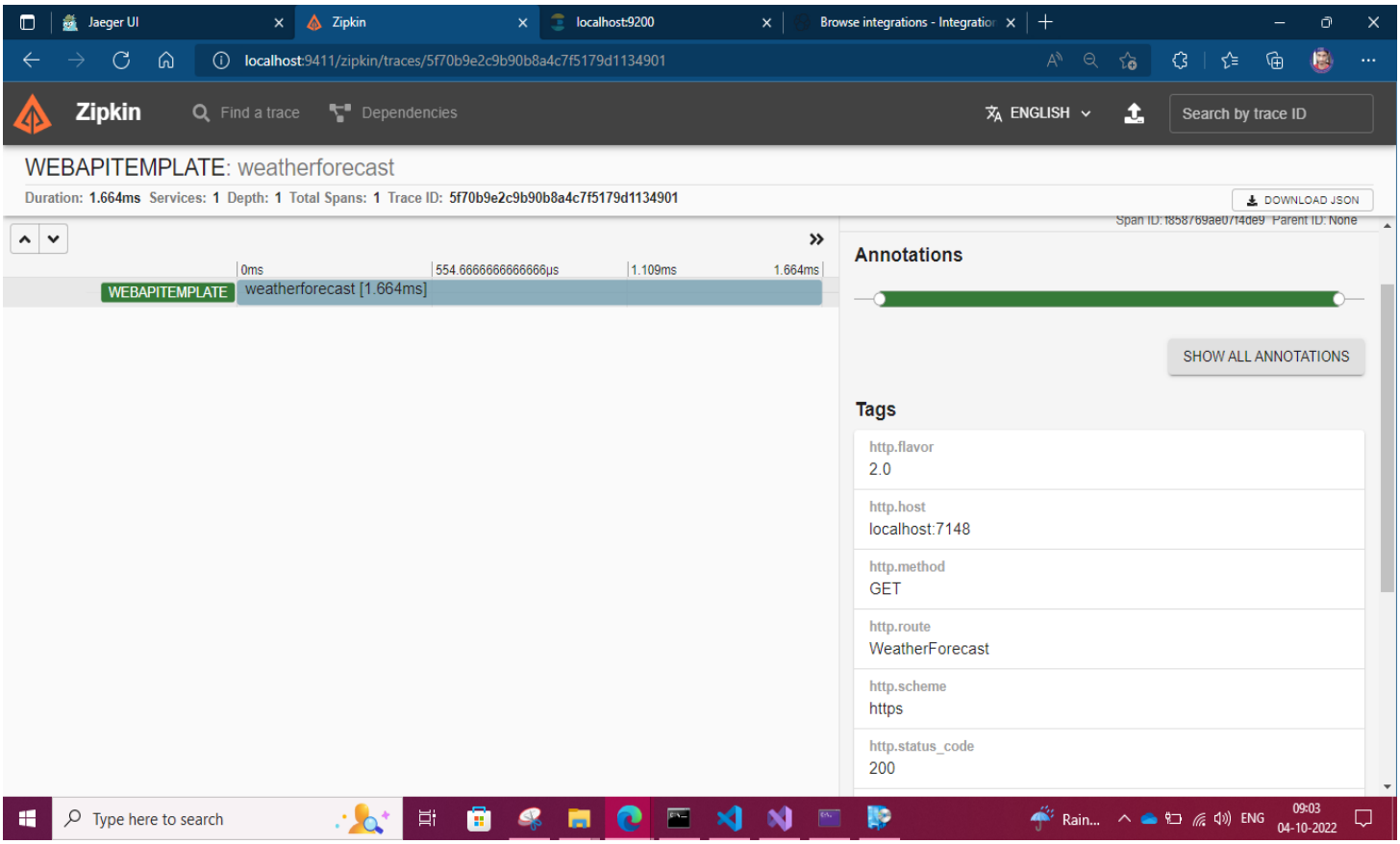
Select a trace and check the results:




Now refresh the Zipkin page and select a service name and click Run Query.



Select a span and see the trace.




Now, we will try to export logs and traces to ELK Stack. Add the following packages to the API:

**prometheus-net.AspNetCore** by andrasm,qed-,lakario,s 7.0.0-pre-000288-4688bd3
ASP.NET Core middleware and stand-alone Kestrel server for exporting metrics to Prometheus
Prerelease

**Serilog.AspNetCore** by Microsoft,Serilog Contributors 6.1.0-dev-00285
Serilog support for ASP.NET Core logging
Prerelease

**Serilog.Enrichers.Environment** by Serilog Contributors 2.2.1-dev-00787
Enrich Serilog log events with properties from System.Environment.
Prerelease

**Serilog.Exceptions** by Muhammad Rehan Saeed (RehanSaeed.com) 8.4.0
Log exception details and custom properties that are not output in Exception.ToString().

**Serilog.Exceptions** by Muhammad Rehan Saeed (RehanSaeed.com) 8.4.0
Log exception details and custom properties that are not output in Exception.ToString ().

**Serilog.Sinks.Debug** by Serilog Contributors 2.0.0
A Serilog sink that writes log events to the debug output window.

Serilog will send data to the ELK Stack and Prometheus is used for adding metrics to the API. Right-click the project and build the project again.

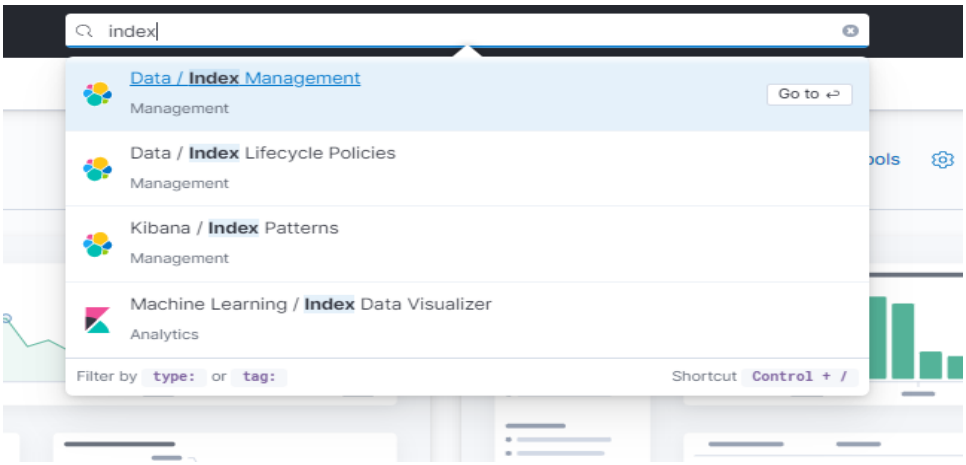
```
Output
Show output from: Build
Build started...
1>------ Build started: Project: WebAPITemplate, Configuration: Debug Any CPU ------
1>WebAPITemplate -> C:\Users\2048766\source\repos\WebAPITemplate\WebAPITemplate\bin\Debug\net6.0\WebAPITemplate.dll
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

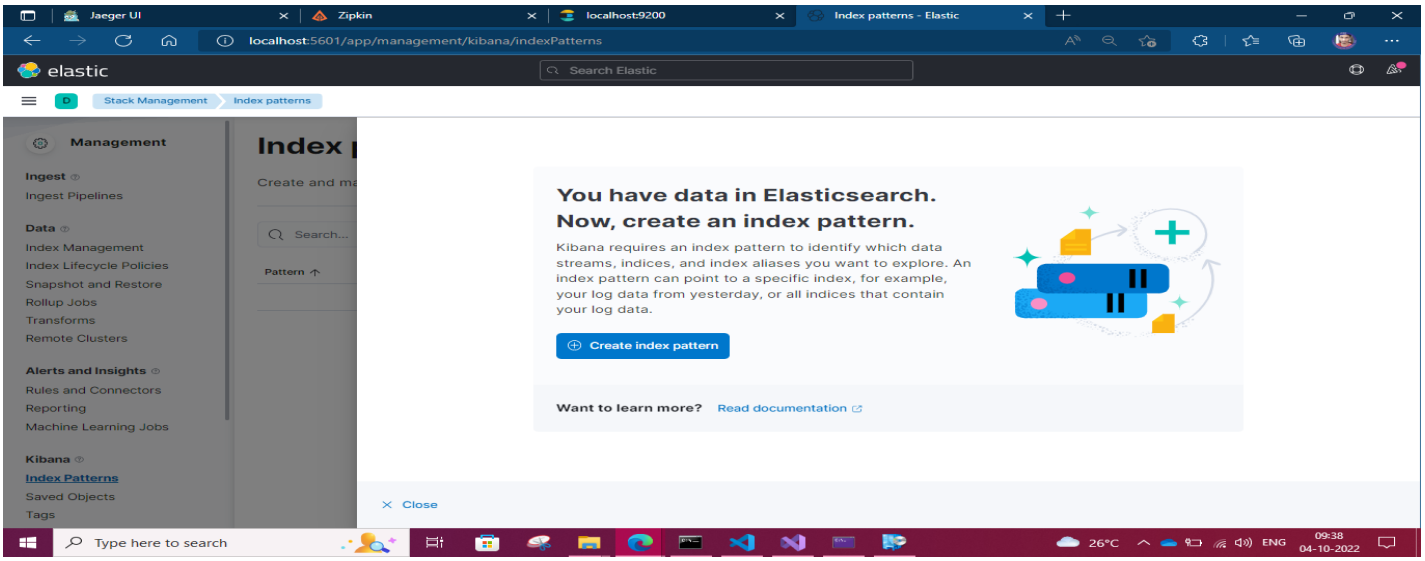
Add the following codes to Program.cs

```
using OpenTelemetry.Trace;
using OpenTelemetry.Resources;
using Serilog;
using Serilog.Exceptions;
using Serilog.Sinks.Elasticsearch;
using System.Reflection;

var builder = WebApplication.CreateBuilder(args);

// Adding OpenTelemetry using ELK
ConfigureLogs();

void ConfigureLogs()
{
    var environment = Environment.GetEnvironmentVariable("ASPNETCORE_ENVIRONMENT");

    var configuration = new ConfigurationBuilder()
        .AddJsonFile("appsettings.json", optional: false, reloadOnChange: true)
        .Build();

    Log.Logger = new LoggerConfiguration()
        .Enrich.FromLogContext()
        .Enrich.WithExceptionDetails()
        .WriteTo.Debug()
        .WriteTo.Console()
        .WriteTo.Elasticsearch(ConfigureELS(configuration, environment))
        .CreateLogger();
}
```

```
ElasticsearchSinkOptions ConfigureELS(IConfigurationRoot configuration, string? environment)
{
    return new ElasticsearchSinkOptions(new Uri($"http://localhost:9200"))
    {
        AutoRegisterTemplate = true,
        IndexFormat = $"{Assembly.GetExecutingAssembly().GetName().Name?.ToLower()}-" +
        $"{environment?.ToLower().Replace(".","-")}-{DateTime.UtcNow:yyyy-MM}"
    };
}
```

Run the API template and refresh the Kibana.




Search index patterns. Select Kibana\Index Patterns. Click Create index pattern.

Jaeger UI | Zipkin | localhost:9200 | Index patterns - Elastic

localhost:5601/app/management/kibana/indexPatterns

elastic — Search Elastic

Stack Management > Index patterns

**Management**

- Ingest
  - Ingest Pipelines
- Data
  - Index Management
  - Index Lifecycle Policies
  - Snapshot and Restore
  - Rollup Jobs
  - Transforms
  - Remote Clusters
- Alerts and Insights
  - Rules and Connectors
  - Reporting
  - Machine Learning Jobs
- Kibana
  - Index Patterns
  - Saved Objects
  - Tags

## Create index pattern

Name

webapitemp*

Use an asterisk (*) to match multiple characters. Spaces and the characters , /, ?, ", <, >, | are not allowed.

Timestamp field

@timestamp

- @timestamp
- --- I don't want to use the time filter ---

Close | Create index pattern

Your index pattern matches 1 source.

webapitemplate-development-2022-10 — Index

Rows per page: 10

Type here to search | 26°C | ENG | 09:41 04-10-2022

Jaeger UI | Zipkin | localhost:9200 | webapitemp* - Elastic

localhost:5601/app/management/kibana/indexPatterns/patterns/ab092040-439a-11ed-b826-6330763f4a26#/?_a=(tab:inde...

elastic — Search Elastic

Stack Management > Index patterns > webapitemp*

**Management**

- Ingest
  - Ingest Pipelines
- Data
  - Index Management
  - Index Lifecycle Policies
  - Snapshot and Restore
  - Rollup Jobs
  - Transforms
  - Remote Clusters
- Alerts and Insights
  - Rules and Connectors
  - Reporting
  - Machine Learning Jobs
- Kibana
  - Index Patterns
  - Saved Objects
  - Tags

# webapitemp*

Time field: '@timestamp'

View and edit fields in **webapitemp***. Field attributes, such as type and searchability, are based on field mappings in Elasticsearch.

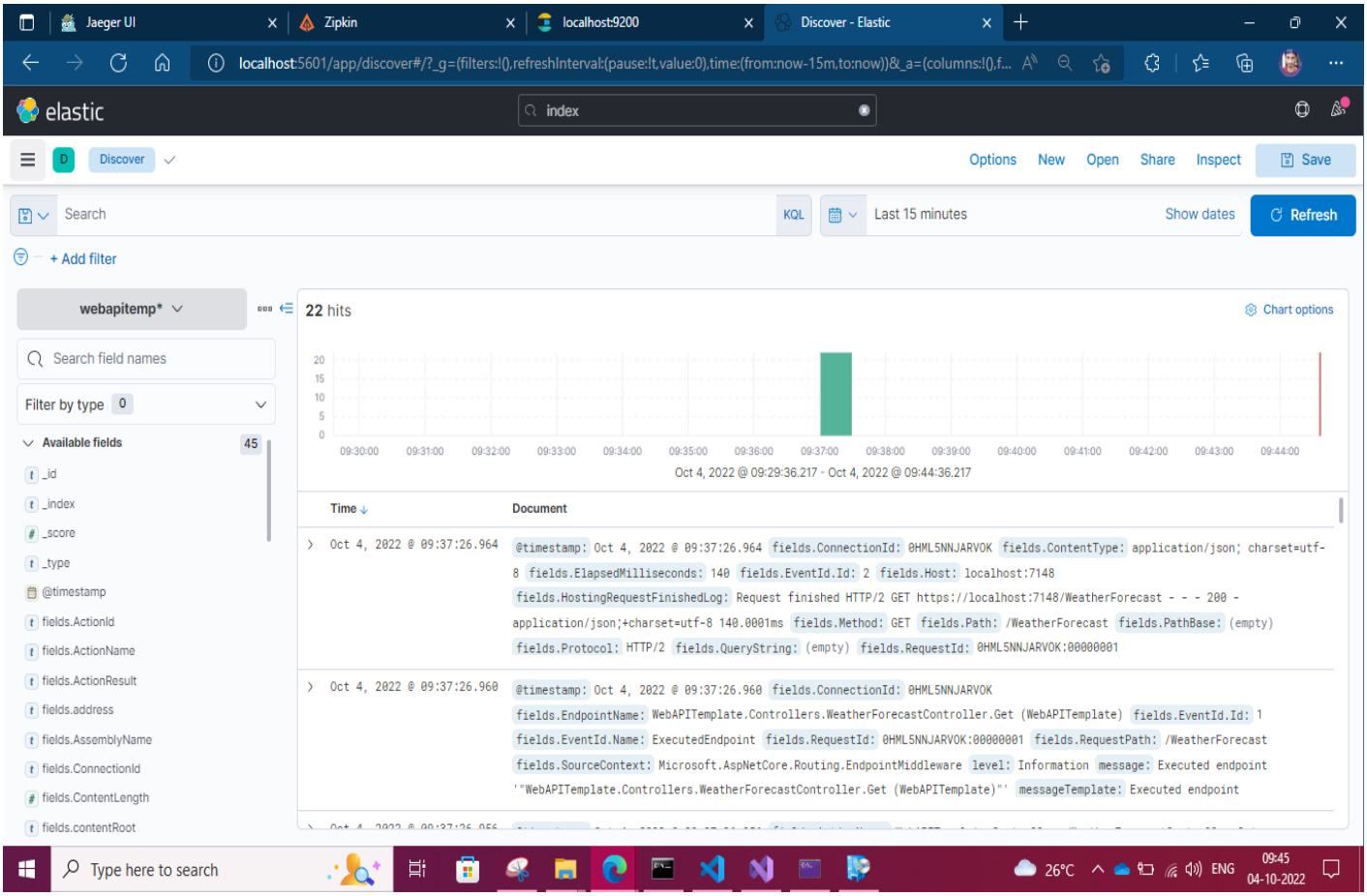
Fields (82) | Scripted fields (0) | Field filters (0)

Search | All field types | Add field

| Name | Type | Format | Searchable | Aggregatable | Excluded |
|---|---|---|---|---|---|
| @timestamp | date | | ● | ● | |
| _id | _id | | ● | ● | |
| _index | _index | | ● | ● | |
| _score | | | | | |
| _source | _source | | | | |
| _type | _type | | ● | ● | |
| fields.ActionId | text | | ● | | |
| fields.ActionId.keyword | keyword | | ● | ● | |

Type here to search | 26°C | ENG | 09:42 04-10-2022

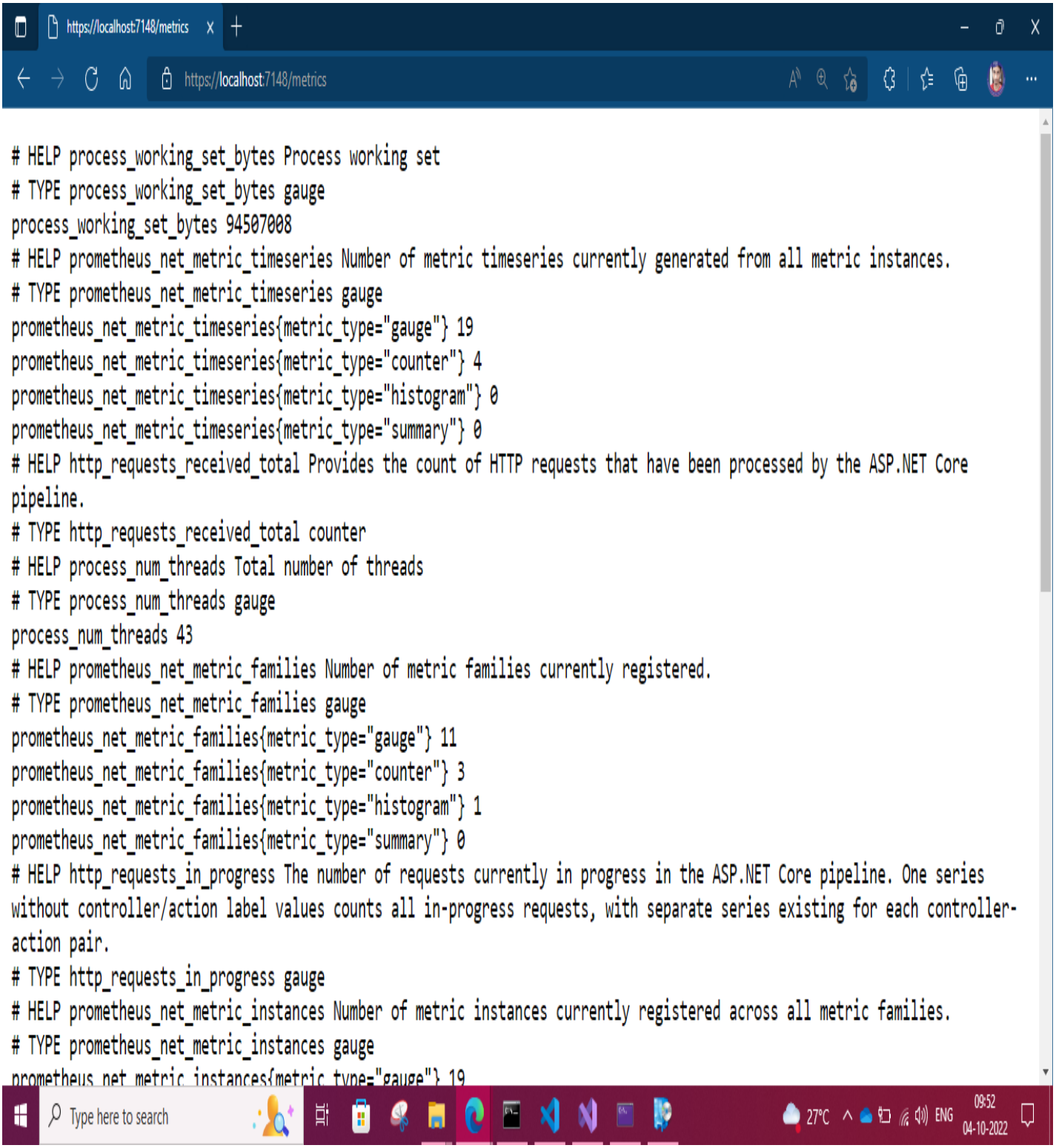
Click the left toggle and select Discover.




Now, we will add Prometheus metrics. Add these codes to Program.cs.

```
using Prometheus;


app.MapControllers();

app.MapMetrics();

app.UseHttpMetrics();

app.Run();
```

Now, run the API and browse http://localhost:7148/metrics. The port number may vary in your API template.



```
# HELP process_working_set_bytes Process working set
# TYPE process_working_set_bytes gauge
process_working_set_bytes 94507008
# HELP prometheus_net_metric_timeseries Number of metric timeseries currently generated from all metric instances.
# TYPE prometheus_net_metric_timeseries gauge
prometheus_net_metric_timeseries{metric_type="gauge"} 19
prometheus_net_metric_timeseries{metric_type="counter"} 4
prometheus_net_metric_timeseries{metric_type="histogram"} 0
prometheus_net_metric_timeseries{metric_type="summary"} 0
# HELP http_requests_received_total Provides the count of HTTP requests that have been processed by the ASP.NET Core
pipeline.
# TYPE http_requests_received_total counter
# HELP process_num_threads Total number of threads
# TYPE process_num_threads gauge
process_num_threads 43
# HELP prometheus_net_metric_families Number of metric families currently registered.
# TYPE prometheus_net_metric_families gauge
prometheus_net_metric_families{metric_type="gauge"} 11
prometheus_net_metric_families{metric_type="counter"} 3
prometheus_net_metric_families{metric_type="histogram"} 1
prometheus_net_metric_families{metric_type="summary"} 0
# HELP http_requests_in_progress The number of requests currently in progress in the ASP.NET Core pipeline. One series
without controller/action label values counts all in-progress requests, with separate series existing for each controller-
action pair.
# TYPE http_requests_in_progress gauge
# HELP prometheus_net_metric_instances Number of metric instances currently registered across all metric families.
# TYPE prometheus_net_metric_instances gauge
prometheus_net_metric_instances{metric_type="gauge"} 19
```

Now, we will add our own user-defined logs to Get() in the controller.

```
WeatherForecastController.cs
WebAPITemplate | WebAPITemplate.Controllers.WeatherForecastContr | Get()

1   using Microsoft.AspNetCore.Mvc;
2   using OpenTelemetry.Trace;
3
4   namespace WebAPITemplate.Controllers
5   {
6       [ApiController]
7       [Route("[controller]")]
        3 references
8       public class WeatherForecastController : ControllerBase
9       {
10          private static readonly string[] Summaries = new[]
11          {
12          "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering",
13      };
14
15          private readonly ILogger<WeatherForecastController> _logger;
16          private readonly Tracer tracer;
17
            0 references
18          public WeatherForecastController(ILogger<WeatherForecastController> logger,
19              TracerProvider provider)
20          {
21              _logger = logger;
22              tracer = provider.GetTracer("TraceSource");
23          }
24
```

```
24
25          [HttpGet("GetWeatherForecast")]
            0 references
26          public IEnumerable<WeatherForecast> Get()
27          {
28              using (var span = tracer.StartActiveSpan("Get"))
29              {
30                  span.SetAttribute("Method", "Returning the list of forecasts to the UI....");
31                  span.AddEvent("The list of forecasts is being returned to the Swagger......");
32              }
33
34              return Enumerable.Range(1, 6).Select(index => new WeatherForecast
35              {
36                  Date = DateTime.Now.AddDays(index),
37                  TemperatureC = Random.Shared.Next(-20, 55),
38                  Summary = Summaries[Random.Shared.Next(Summaries.Length)]
39              })
40              .ToArray();
41          }
42      }
43  }
44
```

Run the API template and check the results in Jaeger, Zipkin and ELK Stack.

Swagger UI

https://localhost:7148/swagger/index.html

# WebAPITemplate 1.0 OAS3

https://localhost:7148/swagger/v1/swagger.json

## WeatherForecast

GET /WeatherForecast/GetWeatherForecast

Parameters

Cancel

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7148/WeatherForecast/GetWeatherForecast' \
  -H 'accept: text/plain'
```

Request URL

https://localhost:7148/WeatherForecast/GetWeatherForecast

Server response

Code Details

200

Response body


```
[
  {
    "date": "2022-10-05T10:06:36.6307415+05:30",
    "temperatureC": 7,
    "temperatureF": 44,
    "summary": "Scorching"
```

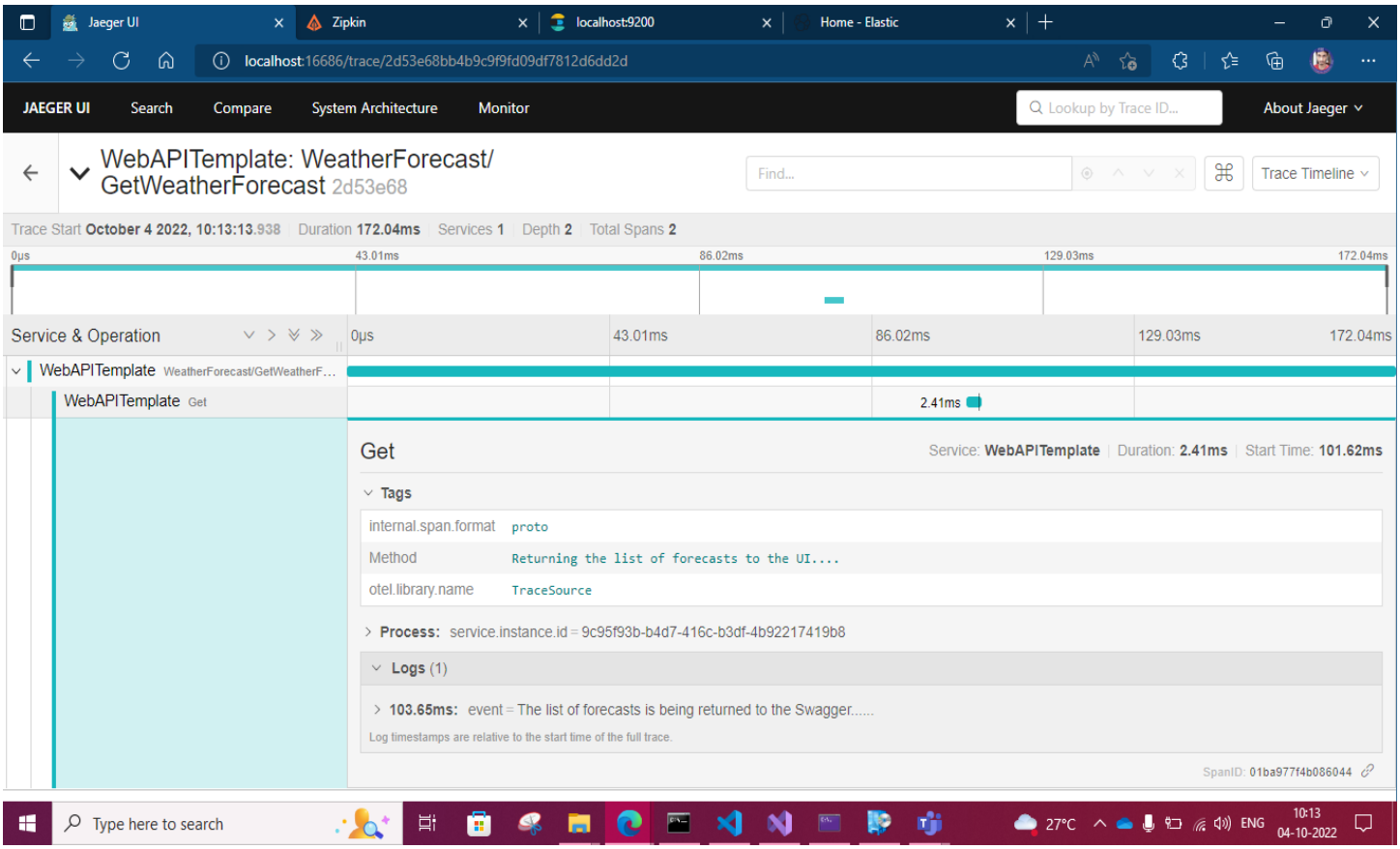

Type here to search

27°C ENG 10:06 04-10-2022

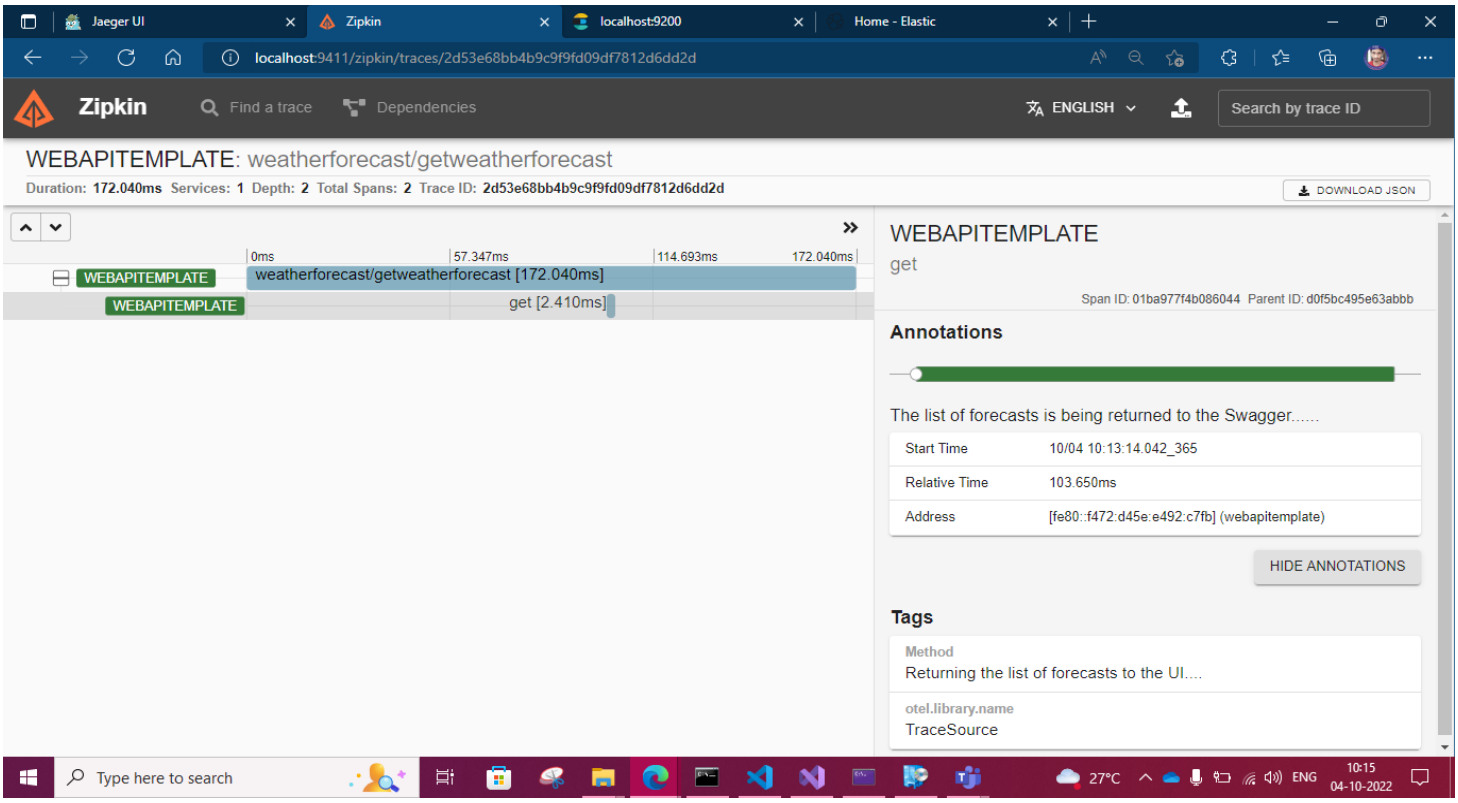
C:\Users\2048766\source\repos\WebAPITemplate\WebAPITemplate\bin\Debug\net6.0\WebAPITemplate.exe

```
[10:03:36 INF] Now listening on: https://localhost:7148
[10:03:36 INF] Now listening on: http://localhost:5148
[10:03:36 INF] Application started. Press Ctrl+C to shut down.
[10:03:36 INF] Hosting environment: Development
[10:03:36 INF] Content root path: C:\Users\2048766\source\repos\WebAPITemplate\WebAPITemplate\
Activity.TraceId:            b689324ec56b8d73b54d95639da0cffa
Activity.SpanId:             5bfd44d62e4bd680
Activity.TraceFlags:         Recorded
Activity.ActivitySourceName: OpenTelemetry.Instrumentation.Http.HttpClient
Activity.DisplayName:        HTTP POST
Activity.Kind:               Client
Activity.StartTime:          2022-10-04T04:33:36.7820082Z
Activity.Duration:           00:00:00.1200046
Activity.Tags:
    http.scheme: http
    http.method: POST
    http.host: localhost:9200
    http.url: http://localhost:9200/_bulk
    http.flavor: 1.1
    http.status_code: 200
Resource associated with Activity:
    service.name: WebAPITemplate
    service.instance.id: d4a80ca4-a891-4a34-aa6d-6af3f50bc4ff

Activity.TraceId:            b4cf3cfe0c358f91db2586effaaaf09c
Activity.SpanId:             2cc85e0a278dd38a
Activity.TraceFlags:         Recorded
Activity.ActivitySourceName: OpenTelemetry.Instrumentation.Http.HttpClient
Activity.DisplayName:        HTTP POST
Activity.Kind:               Client
```

## Jaeger

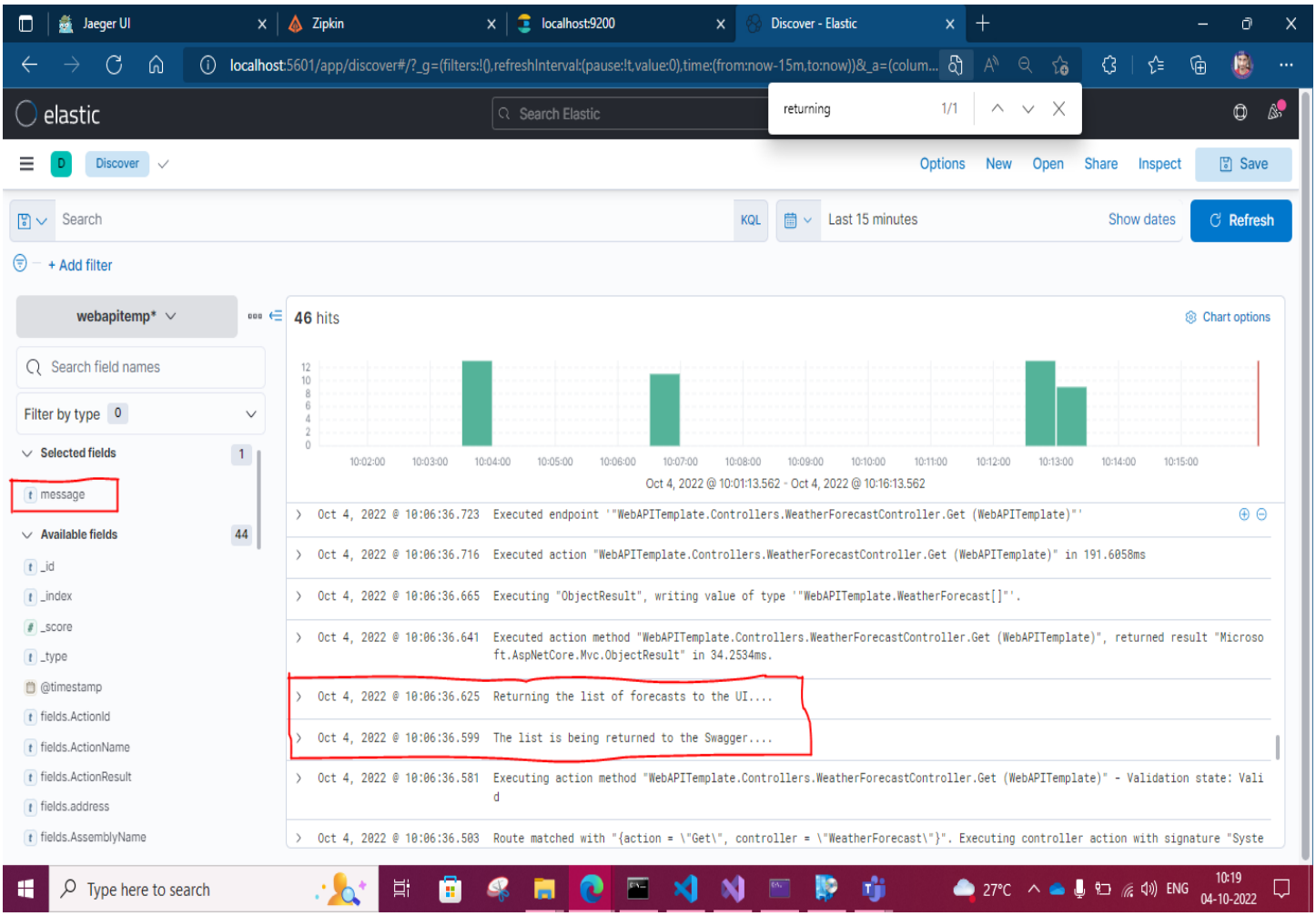



## Zipkin



## ELK Stack

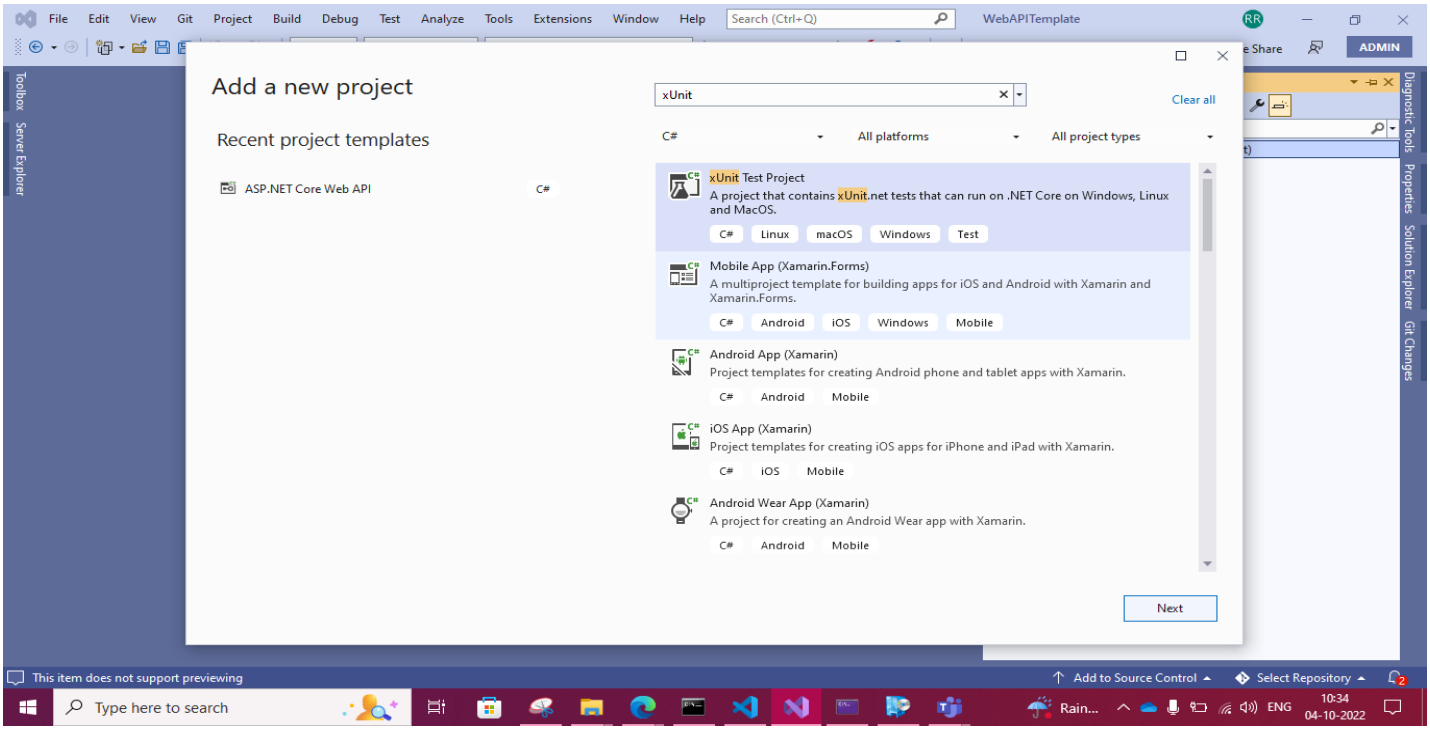
Refresh the Kibana webpage. If possible, refresh the Discover page. Then, reselect the filter in the left sidebar. Select message from the left sidebar in order to view the logs clearly.



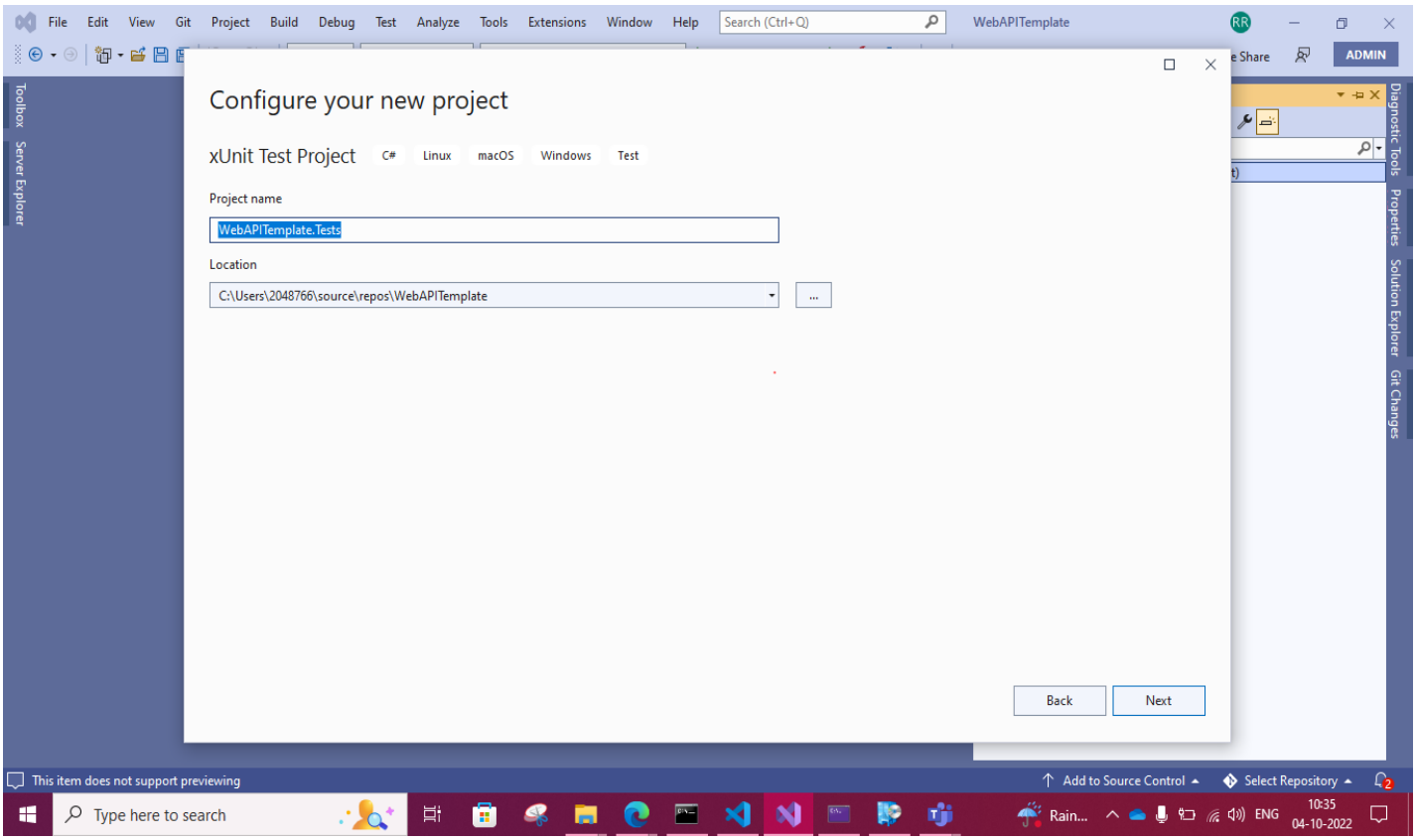

In this way, we can trace each and every request as well as response made by any source to our website. This is a really advanced concept and we should try to master this.

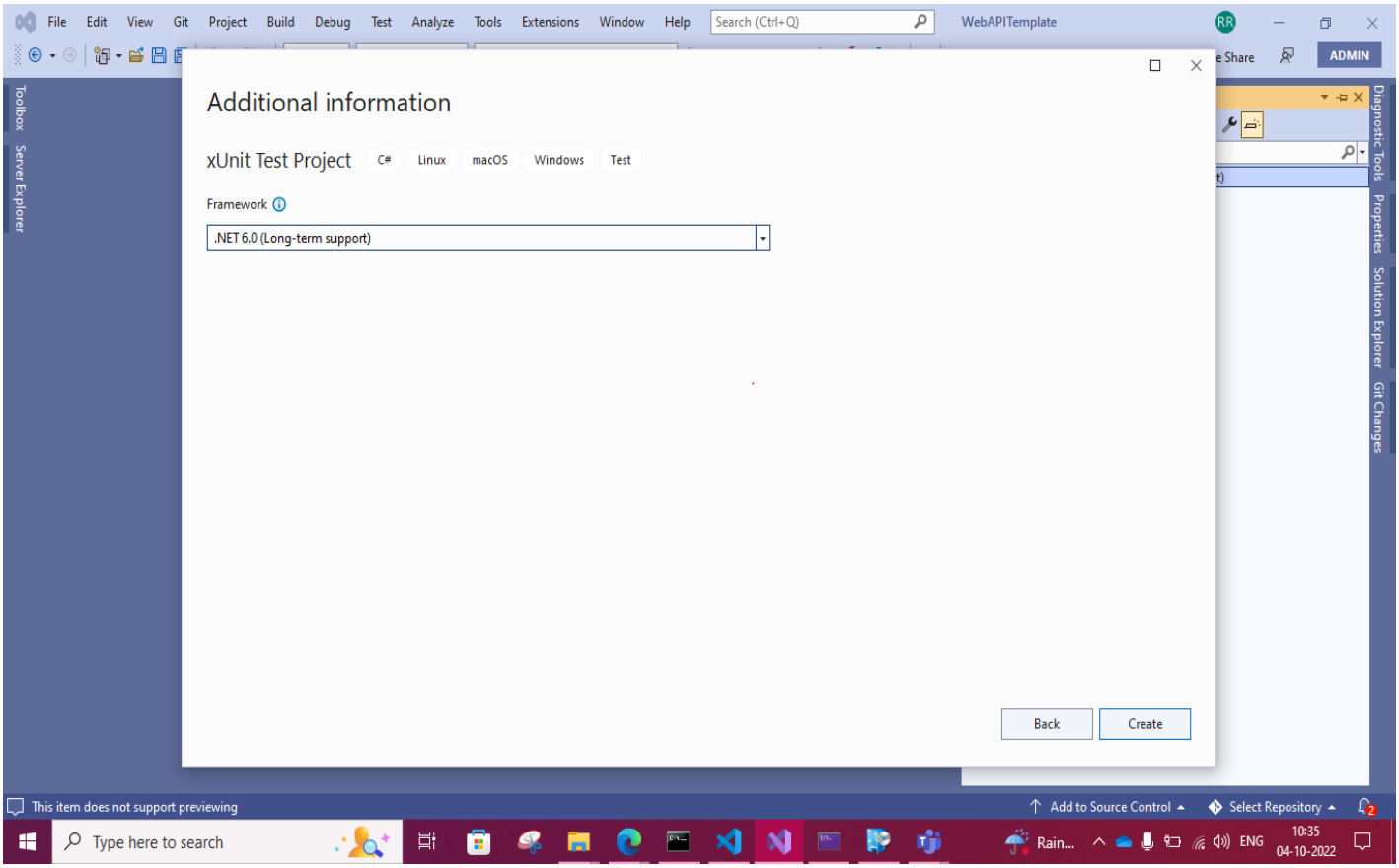
Now, we will add Unit tests.

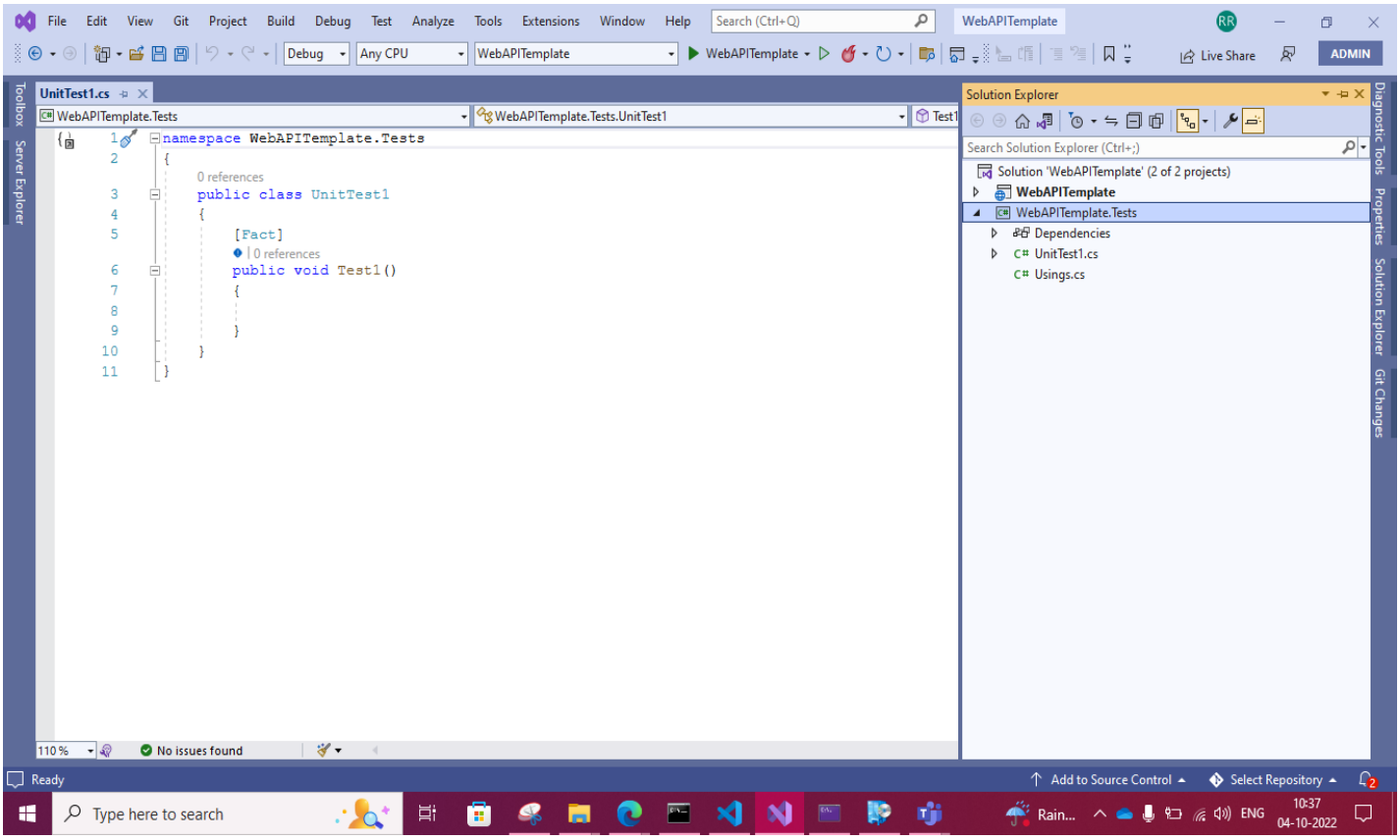
Right-click the Solution Folder and select Add -> Click New Project.



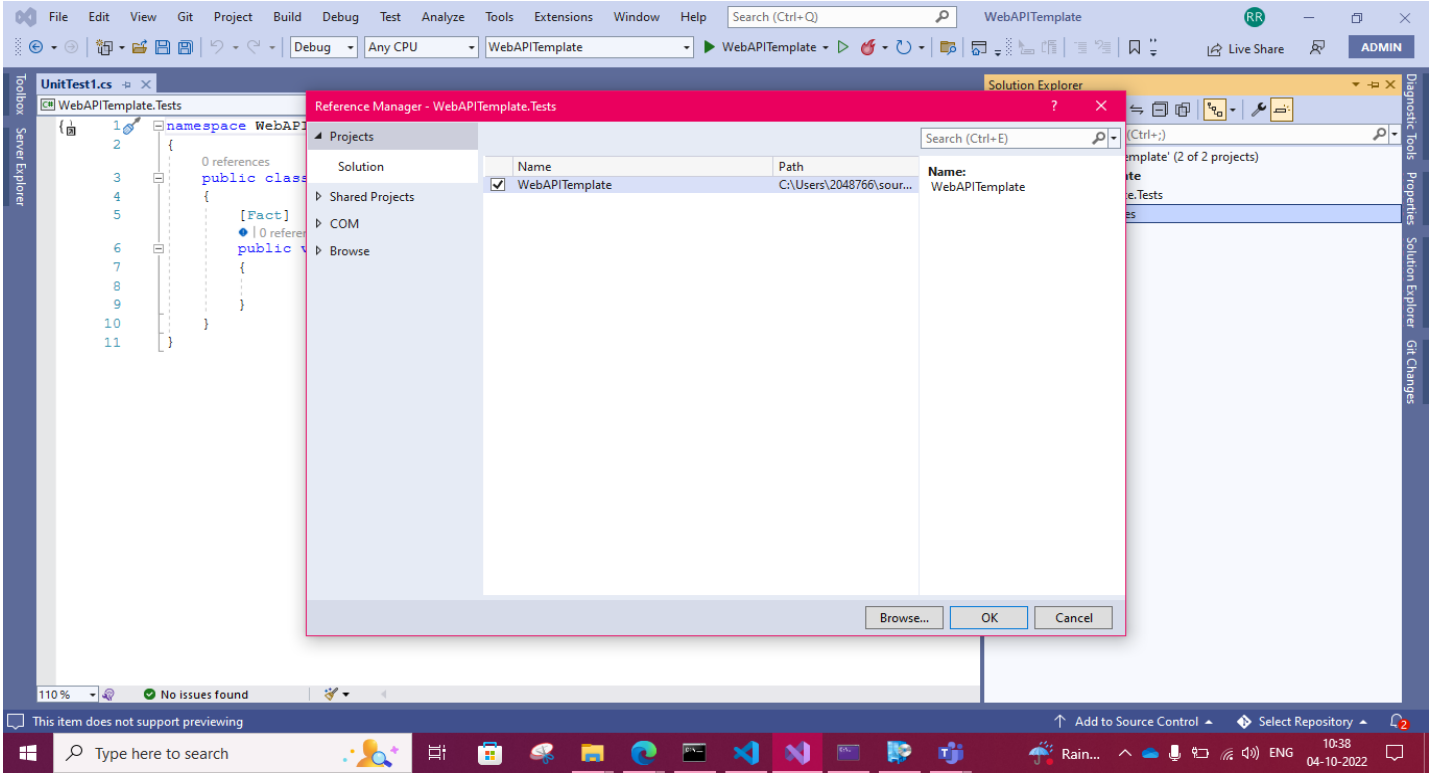

Search xUnit and then select xUnit Test Project. Then click on Next.





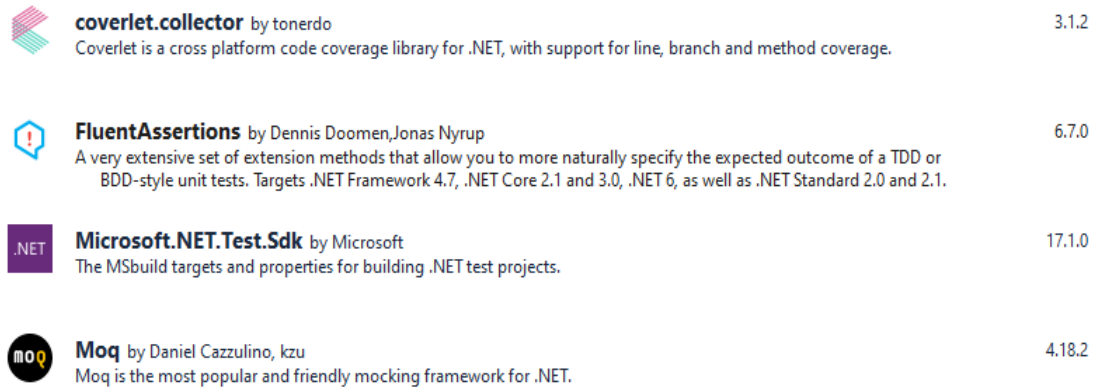




Now, Right-click Dependencies in WebAPITemplate.Tests and select Add Project Reference.

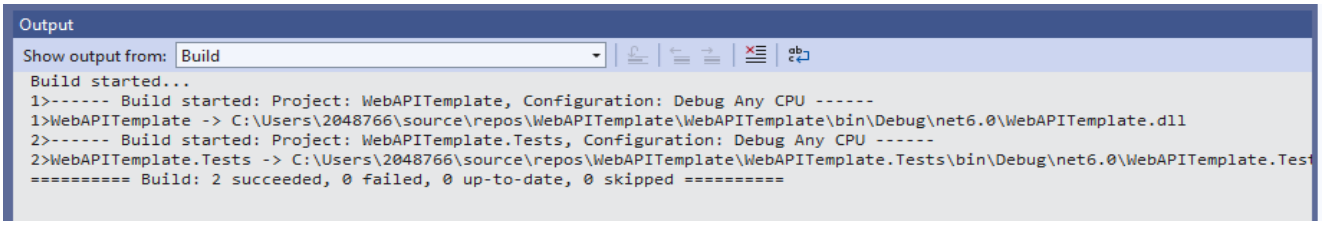



Select the checkbox and click OK. Now right-click the Dependencies and select Manage NuGet Packages. Then install

- Moq
- FluentAssertions



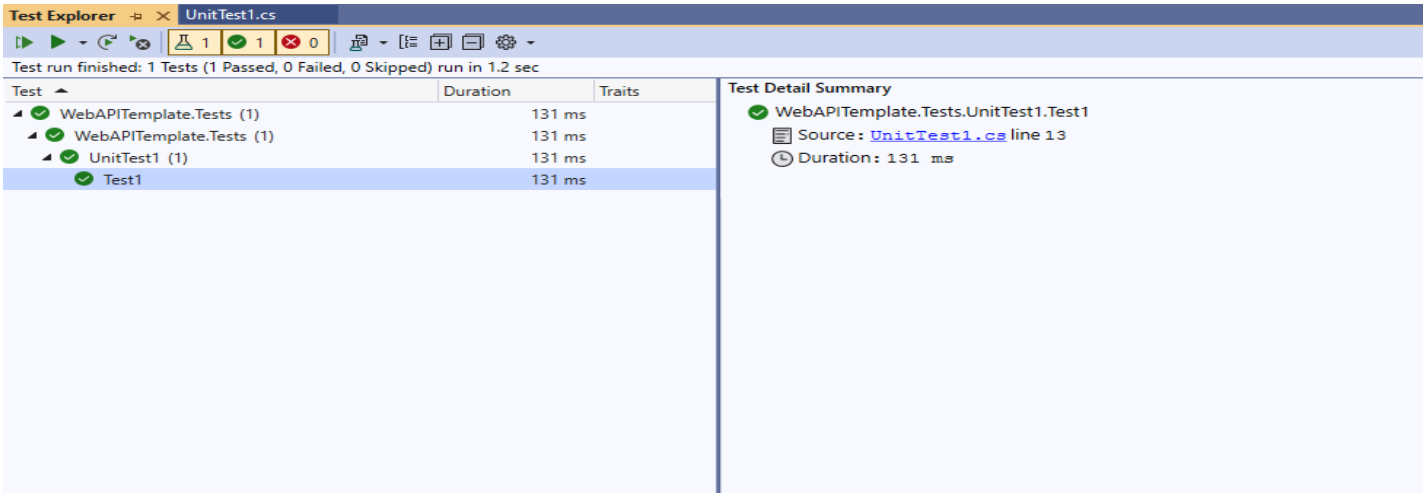

Then, right-click the WebAPITemplate.Tests and click Build. This will build the Unit Test.




Now, add the following codes to UnitTest1.cs

```
1   using FluentAssertions;
2   using Microsoft.Extensions.Logging;
3   using Moq;
4   using OpenTelemetry.Trace;
5   using WebAPITemplate;
6   using WebAPITemplate.Controllers;
7
8   namespace WebAPITemplate.Tests
9   {
        0 references
10      public class UnitTest1
11      {
12          [Fact]
            | 0 references
13          public void Test1()
14          {
15              // Arrange
16              var logger = new Mock<ILogger<WeatherForecastController>>();
17              var provider = new Mock<TracerProvider>();
18
19              // Act
20              var controller = new WeatherForecastController(logger.Object, provider.Object);
21              var result = controller.Get();
22
23              // Assert
24              Assert.IsAssignableFrom<IEnumerable<WeatherForecast>>(result);
25              Assert.Equal(6, result.Count());
26          }
27      }
28  }
```

Click Test Explorer from the View menu and run all the tests.




All the test cases are successful

*Thank You!*