



Dėstytojas
Robertas Ūselis

.NET API Projektas

Data



Šiandien išmoksime

01

Kas yra API?

02

REST API

03

Projekto struktūra

04

HTTP užklausos



Prerequisites

Postman (that's it)



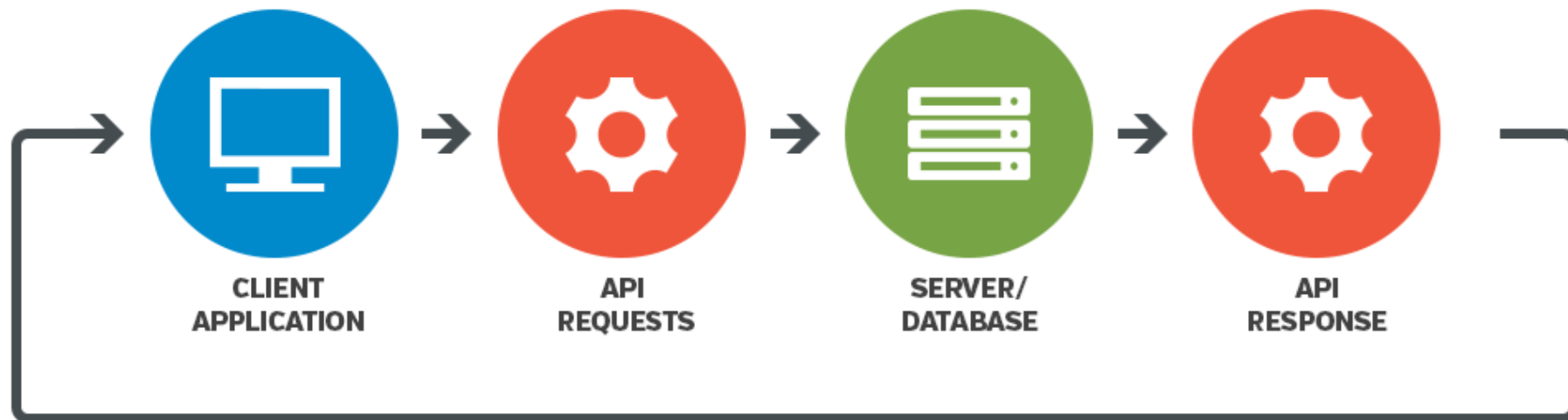
Kas yra API?

APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols. For example, the weather bureau's software system contains daily weather data. The weather app on your phone "talks" to this system via APIs and shows you daily weather updates on your phone.



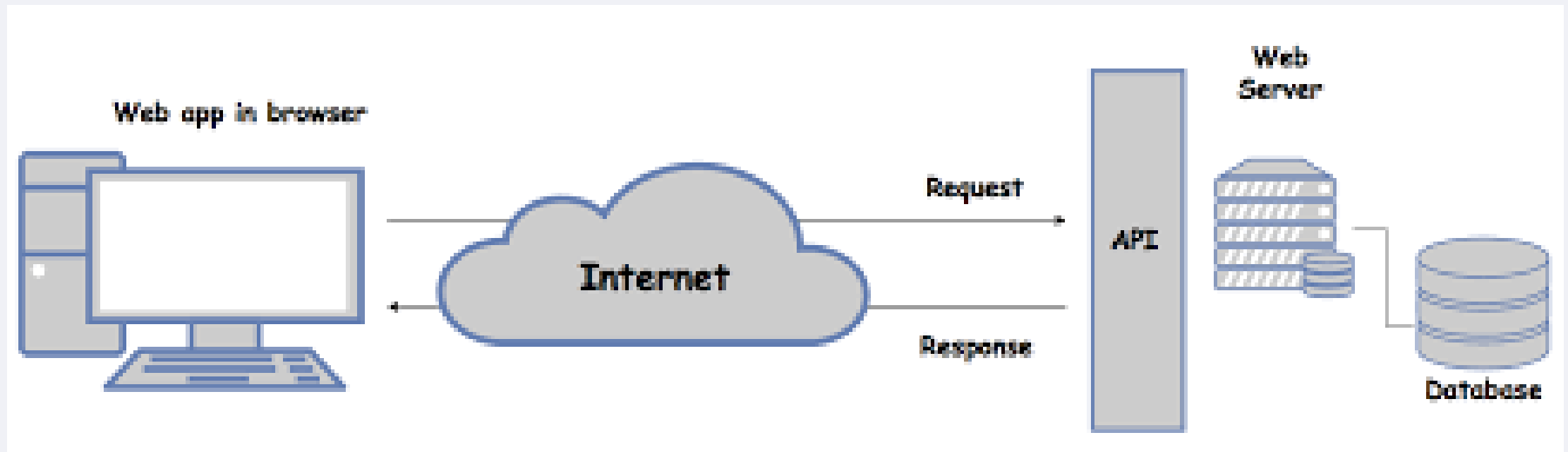
Kas yra API?

HOW AN API WORKS



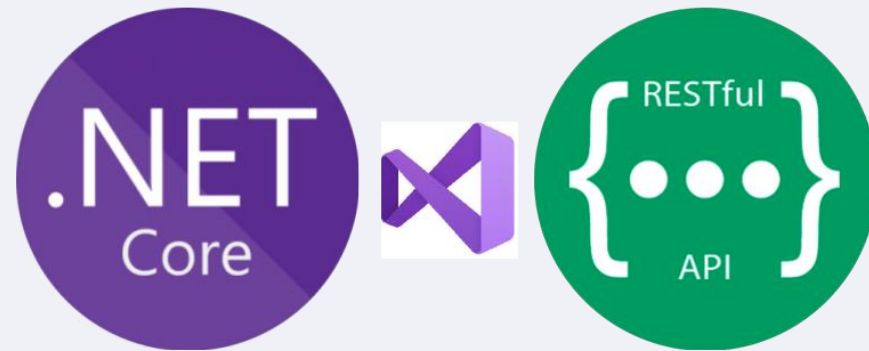


Kas yra API?





Kas yra REST API?



Create



Read



Update



Delete



Kas yra HTTP Route?

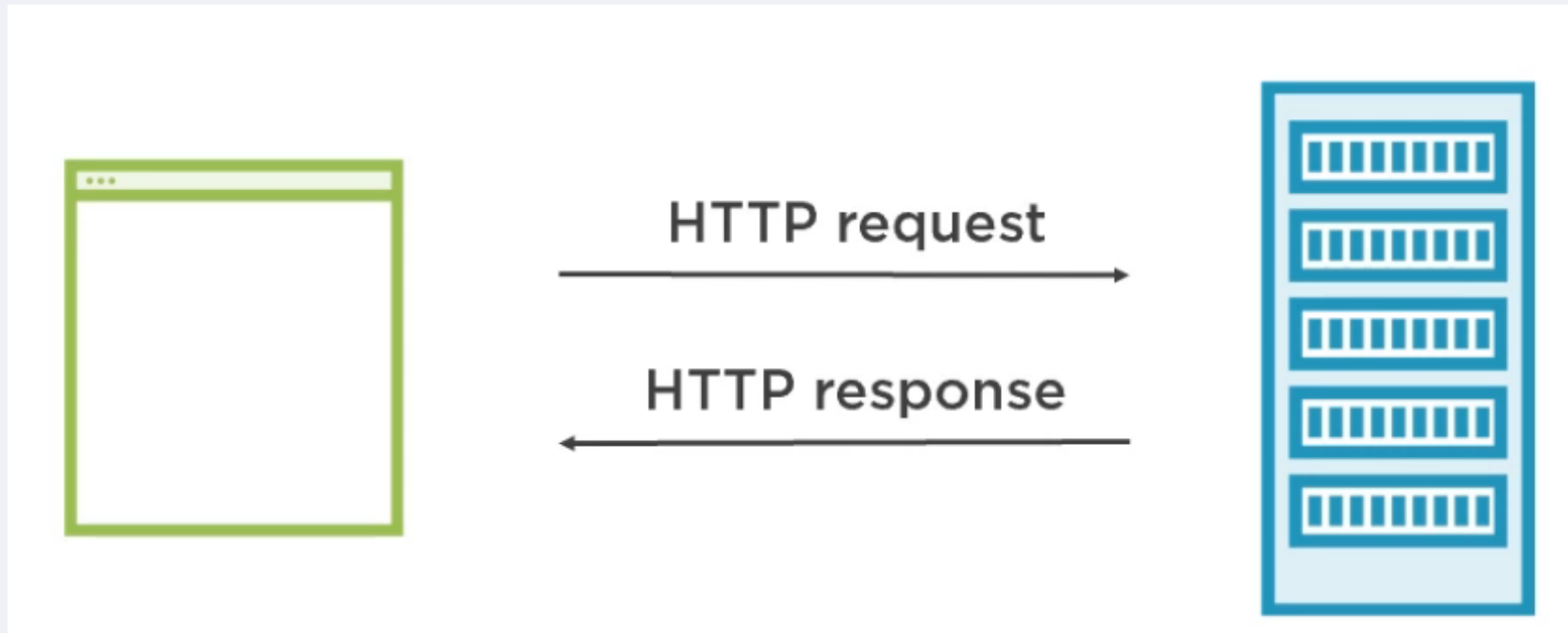
```
foo://example.com:8042/over/there?name=ferret#nose
```

Diagram illustrating the components of a URI (Uniform Resource Identifier):

- scheme
- authority
- path
- query
- fragment



Kas yra HTTP komunikacija?





Kas yra HTTP Header?



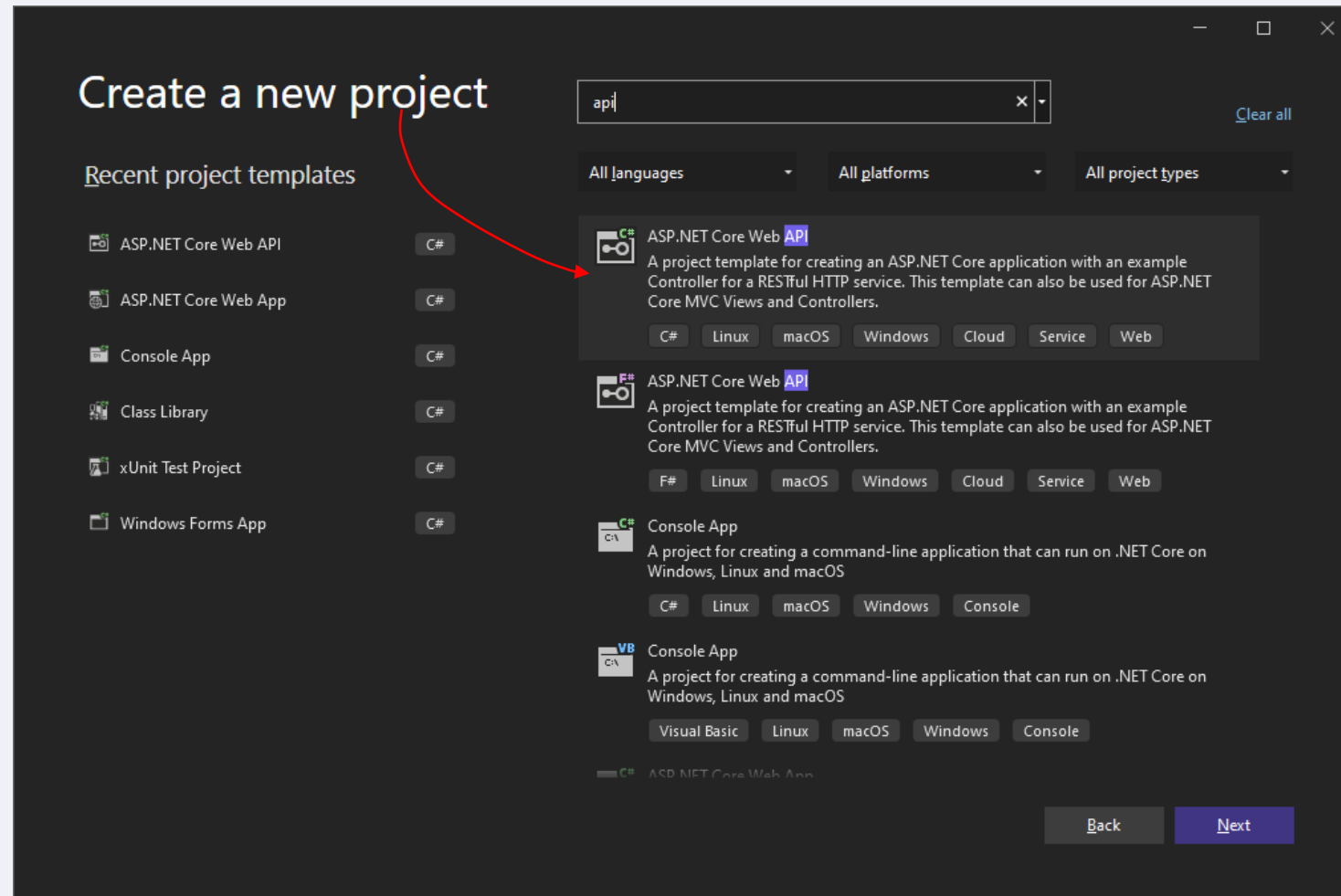
```
▼ Request Headers (771 B) Raw
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Cookie: CONSENT=YES+shp.gws-20210913-0-RC2.It+FX+279; NID=511=l2eiU_GJ2QYAPftaaw
lIe2U59cH7s24hz0xut_JMS8AFX5qq4L-twmJSuvWyuqLkZVoywZX-3ToXPkt2OyBGOq8-aen9JTX
BrixzuXYP1mIQUJHFYzXVH4Az4XrBurRm9MDYhOEZ9to6JSA6wCgPjZ8qyEffY9SUC2TZ-mj_kA; 1P
_JAR=2021-09-20-13; ANID=AHWqTUnuCGYuADCnGamfe0dzbPB0dyzFY8A5qzbYVZfYI4ZU9uy
5JEblqfAKxSvK
Host: www.google.lt
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
TE: trailers
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:92.0) Gecko/20100101 Firefox/92.0
```



```
▼ Response Headers (986 B) Raw
alt-svc: h3=":443"; ma=2592000,h3-29=":443"; ma=2592000,h3-T051=":443"; ma=2592000,h3-
Q050=":443"; ma=2592000,h3-Q046=":443"; ma=2592000,h3-Q043=":443"; ma=2592000,quic
=":443"; ma=2592000; v="46,43"
cache-control: private, max-age=0
content-encoding: br
content-length: 33302
content-type: text/html; charset=UTF-8
date: Mon, 20 Sep 2021 13:25:42 GMT
expires: -1
p3p: CP="This is not a P3P policy! See g.co/p3phelp for more info."
server: gws
set-cookie: 1P_JAR=2021-09-20-13; expires=Wed, 20-Oct-2021 13:25:42 GMT; path=/; domai
n=.google.lt; Secure; SameSite=none
set-cookie: NID=511=hoBx61wTbhSt9BNBQaNVjnCh1dFJImDkVfeDIGKnuanRRf_NtEMYHO3
WboT87-IATdXwTQ_n6QRRheWman3W1DlfzLWfeFHWxkLJq6XnO3P2nYHljUBlkO4VNvBdXOo
bTn5tv24NTXT3ZSLUgv2-QpH4j5rFvdZbvo2exedZzTk; expires=Tue, 22-Mar-2022 13:25:42 GM
T; path=/; domain=.google.lt; Secure; HttpOnly; SameSite=none
strict-transport-security: max-age=31536000
X-Firefox-Spdy: h2
x-frame-options: SAMEORIGIN
x-xss-protection: 0
```

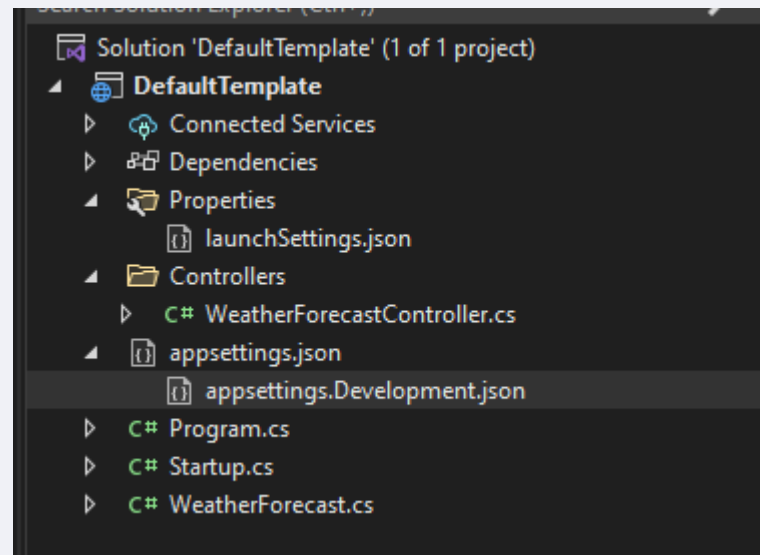


Susikuriame API projektą





Template'o struktūra





WeatherForecastController

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



appsettings.json

```
1  {
2    "Logging": {
3      "LogLevel": {
4        "Default": "Information",
5        "Microsoft": "Warning",
6        "Microsoft.Hosting.Lifetime": "Information"
7      }
8    }
9  }
10
```



Program.cs

```
namespace DefaultTemplate
{
    public class Program
    {
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                });
    }
}
```

Startup.cs (nebijokit)



```
public class Startup
{
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    // This method gets called by the runtime. Use this method to add services to the container.
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddControllers();
        services.AddSwaggerGen(c =>
        {
            c.SwaggerDoc("v1", new OpenApiInfo { Title = "DefaultTemplate", Version = "v1" });
        });
    }

    // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
            app.UseSwagger();
            app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "DefaultTemplate v1"));
        }

        app.UseHttpsRedirection();

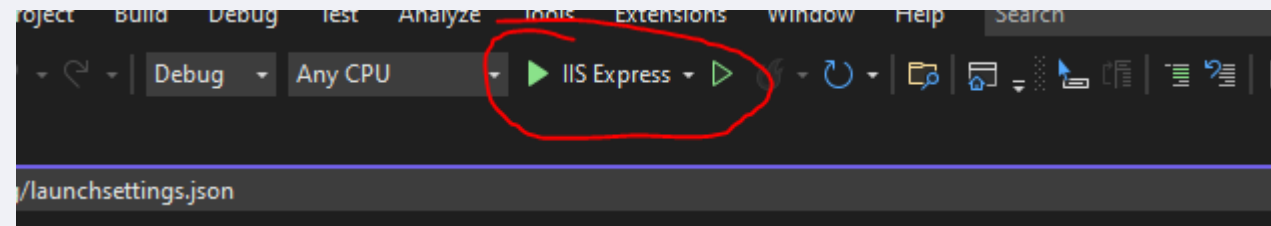
        app.UseRouting();

        app.UseAuthorization();

        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllers();
        });
    }
}
```

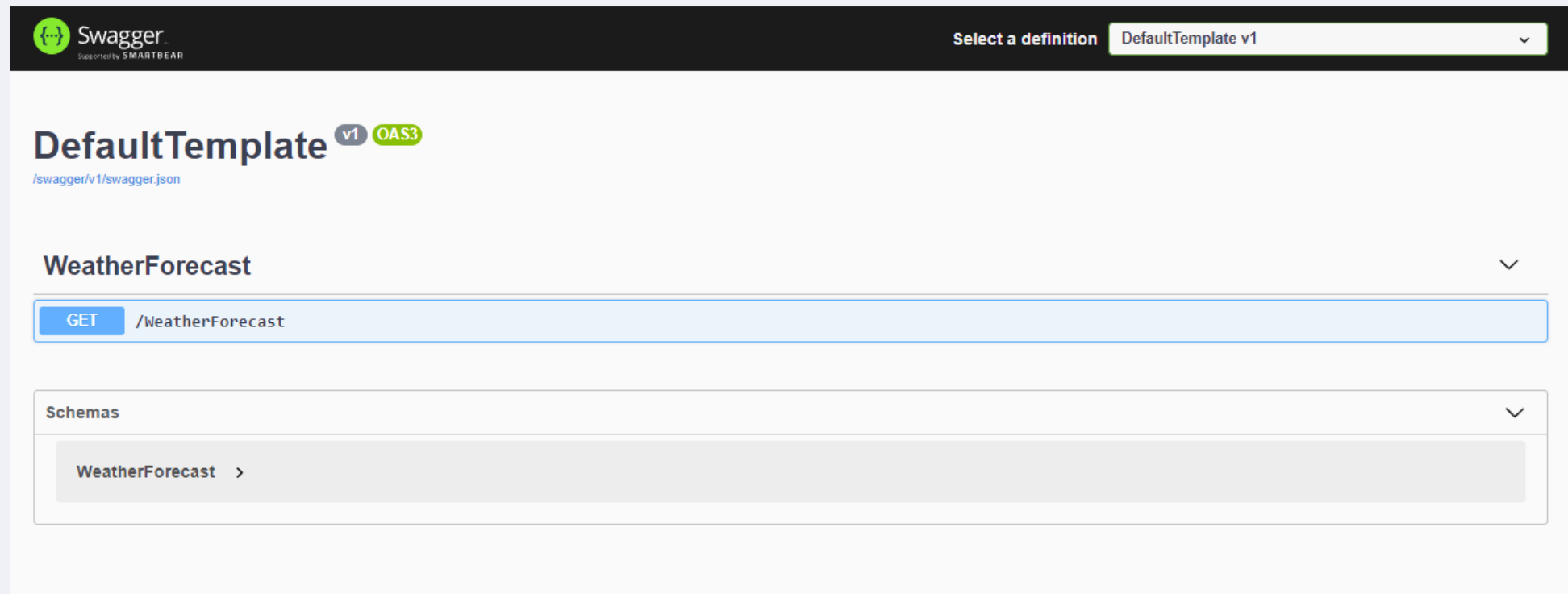



Pasileidžiame projektą





Swashbuckle nuget'o dėka gauname tokį vaizdą





Tą patį galime pamatyti ir su Postman'u

https://localhost:44367/WeatherForecast

GET https://localhost:44367/WeatherForecast

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

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "date": "2022-06-01T17:06:45.3290431+03:00",
4     "temperatureC": 7,
5     "temperatureF": 44,
6     "summary": "Mild"
7   },
8   {
9     "date": "2022-06-02T17:06:45.3294391+03:00",
10    "temperatureC": -2,
11    "temperatureF": 29,
12    "summary": "Bracing"
13  },
14  ]
```



Kas yra REST servisas?

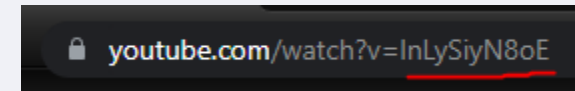
Terminai:

1. Client - klientas yra žmogus/programa naudojanti mūsų API. Klientas išsiunčia **HTTP** užklausą norėdamas gauti, išsaugoti ar atnaujinti informaciją.
2. Resursas - tai informacija, kurią API gali suteikti klientui, Facebook'o resursai būtų klientai, nuotraukos ir t.t.
3. Server - serveris yra vieta į kurią kreipiasi API gauti/atnaujinti resursus klientui.

Daugiau iš teorinės pusės: <https://www.redhat.com/en/topics/api/what-is-a-rest-api>



Kas yra HTTP užklausos?



Pagrindinės HTTP užklausos:

1. GET - Skirtas gauti duomenis iš serverio, norint patikslinti užklausą gali “atsinešti” su savimi parametą per **url**
2. POST - Skirtas išsaugoti naujus duomenis serveryje. Didžiausias skirtumas , kad duomenys nešasi savo kūne(body) ir naudojant SSL apsaugas šis būna užšifruotas, todėl informacija yra daug saugiau gabenama.
3. PUT - Skirtas atnaujinti duomenis. Technologiškai jo struktūra yra tokia pati kaip POST, bet pagal REST principus PUT užklausa turi būti **idempotent** - tai reiškia jog pasiuntus užklausą daugiau negu vieną kartą rezultatas po pirmojo karto neturi keistis.
4. Delete - Skirtas ištrinti įrašą, galima naudoti body, bet nepatariama.

Taip pat kiekviena HTTP užklausa su savimi turi ir Header sekciją



Kas yra Controller?

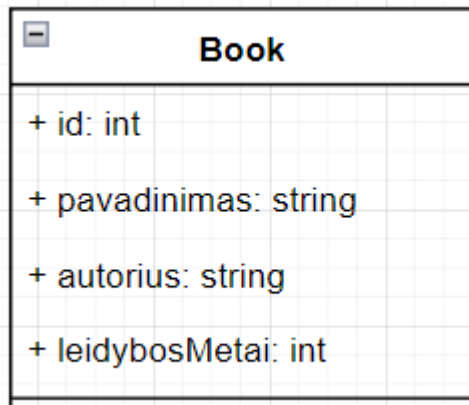
Controller'is yra klasė, kurioje apsirašome **endpoint'us** į kuriuos ateis **HTTP** užklausos.

Pvz WeatherForecast Get() controller'is su endpointu 'https://localhost:44367/WeatherForecast'



Užduotis

- Sukurkite paprastą web puslapį kuris siųs GET užklausas objektus į jūsų sukurtą naują Controller'į
- Controlleris turi gražinti sukurtų objektų (Book) List'ą ir operuos pagal gautos užklausos tipą
- Sąrašas turi turėti ne mažiau kaip 10 knygų. Bent 3 knygos turi būti to paties autoriaus





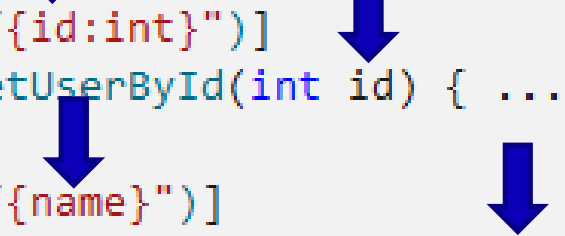
Kas yra Route parameters and constraints?

C#

 Copy

```
[Route("users/{id:int}")]
public User GetById(int id) { ... }

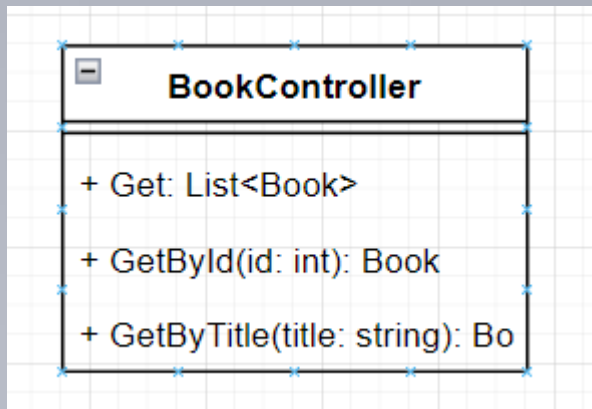
[Route("users/{name}")]
public User GetByName(string name) { ... }
```





Užduotis

- Sukurkite paprastą web puslapį kuris siųs GET užklausas Controller'į
- Controlleris turi grąžinti sukurtą objektą Book
- Turi būti metodai kurie grąžina knygą pagal Id ir pagal pavadinimą





Kas yra Query parameters?

,

```
[HttpGet]
```

```
0 references
```

```
public Character Get(string name, string surname)
{
```



Užduotis

- Sukurkite paprastą web puslapį kuris siųs GET užklausas Controller'į
- Controlleris turi gražinti sukurtų objektų Book list'ą
- Turi būti metodas kuris įvedus visą pavadinimą arba jo dalį gražina knygų listą arba įvedus autorių gražina (būtinai visą tiksliai) to autoriaus knygas
- Neįvedus nieko, gražinama klaida

