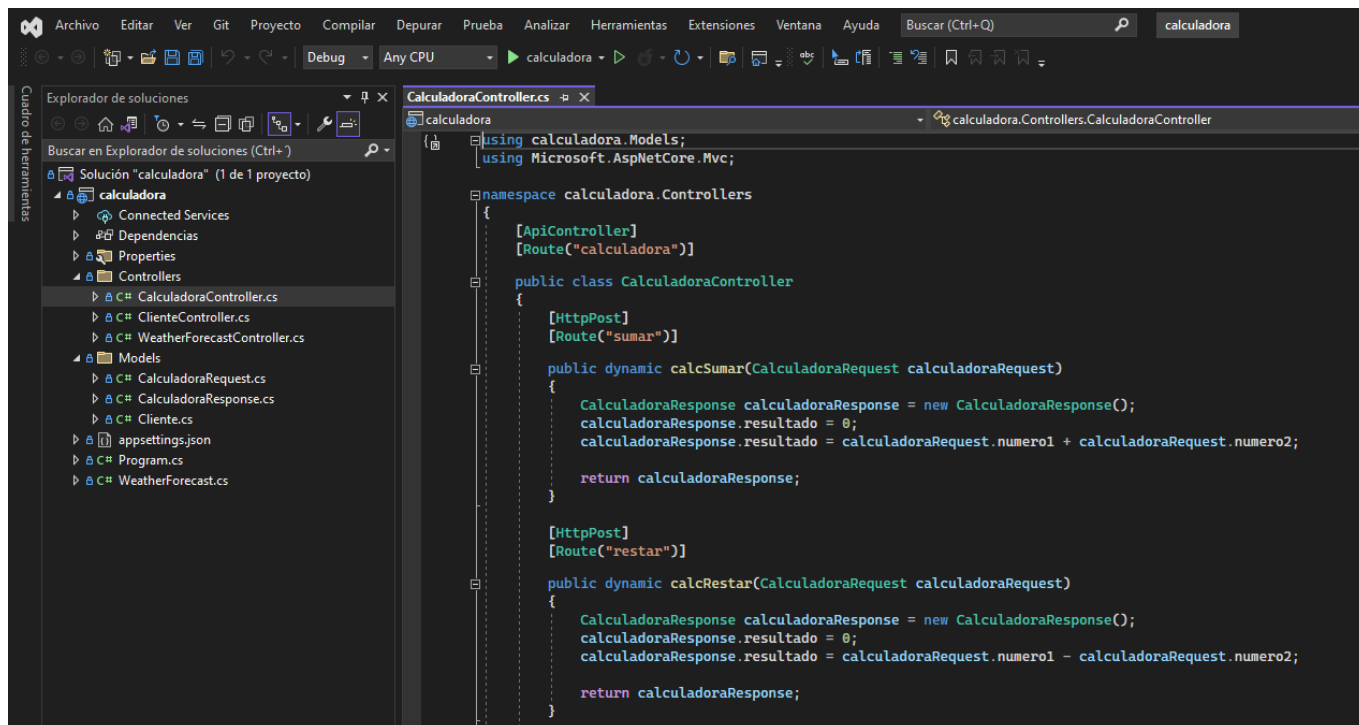


Código fuente:<https://github.com/juliopari/net6-csharp-calculadora-rest/tree/main/calculadora-vs>

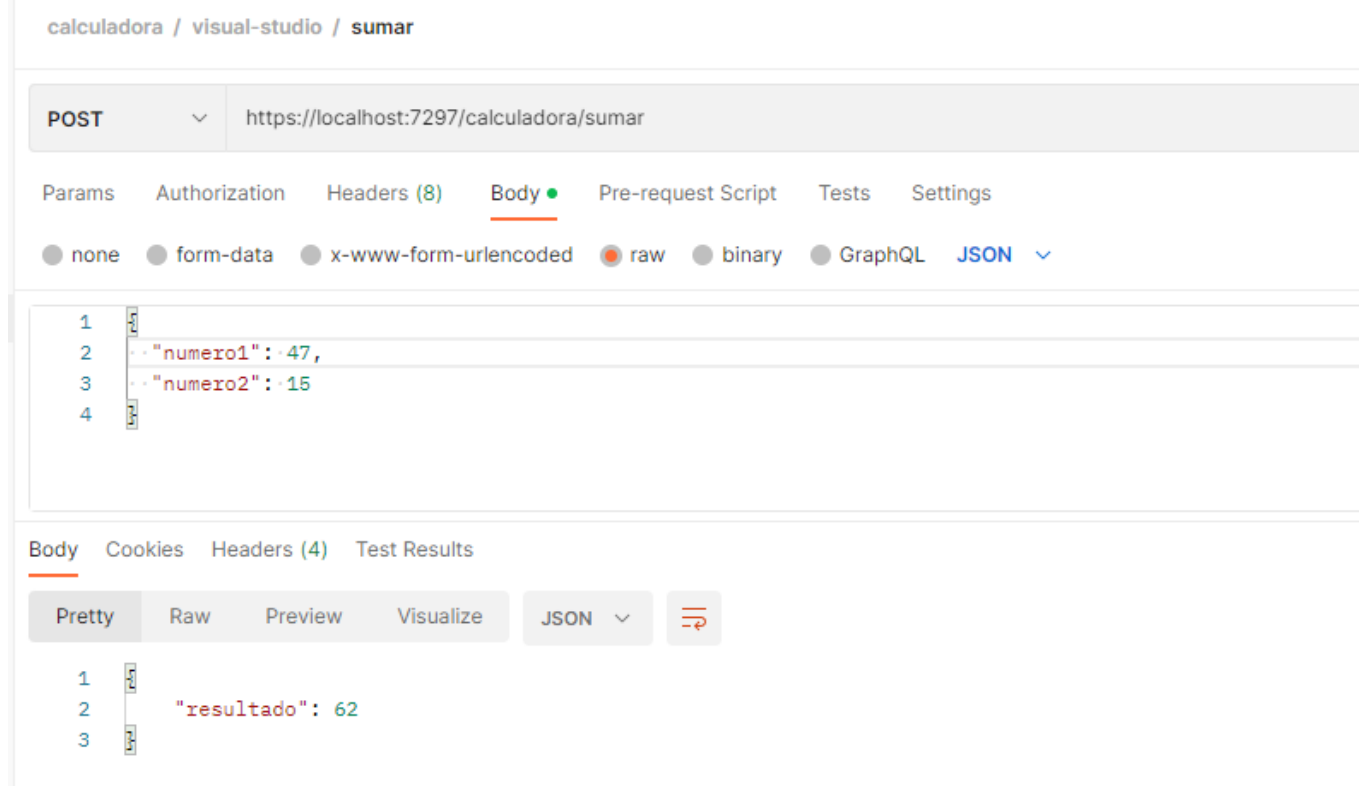
```
using calculadora.Models;
using Microsoft.AspNetCore.Mvc;

namespace calculadora.Controllers
{
    [ApiController]
    [Route("calculadora")]
    public class CalculadoraController
    {
        [HttpPost]
        [Route("sumar")]
        public dynamic calcSumar(CalculadoraRequest calculadoraRequest)
        {
            CalculadoraResponse calculadoraResponse = new CalculadoraResponse();
            calculadoraResponse.resultado = 0;
            calculadoraResponse.resultado = calculadoraRequest.numero1 + calculadoraRequest.numero2;

            return calculadoraResponse;
        }

        [HttpPost]
        [Route("restar")]
        public dynamic calcRestar(CalculadoraRequest calculadoraRequest)
        {
            CalculadoraResponse calculadoraResponse = new CalculadoraResponse();
            calculadoraResponse.resultado = 0;
            calculadoraResponse.resultado = calculadoraRequest.numero1 - calculadoraRequest.numero2;

            return calculadoraResponse;
        }
    }
}
```



calculadora / visual-studio / sumar

POST ▼ https://localhost:7297/calculadora/sumar

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** ▼

```
1 {
2   "numero1": 47,
3   "numero2": 15
4 }
```

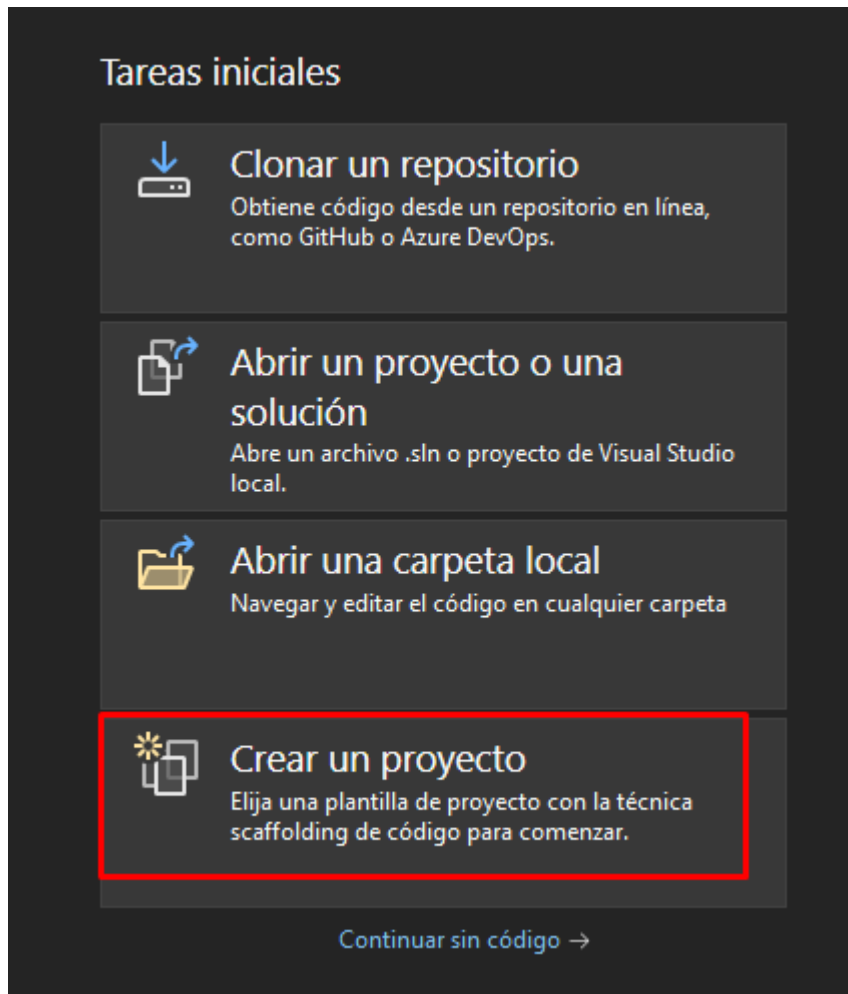
Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize JSON ▼ ↻

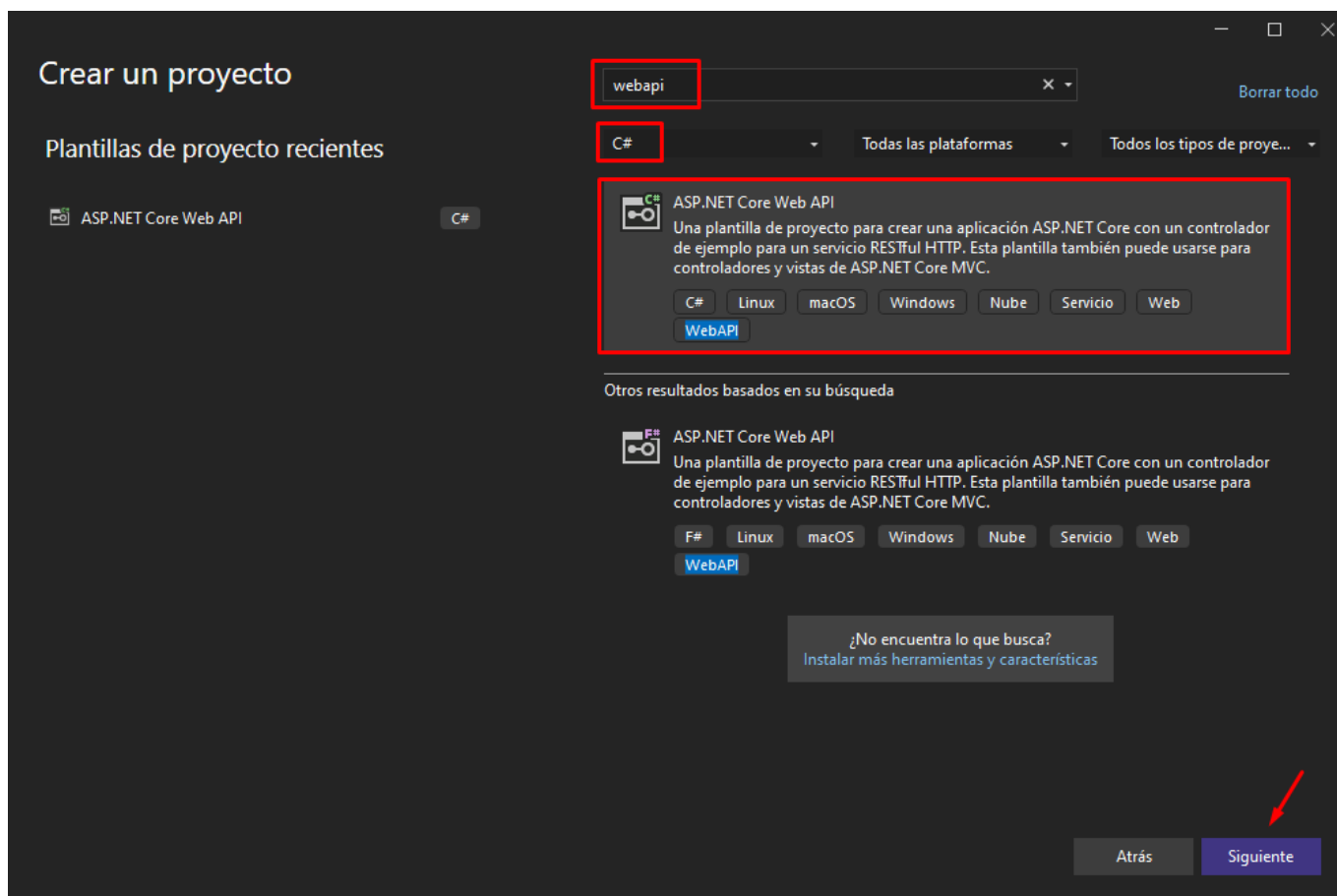
```
1 {
2   "resultado": 62
3 }
```

Pasos iniciales

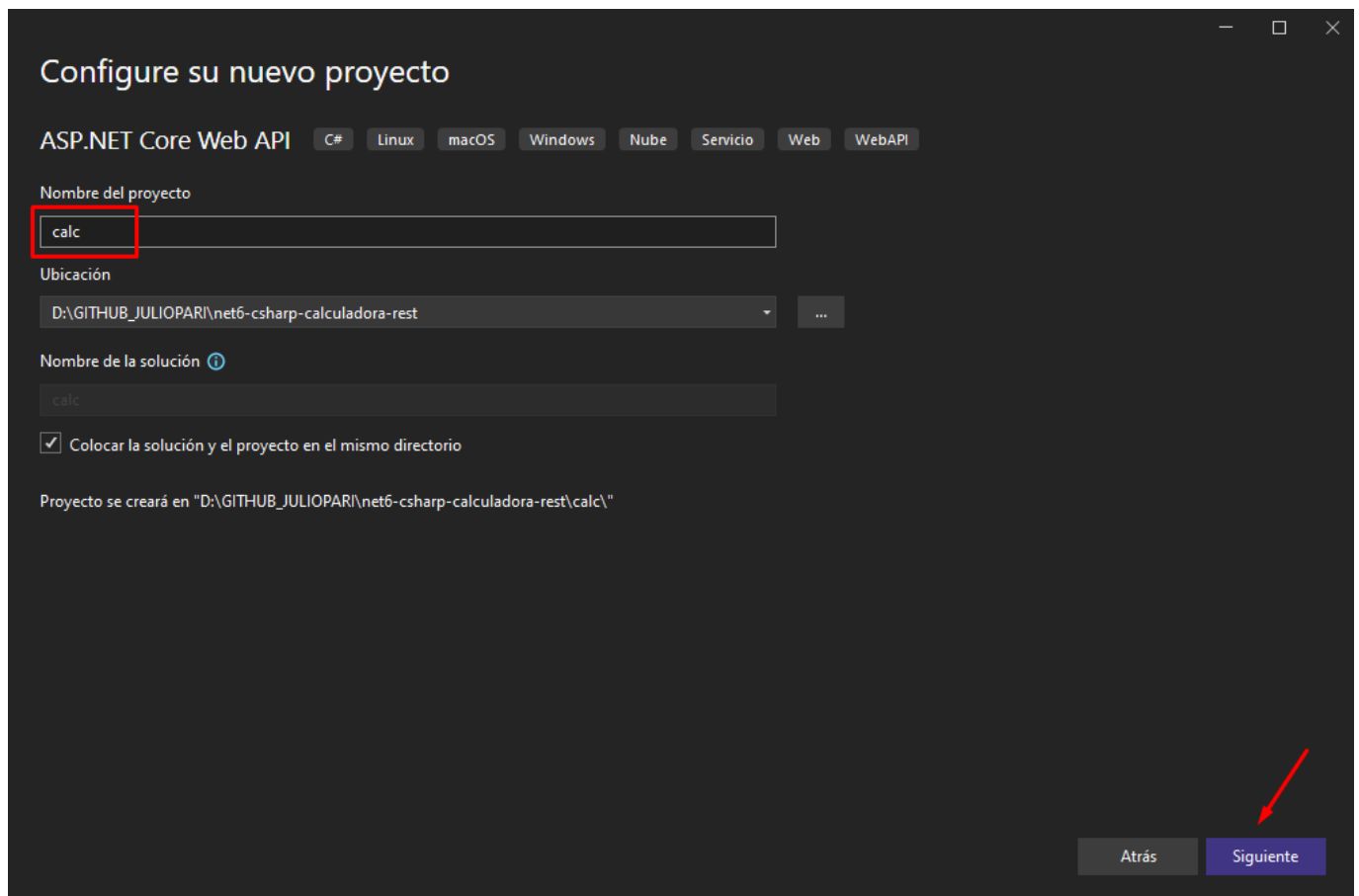
_1. Seleccionar «Crear un proyecto»



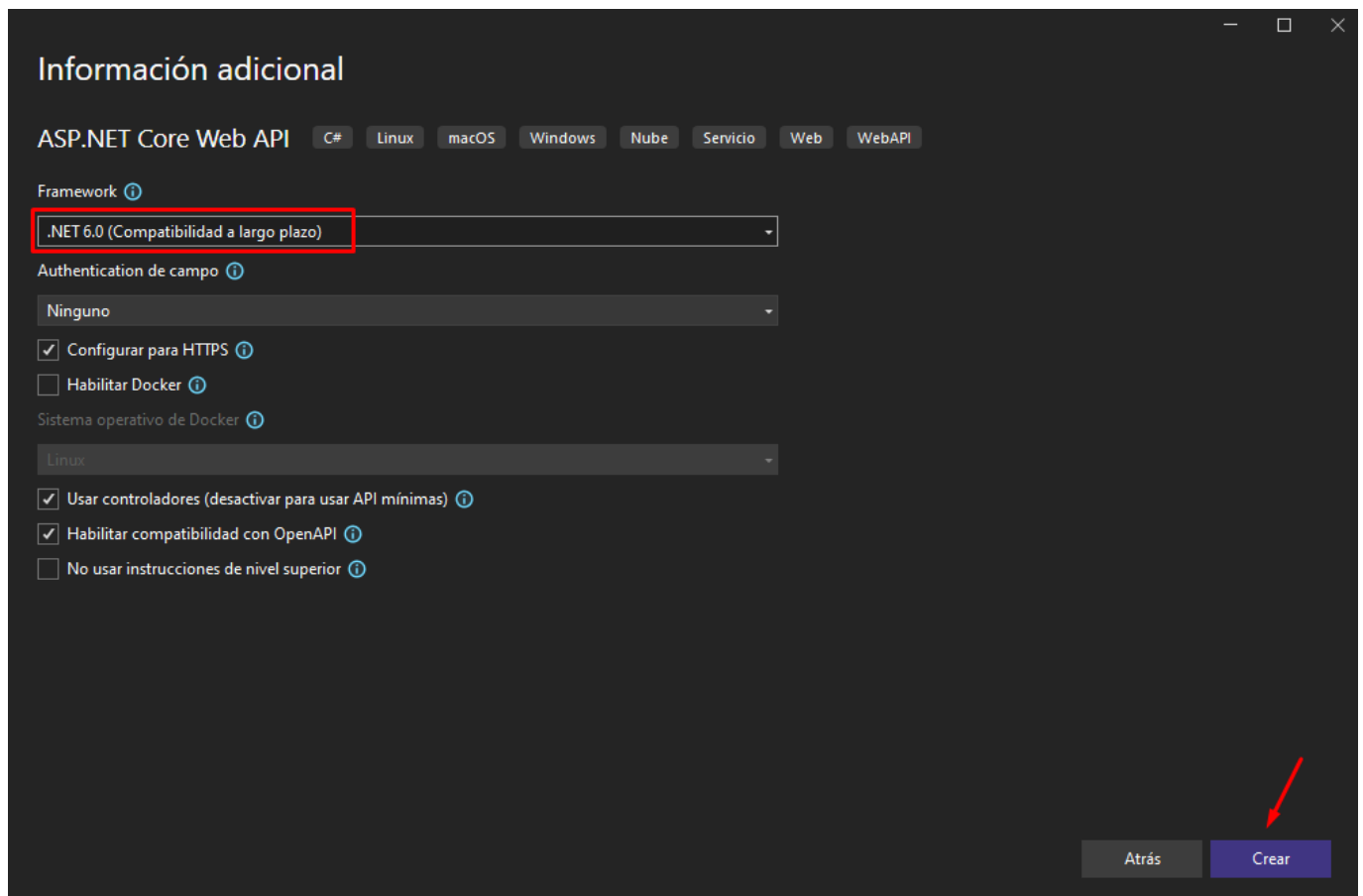
_2. Seleccionar «C#» y tipo «webapi», debemos seleccionar el tipo «**ASP.NET Core Web API**» y le damos siguiente



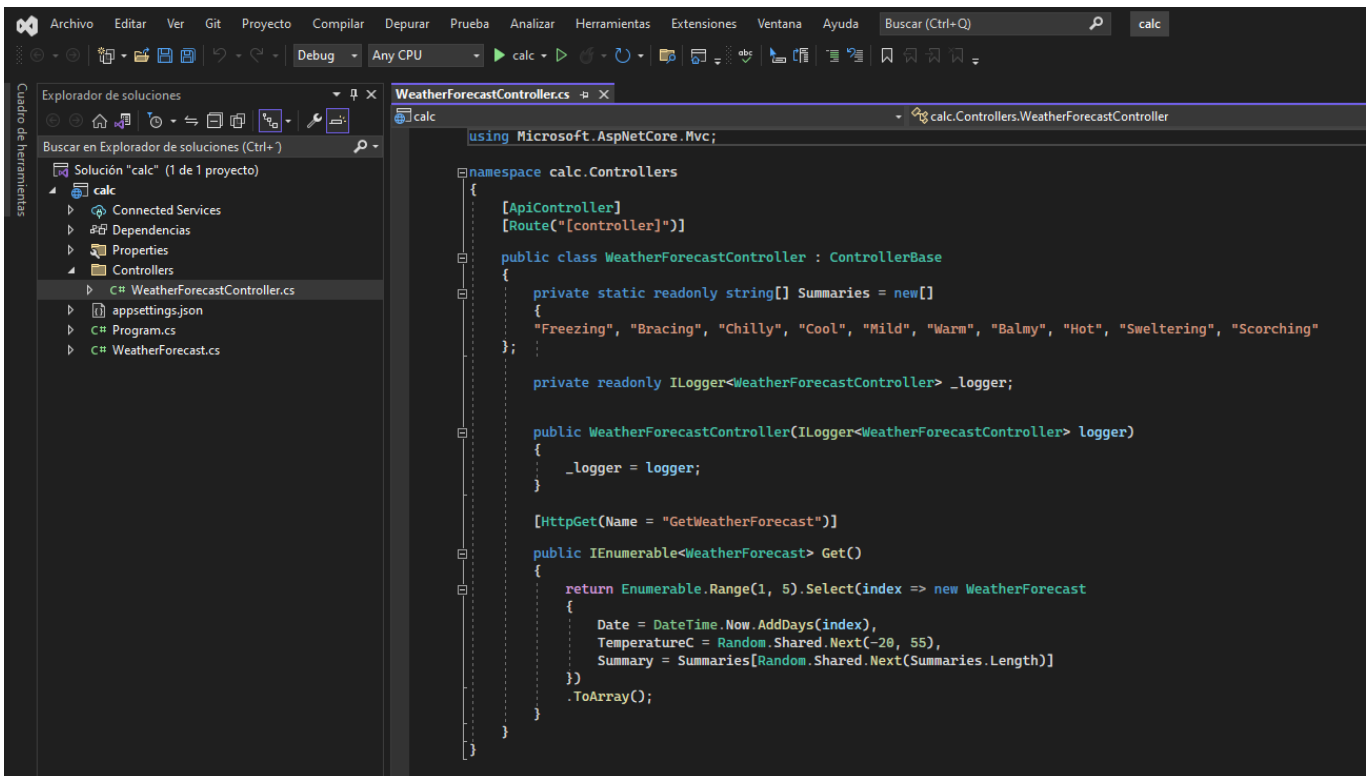
_3. Indicar el nombre del proyecto, en este caso «calc», luego clic en «Siguiente»



_4. Seleccionar el framework «.Net 6.0», luego clic en «Crear»



_5. Se apertura el IDE Visual Studio 2022 con nuestro proyecto, ya podemos ejecutarlo, presionar «>» ícono de iniciar.



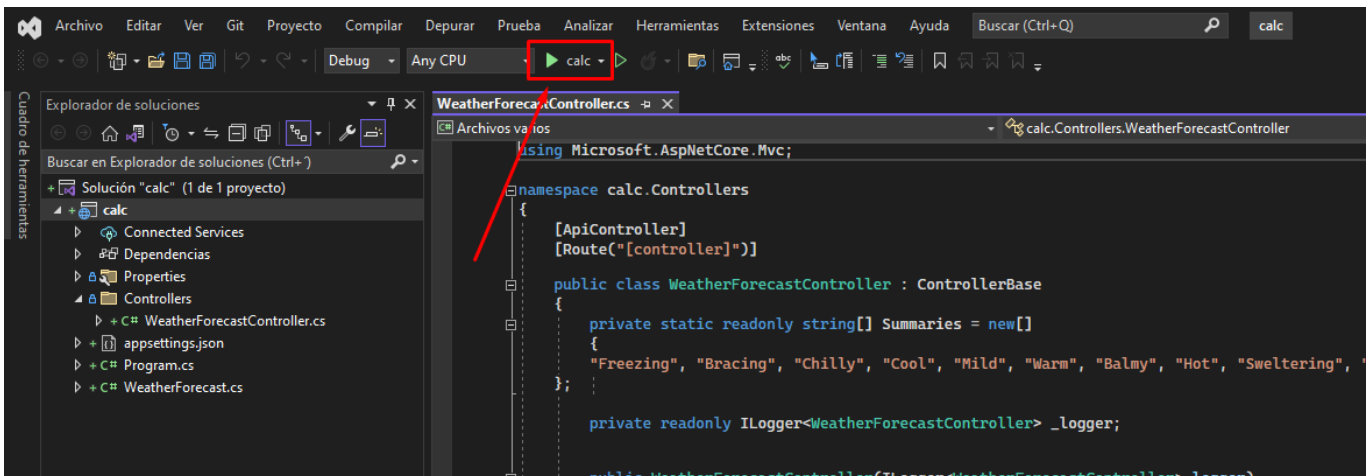
```
using Microsoft.AspNetCore.Mvc;

namespace calc.Controllers
{
    [ApiController]
    [Route("[controller]")]
    public class WeatherForecastController : ControllerBase
    {
        private static readonly string[] Summaries = new[]
        {
            "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"
        };

        private readonly ILogger<WeatherForecastController> _logger;

        public WeatherForecastController(ILogger<WeatherForecastController> logger)
        {
            _logger = logger;
        }

        [HttpGet(Name = "GetWeatherForecast")]
        public IEnumerable<WeatherForecast> Get()
        {
            return Enumerable.Range(1, 5).Select(index => new WeatherForecast
            {
                Date = DateTime.Now.AddDays(index),
                TemperatureC = Random.Shared.Next(-20, 55),
                Summary = Summaries[Random.Shared.Next(Summaries.Length)]
            })
            .ToArray();
        }
    }
}
```



```
using Microsoft.AspNetCore.Mvc;

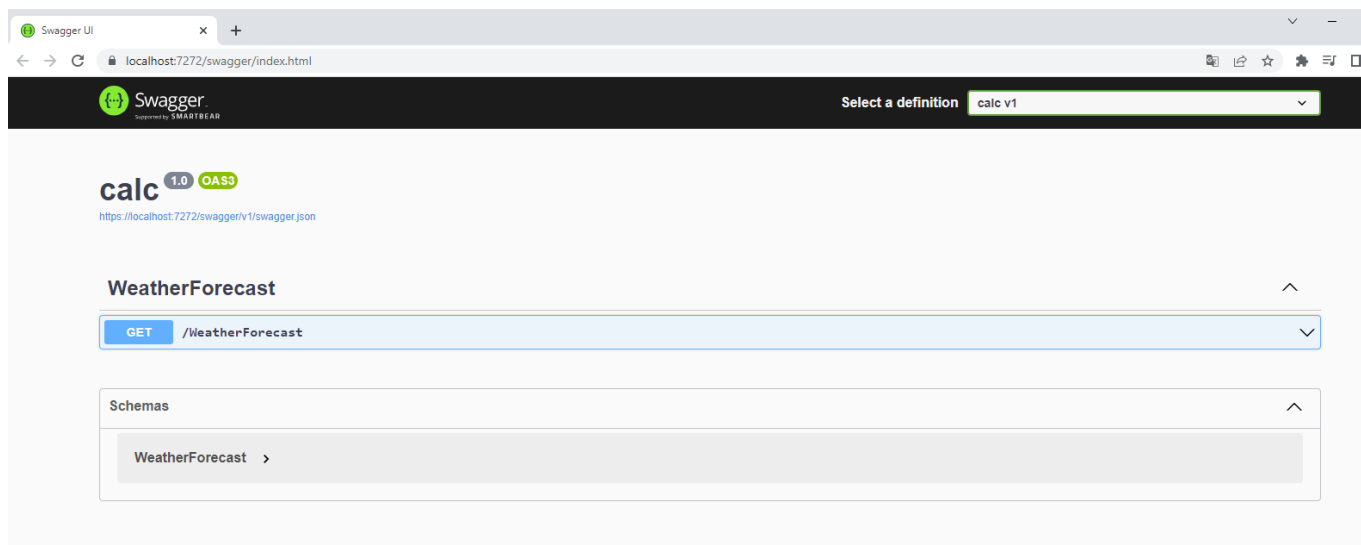
namespace calc.Controllers
{
    [ApiController]
    [Route("[controller]")]
    public class WeatherForecastController : ControllerBase
    {
        private static readonly string[] Summaries = new[]
        {
            "Freezing", "Bracing", "Chilly", "Cool", "Mild", "Warm", "Balmy", "Hot", "Sweltering", "Scorching"
        };

        private readonly ILogger<WeatherForecastController> _logger;

        public WeatherForecastController(ILogger<WeatherForecastController> logger)
        {
            _logger = logger;
        }

        [HttpGet(Name = "GetWeatherForecast")]
        public IEnumerable<WeatherForecast> Get()
        {
            return Enumerable.Range(1, 5).Select(index => new WeatherForecast
            {
                Date = DateTime.Now.AddDays(index),
                TemperatureC = Random.Shared.Next(-20, 55),
                Summary = Summaries[Random.Shared.Next(Summaries.Length)]
            })
            .ToArray();
        }
    }
}
```

_6. Se abrirá la interfaz de Swagger, podremos visualizar las APIs implementadas, podemos probar desde aquí o desde un postman



Julio Pari (IT Architect IBM)



Si te ha interesado este artículo y deseas un apoyo o asesoría en algún requerimiento, envíame un mensaje a: (info@juliopari.com) o sino a través de LinkedIn:

<https://www.linkedin.com/in/juliopari/>