A photograph of a modern building with a glass facade, reflecting the sky and surrounding environment. The building is situated next to a paved road with a yellow curb and some greenery. The sky is blue with some clouds.

# **Semana 5** **NET Core HTTP y Enrutamiento** **(Continuación)**

# Agenda



- ✓ **Reflexión** sobre semana 4
- ✓ Map Get y Map Post
- ✓ Parametros
- ✓ Restricciones de enrutamiento
- ✓ Web root y archivos estáticos





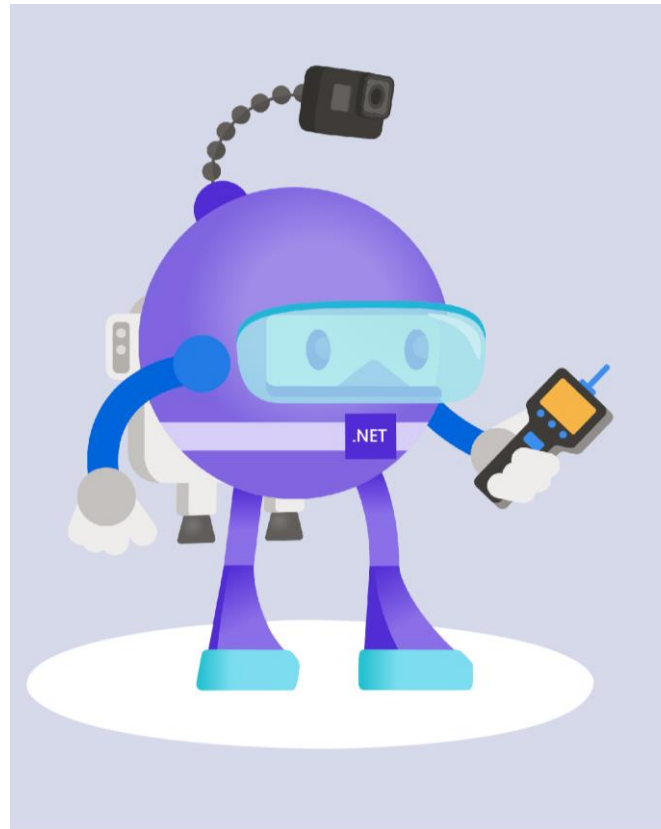
# Espacio de foro con estudiantes



En este espacio discuta con sus estudiantes sobre los conocimientos adquiridos en **HTTP y enrutamiento**.

# Minimal API

ASP.NET Core Minimal APIs proporciona una forma sencilla y elegante de crear endpoints para nuestra aplicación web sin la necesidad de usar controladores tradicionales



*MapGet*

*MapPost*

# Map GET

**MapGet** se usa para definir un endpoint que responde a solicitudes HTTP GET. Las solicitudes **GET** se utilizan comúnmente para recuperar datos o recursos del servidor.

## Ejemplo de código

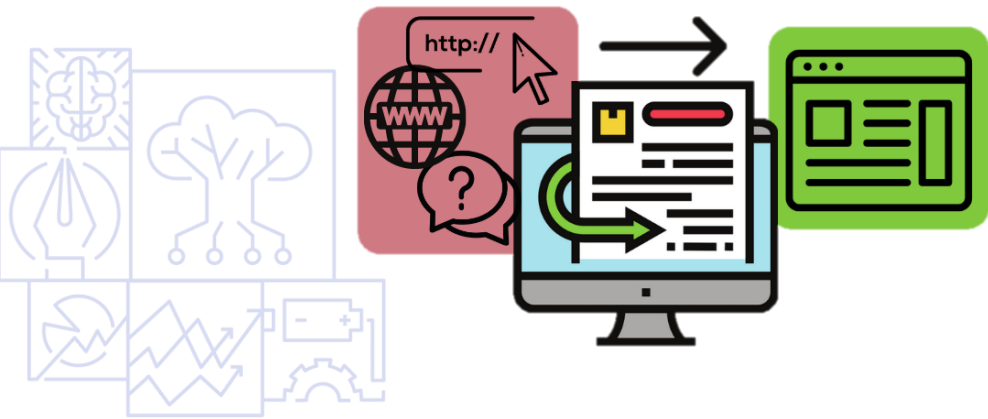
```
namespace MinimalApiScrutor.Features.Users;
0 referencias
public class GetUsers : IEndpoint
{
    0 referencias
    public void MapEndpoint(IEndpointRouteBuilder endpoints)
    {
        endpoints.MapGet("/api/users", () => new List<User>
        {
            new User { Id = Guid.NewGuid(), Name = "User 1", Email = "user@mail.com" },
            new User { Id = Guid.NewGuid(), Name = "User 2", Email = "user2@mail.com" },
        });
    }
}
```



# Map Post

El método **MapPost** se usa para definir un endpoint que responde a solicitudes HTTP POST.

Las solicitudes POST se suelen usar para enviar datos al servidor, como cuando se envían formularios o se crea un nuevo recurso.



```
0 referencias
public class CreateUser : IEndpoint
{
    0 referencias
    public void MapEndpoint(IEndpointRouteBuilder endpoints)
    {
        endpoints.MapPost("/api/users", (CreateUserRequest request) =>
        {
            var user = new User
            {
                Id = Guid.NewGuid(),
                Name = request.Name,
                Email = request.Email,
                Password = request.Password
            };

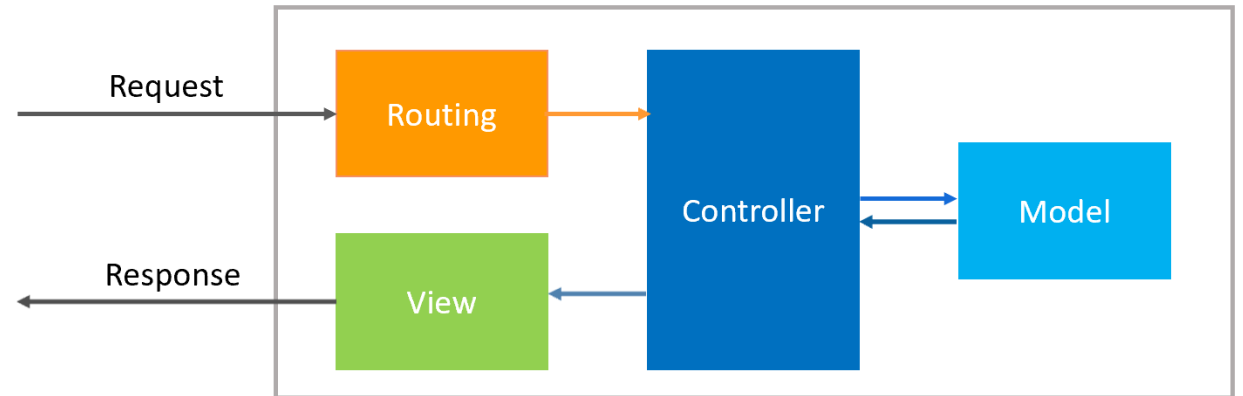
            // TODO: Save user to database

            return Results.Ok(new CreateUserResponse
            {
                UserId = user.Id
            });
        });
    }
}
```



# Enrutamiento o Routing

El *routing* es la forma en la que **Asp.Net Core** decide qué código ejecutar cuando recibe una petición



```
app.MapGet("/beers", () => { /* ... */ });
app.MapPost("/beers", () => { /* ... */ });
app.MapPut("/beers/{id}", (int id) => { /* ... */ });
app.MapPatch("/beers/{id}", (int id) => { /* ... */ });
app.MapDelete("/beers/{id}", (int id) => { /* ... */ });
```

# Parámetros

Los **parámetros** son los valores que una ruta (endpoint) puede recibir cuando se hace una solicitud HTTP. Estos parámetros pueden ser de diferentes tipos y sirven para personalizar la respuesta de la API o para realizar operaciones con ellos, como filtrar datos, realizar cálculos, o especificar condiciones.

```
{
  "FirstName"      : "Sam",
  "LastName"       : "Jackson",
  "employeeID"     : 5698523,
  "Designation"    : "Manager",
  "LanguageExpertise" : ["Java", "C#", "Python"]
  "Car"            : " "
}
```

Q <https://www.example.com/widgets?color=blue&sort=newest>

Annotations:

- start of parameters (points to the question mark)
- separator (points to the ampersand)
- key (points to the word "color")
- value (points to the word "blue")





# Restricciones de Enrutamiento

Las **restricciones de enrutamiento** son reglas que se aplican a las rutas (URLs) en aplicaciones web para limitar o validar qué tipos de valores pueden coincidir con los parámetros en una ruta.



```
[Route("users/{id:int:min(1)}")]  
0 referencias  
public User GetUserById(int id)  
{ }
```

```
app.MapGet("{message:regex(^\\d{{3}}-\\d{{2}}-\\d{{4}}$)}",  
( ) => "Inline Regex Constraint Matched");
```

```
app.MapControllerRoute(  
    name: "people",  
    pattern: "people/{ssn}",  
    constraints: new { ssn = "^\\d{3}-\\d{2}-\\d{4}$", },  
    defaults: new { controller = "People", action = "List" });
```

# Web Root

El **web root** es el directorio dentro de tu proyecto que contiene los archivos estáticos que serán accesibles directamente desde la web. Estos archivos incluyen elementos que el navegador puede descargar directamente, como HTML, CSS, imágenes, JavaScript y otros recursos estáticos.




# Practica Programada 01



El docente realiza una práctica y/o explicación magistral utilizando MapGet y MapPost en el ámbito de enrutamiento.

Realice las mejoras respectivas aplicando los conceptos en el ejercicio realizado la semana anterior. (**Autos el campeón**).





**¡Nos vemos la próxima clase!**