Create Web API Project Lab

Perform these labs on your own computer using Visual Studio 2022 to ensure you understand the lessons presented in the corresponding videos and lectures.

Lab 1: Create Web API Project Using Visual Studio 2022

Startup Visual Studio 2022 and select **Create New Project** as shown in Figure 1.

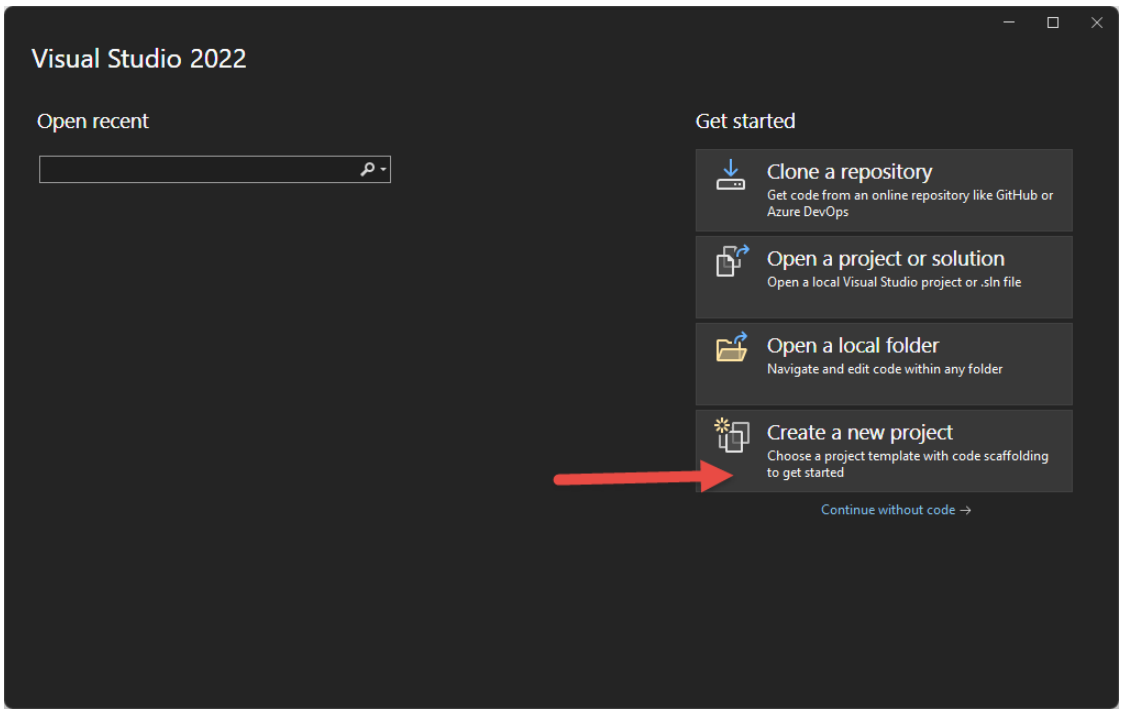

Figure 1: Select what you want to do in Visual Studio startup screen

Create a New Project Screen

Locate the project template **ASP.NET Core Web API** and select that one as shown in Figure 2.

Click the **Next** button to continue to the next screen

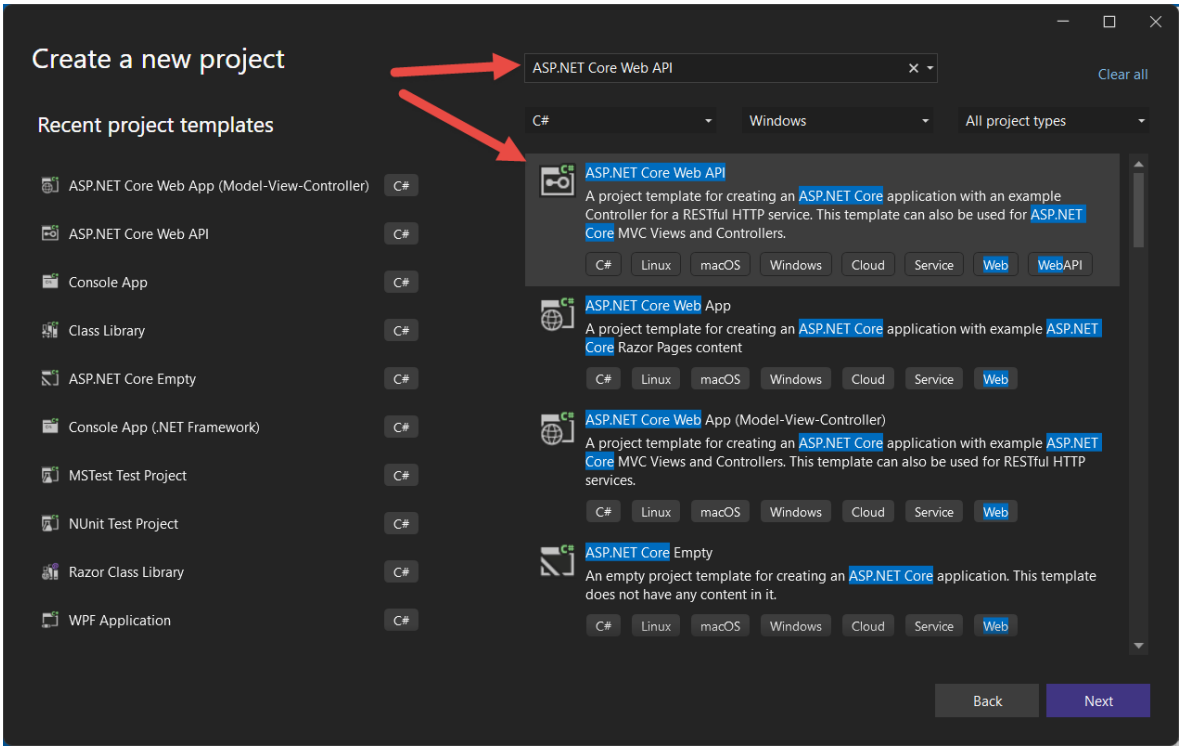

Figure 2: Select the ASP.NET Core Web API Project.

Configure Your New Project Screen

Set the **Project name** to **AdvWorksAPI**.

Set the **Location** to where you want the project to reside.

Check the Place solution and project in the same directory check box as shown in Figure 3.

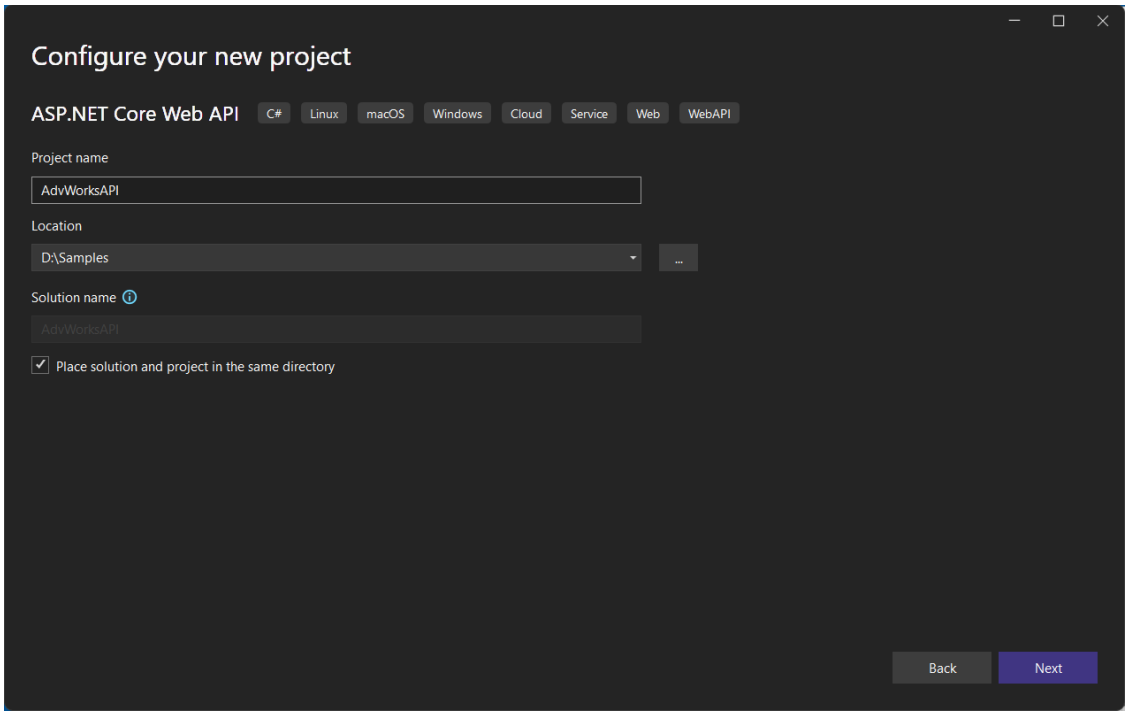

Figure 3: Configure your new project

Additional Information Screen

Choose .NET 6.0 (Long-term support)

Choose Authentication Type = None

Uncheck Configure for HTTPS

Check the "Use controllers (uncheck to use minimal APIs)".

Check Enable OpenAPI support as shown in Figure 4.

Click the **Create** button to create the new project.

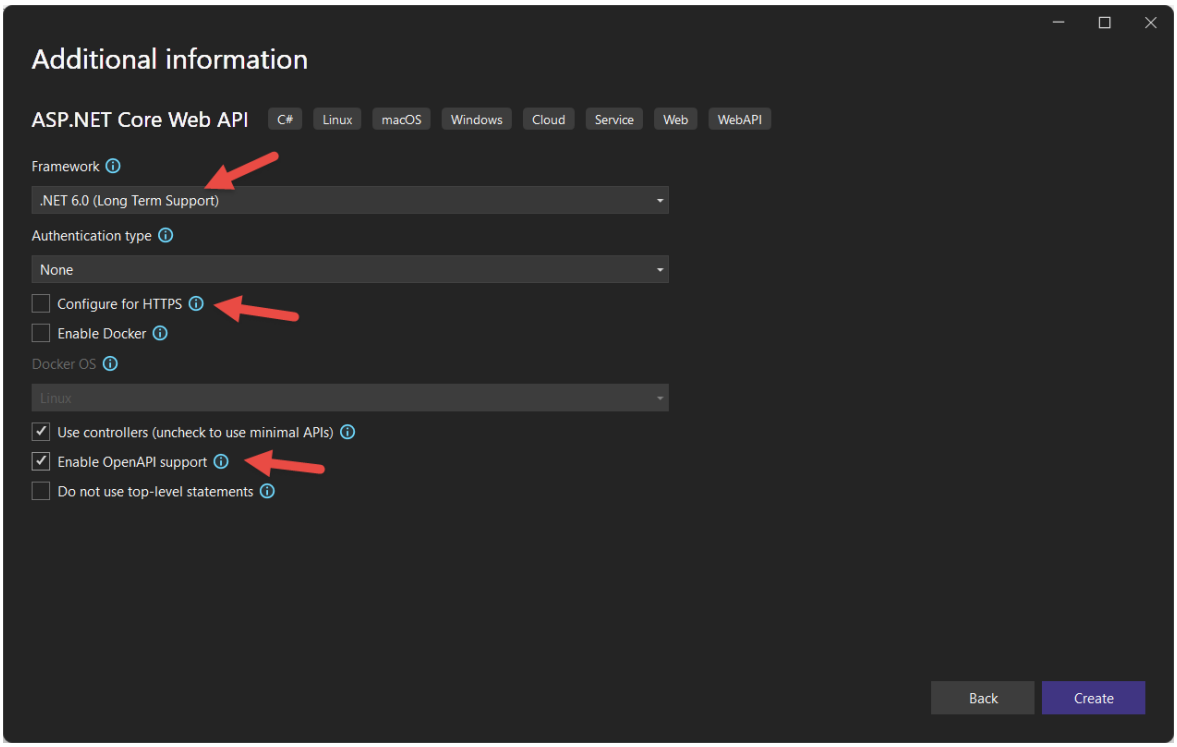

Figure 4: Set the project information

Try it Out

Select **Debug | Start Debugging** (F5) from the VS menu to build the Web API project and launch a browser.

NOTE:	If you get a dialog box that asks if you should trust the IIS Express certificate, select **Yes**. In the Security Warning dialog that appears next, select **Yes**.

When the browser appears, it will look like Figure 5.

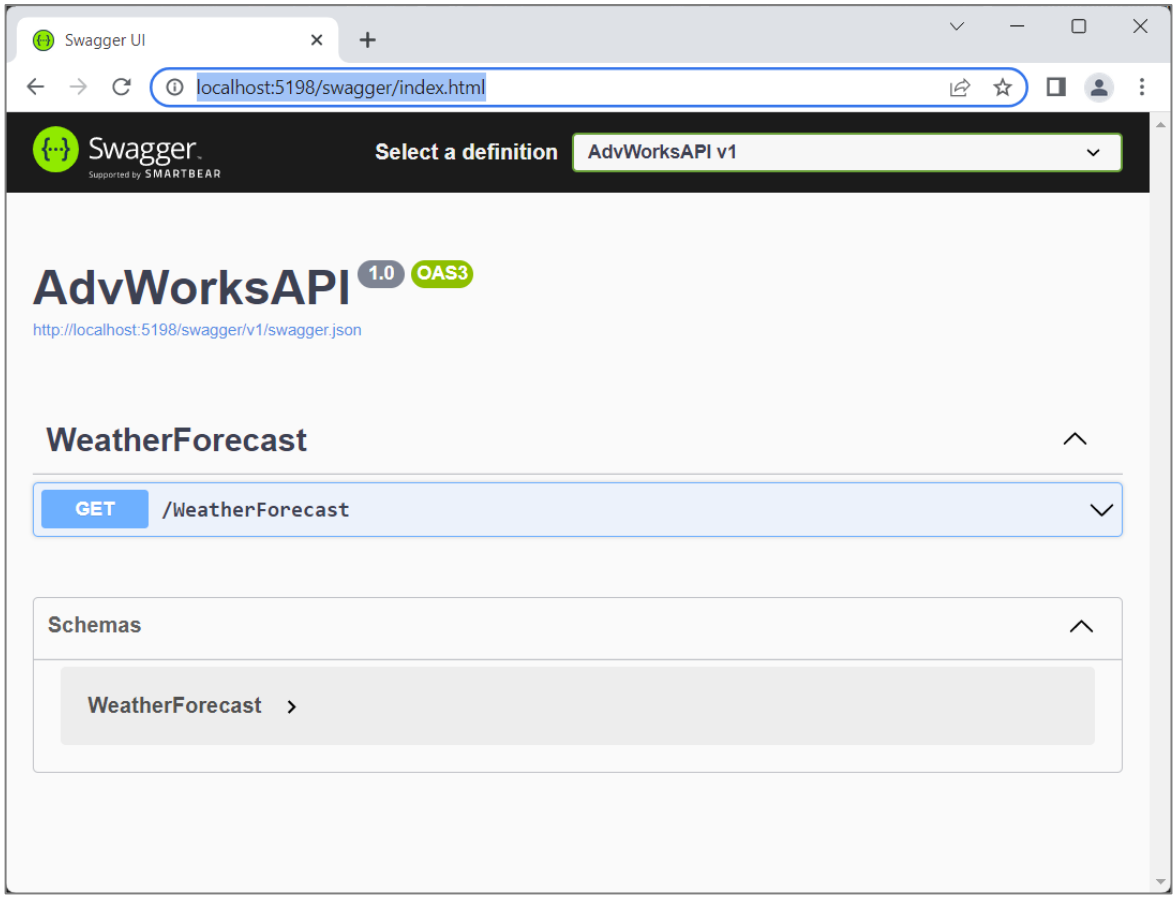

Figure 5: The Swagger Open API page is displayed

Click on the **GET /WeatherForecast** button to display some weather information.

Lab 2: Comment the Program.cs File

The Program.cs file is a template that comes from Microsoft. It is not very well documented, so let's add some comments.

Open the **Program.cs** file from the Solution Explorer window and replace all of contents with the code shown below.

```
// ***********************************************
// Create a WebApplicationBuilder object
// to configure the how the ASP.NET service runs
// ***********************************************
var builder = WebApplication.CreateBuilder(args);

// ***********************************************
// Add and Configure Services
// ***********************************************


// Configure ASP.NET to use the Controller model
builder.Services.AddControllers();

// Configure Open API (Swagger)
// More Info: https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// ***********************************************
// After adding and configuring services
// Create an instance of a WebApplication object
// ***********************************************
var app = builder.Build();

// ***********************************************
// Configure the HTTP Request Pipeline
// ***********************************************
if (app.Environment.IsDevelopment()) {
  // When in Development mode
  // Enable the Open API (Swagger) page
  app.UseSwagger();
  app.UseSwaggerUI();
}

// Enable Authorization Middleware
app.UseAuthorization();

// Enable the Endpoints of Controller Action Methods
app.MapControllers();

// Run the Application
app.Run();
```

Try it Out

Run the application to make sure everything is still working correctly.

Optional Lab 3: Using Postman

Navigate to https://www.postman.com/pricing/ and install the free version of Postman as shown in Figure 6.

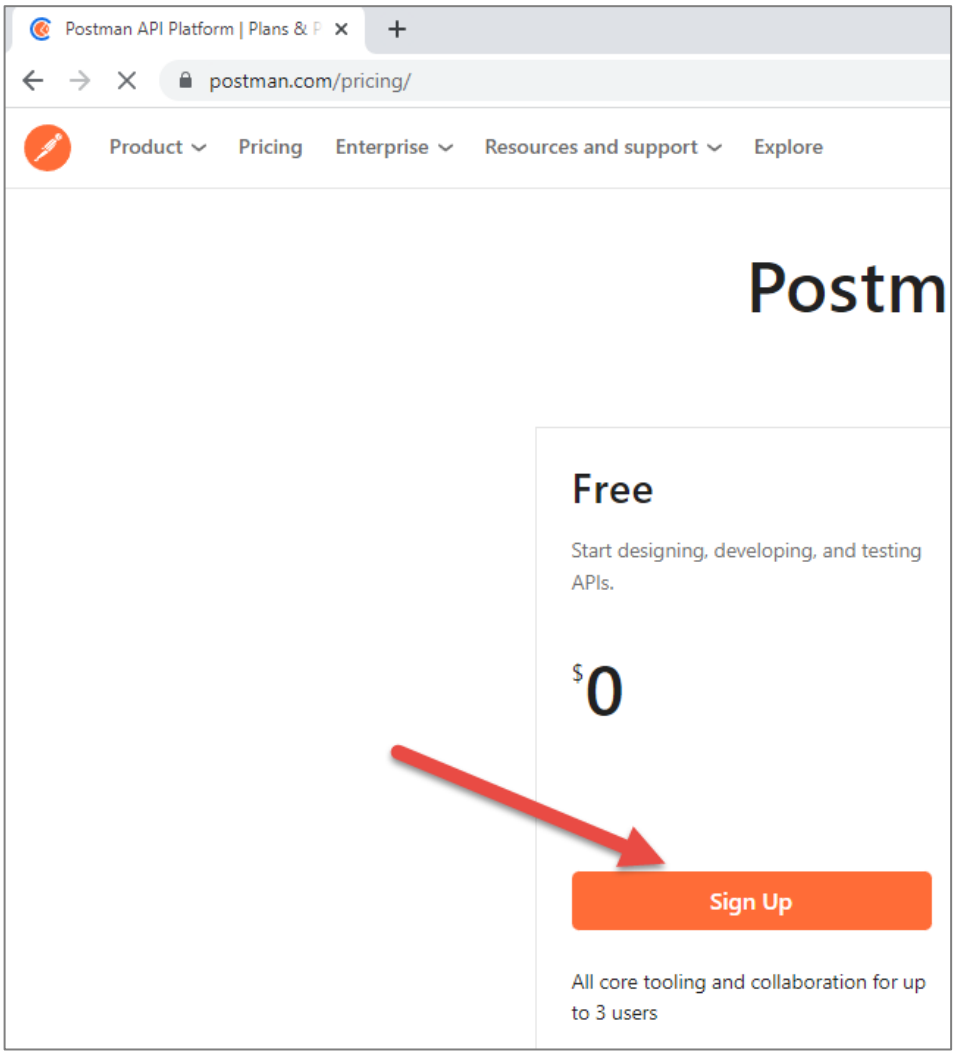

Figure 6: Install the free version of Postman

Try it Out

Open Postman (Figure 7) and enter the following

```
http://localhost:5198/weatherforecast
```

NOTE:	Change the PORT number with the port from your running Web API project.

Click on the Send button.

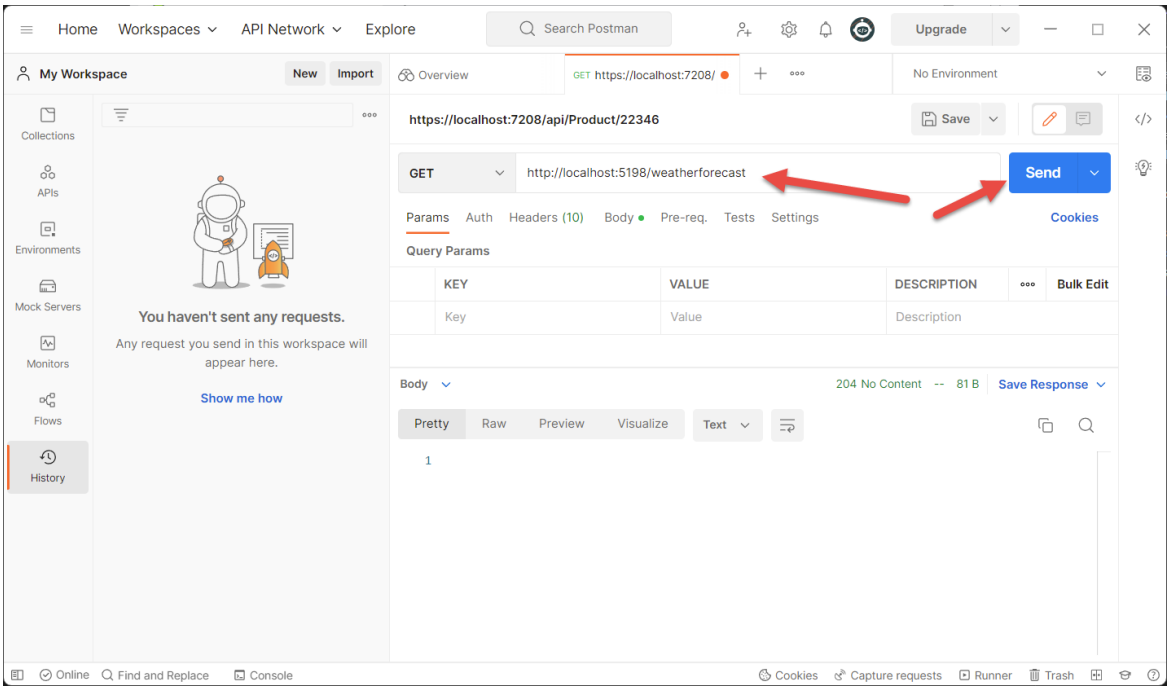

Figure 7: Postman allows you to try out your Web APIs

You should see the return results at the bottom of the Postman screen as shown in Figure 8.

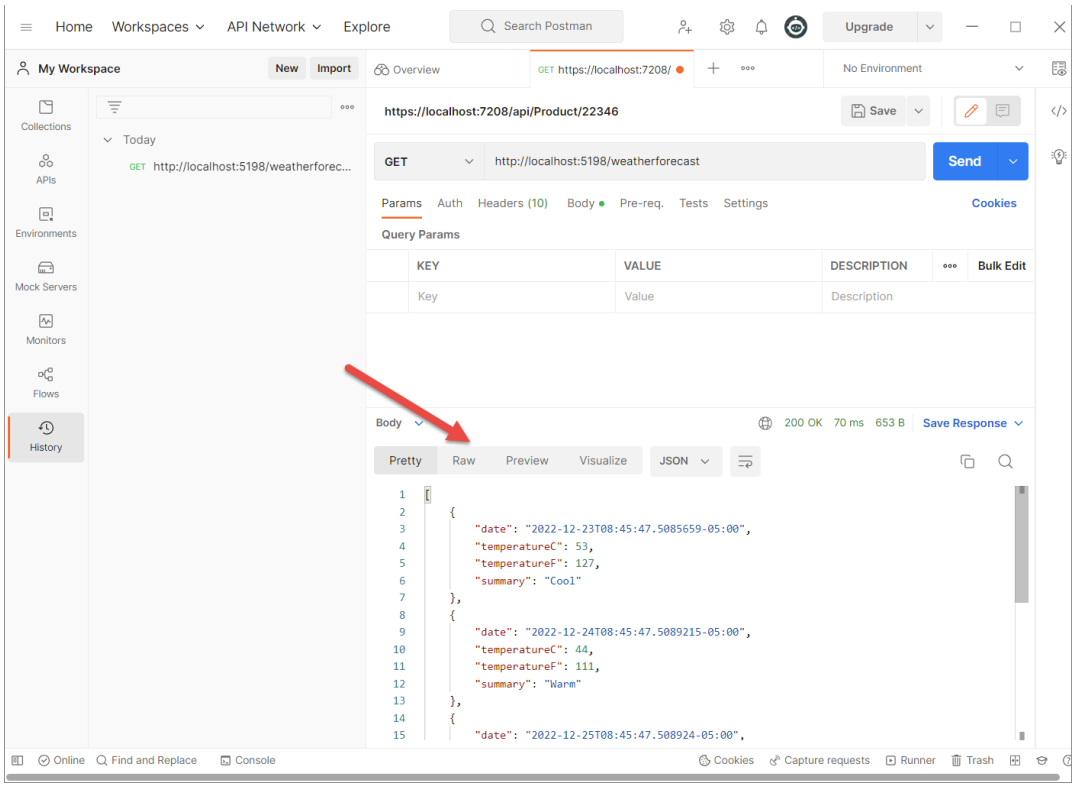

Figure 8: Postman displays the results in nicely formatted JSON