Some application types (e.g. [ASP.NET Core Blazor](https://docs.microsoft.com/en-us/aspnet/core/blazor/)) use dependency injection but do not create a service scope that aligns with the desired DbContext lifetime. Even where such an alignment does exist, the application may need to perform multiple units-of-work within this scope. For example, multiple units-of-work within a single HTTP request.

**start.cs**

public void ConfigureServices(IServiceCollection services)

{

var conn = Configuration.GetConnectionString("Conexion");

services.AddDbContext<AppDbContext>(options => options.UseSqlServer(conn));

// services.AddDbContextFactory<AppDbContext>(

// options =>

// options.UseSqlServer(@"Server=(localdb)\mssqllocaldb;Database=Test"));

services.AddControllers();

}

**appsettings.cs**

"AllowedHosts": "\*",

"ConnectionStrings": {

"Conexion": "data source=(localdb)\\MeLi;Initial Catalog=MeLi;Integrated Security=True;"

}

}

**AppDbContext**

public class AppDbContext : DbContext

{

public AppDbContext(DbContextOptions<AppDbContext> options)

: base(options)

{

}

}

**MyController**

Option 1

private readonly IDbContextFactory<AppDbContext> \_contextFactory;

public LOBController(IDbContextFactory<AppDbContext> contextFactory)

{

\_contextFactory = contextFactory;

}

[HttpGet]

public IActionResult Get()

{

try

{

using (var context = \_contextFactory.CreateDbContext())

{

var listado = context.LOB\_All2.ToList();

if (listado.Count == 0) return StatusCode(404, "Empty");

return Ok(listado);

}

}

catch (Exception ex)

{

return null;

//return BadRequest(ex.Message);

}

}

Option 2

private readonly AppDbContext context;

//Constructor, del mismo nombre que la clase

public LOBController(AppDbContext \_context)

{

this.context = \_context;

}

[HttpGet]

public IActionResult Get()

{

try

{

var listado = context.LOB\_All2.ToList();

if (listado.Count == 0) return StatusCode(404, "Empty");

return Ok(listado);

}

catch (Exception ex)

{

return BadRequest(ex.Message);

}

}