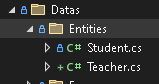
Backend

* [Bevezetés a kliens- és szerveroldali webalkalmazások készítésébe](http://webprogramozas.inf.elte.hu/tananyag/wf2/lecke12_lap1.html)

# Tanár adatok elérése a meglévő projektben

## Tanár model

A Datas/Entities mappába létrehozzuk az új modelt, a Teacher osztályt!



* [What Is GUID In C# (c-sharpcorner.com)](https://www.c-sharpcorner.com/article/what-is-guid-in-c-sharp/)
* [Working With Guid in C# - Code Maze (code-maze.com)](https://code-maze.com/csharp-guid-class/)

public class Teacher

{

public Teacher()

{

Id = Guid.NewGuid();

FirstName = string.Empty;

LastName = string.Empty;

BirthsDay = new DateTime();

IsWoomen = false;

IsHeadTeacher = false;

}

public Teacher(string firstName, string lastName, DateTime birthsDay, bool isWoomen, bool isHeadTeacher)

{

Id = Guid.NewGuid();

FirstName = firstName;

LastName = lastName;

BirthsDay = birthsDay;

IsWoomen = isWoomen;

IsHeadTeacher = isHeadTeacher;

}

public Teacher(Guid id, string firstName, string lastName, DateTime birthsDay, bool isWoomen, bool isHeadTeacher)

{

Id = id;

FirstName = firstName;

LastName = lastName;

BirthsDay = birthsDay;

IsWoomen = isWoomen;

IsHeadTeacher = isHeadTeacher;

}

public Guid Id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public DateTime BirthsDay { get; set; }

public bool IsWoomen { get; set; }

public bool IsHeadTeacher { get; set; }

public override string ToString()

{

return $"{LastName} {FirstName}";

}

## Kréta adatbázis kontextus

A KretaContext adatmodel leképező osztályban felvesszük a tanár adatokat tartalmazó leképező halmazt.

* [DbContext in Entity Framework Core (entityframeworktutorial.net)](https://www.entityframeworktutorial.net/efcore/entity-framework-core-dbcontext.aspx)
* [In-memory Database Provider - EF Core | Microsoft Learn](https://learn.microsoft.com/en-us/ef/core/providers/in-memory/?tabs=dotnet-core-cli)
* [Using MySQL With Entity Framework (c-sharpcorner.com)](https://www.c-sharpcorner.com/UploadFile/8af3e0/using-mysql-with-entity-framework/)

public class KretaContext : DbContext

{

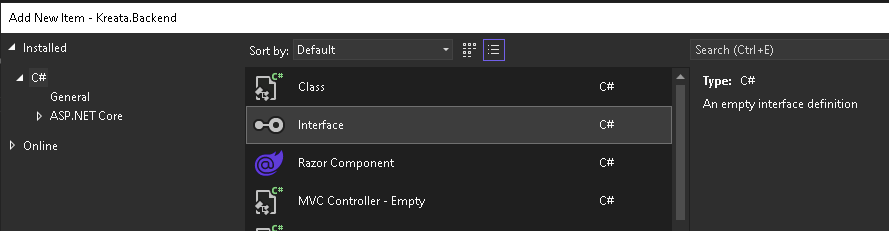
public DbSet<Student> Students { get; set; }

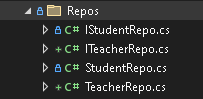
**public DbSet<Teacher> Teachers { get; set; }**

### Repository réteg

* [interface – C# referencia - C# | Microsoft Learn](https://learn.microsoft.com/hu-hu/dotnet/csharp/language-reference/keywords/interface)
* [Interfészek | C# Tutorial.hu (csharptutorial.hu)](https://csharptutorial.hu/docs/hellovilag-hellocsharp/5-osztalyok-objektumok/interfeszek/)
* [C# Interface (w3schools.com)](https://www.w3schools.com/cs/cs_interface.php)

Létrehozzuk a ITeacherRepo interfacet és a TeacherRepo osztályt!





Az ITeacher interface-ben deklaráljuk, hogy a TeacherRepo milyen interfaceket biztosít a külvilág felé. Azaz megadjuk a külső objektumok milyen metódusokon keresztül tudnak kommunikálni a repository réteggel.

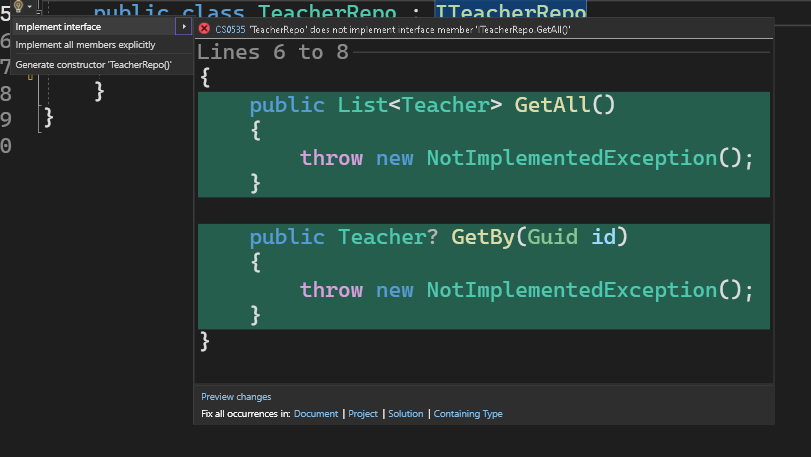
public interface ITeacherRepo

{

Task<List<Teacher>> GetAll();

Task<Teacher?> GetBy(Guid id);

}



A TeacherRepo osztály ettől az interfac-től öröklődik! Implementáljuk az interfacet!

public class TeacherRepo : ITeacherRepo

{

}

Implementálás után:

public class TeacherRepo : ITeacherRepo

{

**public** async Task<List<Teacher>> **GetAll()**

**{**

**throw new NotImplementedException();**

**}**

**public** async Task<Teacher?> **GetBy(Guid id)**

**{**

**throw new NotImplementedException();**

**}**

}

A repository a KretaContext által kezel Set-eken keresztül éri el az adatbázist, ezért injektálunk (dependency injection) ebbe az osztályba egy KretaContext osztályt!

private readonly KretaContext \_dbContext;

public TeacherRepo(KretaContext dbContext)

{

\_dbContext = dbContext;

}

A GetAll metódus az összes tanár adatát adja vissza:

public async Task<List<Teacher>> GetAll()

{

return await \_dbContext.Teachers.ToListAsync();

}

A GetBy metódus az id-vel azonos tanár adatait adja vissza, vagy null értékkel tér vissza, ha nem talál.

public async Task<Teacher?> GetBy(Guid id)

{

return await \_dbContext.Teachers.FirstOrDefaultAsync(s => s.Id == id);

}

## Dependency Injection

A KretaBackendExtensions fájlba a ConfigureRepo metódusba lehetővé tesszük a TeacherRepo osztály injectálását:

public static void ConfigureRepos(this IServiceCollection services)

{

services.AddScoped<IStudentRepo, StudentRepo>();

services.AddScoped<ITeacherRepo, TeacherRepo>();

}

## Controller elkészítése

Vizsgáljuk meg a StudentControllert:

Annotációkat használunk, hogy bizonyos kódokat ne keljen megírni!

* [ASP.NET Core webes API-vezérlők - Training | Microsoft Learn](https://learn.microsoft.com/hu-hu/training/modules/build-web-api-aspnet-core/4-aspnet-controllers)
* [Webes API létrehozása ASP.NET Core vezérlőkkel - Training | Microsoft Learn](https://learn.microsoft.com/hu-hu/training/modules/build-web-api-aspnet-core/)
* [2.1 Az API-okról röviden - YouTube](https://www.youtube.com/watch?v=WfZVWAIqscY)
* [Web API Tutorials .Net Core 7 C# - YouTube](https://www.youtube.com/playlist?list=PL3ewn8T-zRWgO-GAdXjVRh-6thRog6ddg)

[ApiController]

[Route("api/[controller]")]

public class StudentController : ControllerBase

{

private IStudentRepo \_studentRepo;

public StudentController(IStudentRepo studentRepo)

{

\_studentRepo = studentRepo;

}

[HttpGet]

public async Task<IActionResult> SelectAllRecordToListAsync()

{

List<Student>? users = new();

if (\_studentRepo != null)

{

users = await \_studentRepo.GetAll();

return Ok(users);

}

return BadRequest("Az adatok elérhetetlenek!");

}

[HttpGet("{id}")]

public async Task<IActionResult> GetBy(Guid id)

{

Student? entity = new();

if (\_studentRepo is not null)

{

entity = await \_studentRepo.GetBy(id);

if (entity!=null)

return Ok(entity);

}

return BadRequest("Az adatok elérhetetlenek!");

}

}

Készítsük el a Teacher controllert:

[ApiController]

[Route("api/[controller]")]

public class TeacherController : ControllerBase

{

private ITeacherRepo \_teacherRepo;

public TeacherController(ITeacherRepo teacherRepo)

{

\_teacherRepo = teacherRepo;

}

[HttpGet("{id}")]

public async Task<IActionResult> GetBy(Guid id)

{

Teacher? entity = new();

if (\_teacherRepo is not null)

{

entity = await \_teacherRepo.GetBy(id);

if (entity != null)

return Ok(entity);

}

return BadRequest("Az adatok elérhetetlenek!");

}

[HttpGet]

public async Task<IActionResult> SelectAllRecordToListAsync()

{

List<Teacher>? users = new();

if (\_teacherRepo != null)

{

users = await \_teacherRepo.GetAll();

return Ok(users);

}

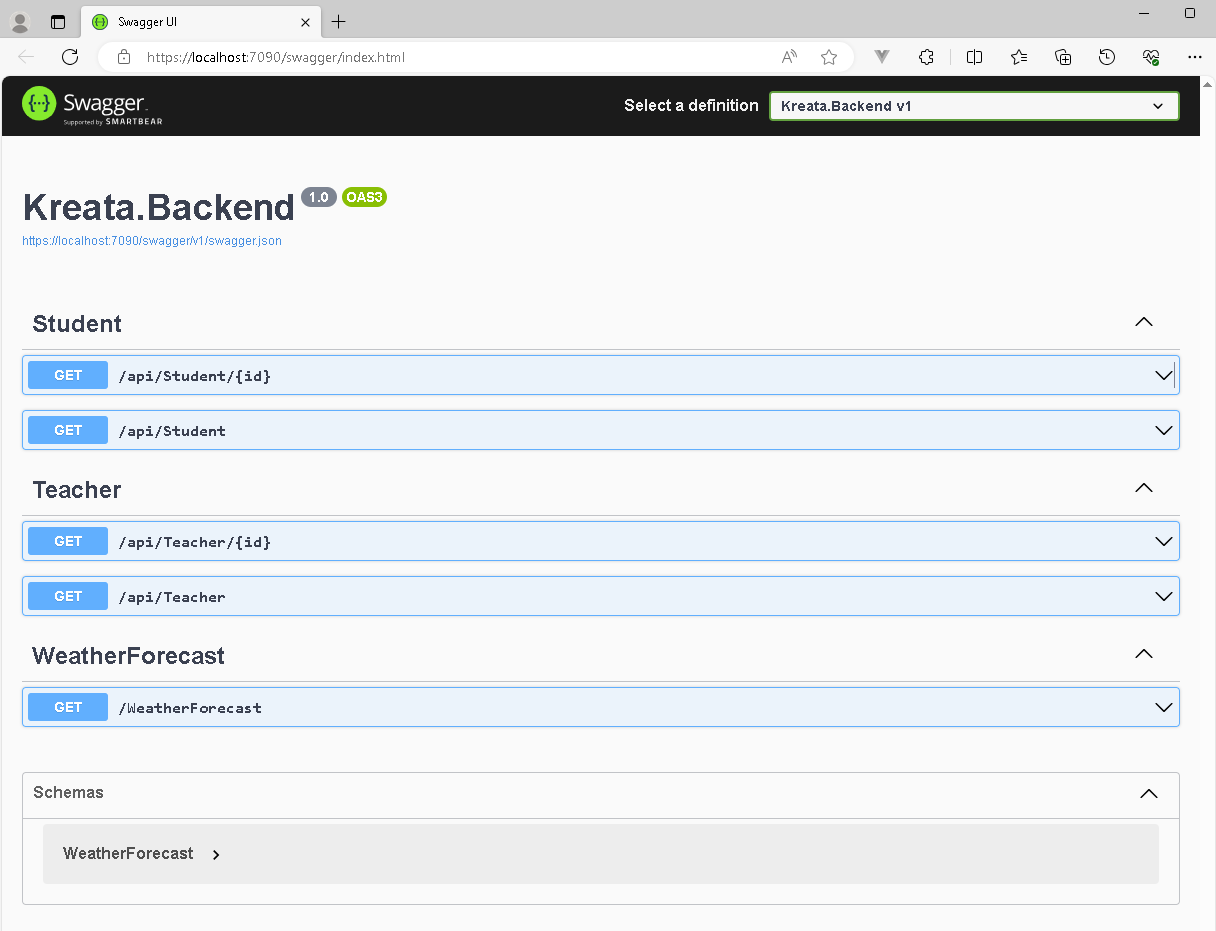
return BadRequest("Az adatok elérhetetlenek!");

}

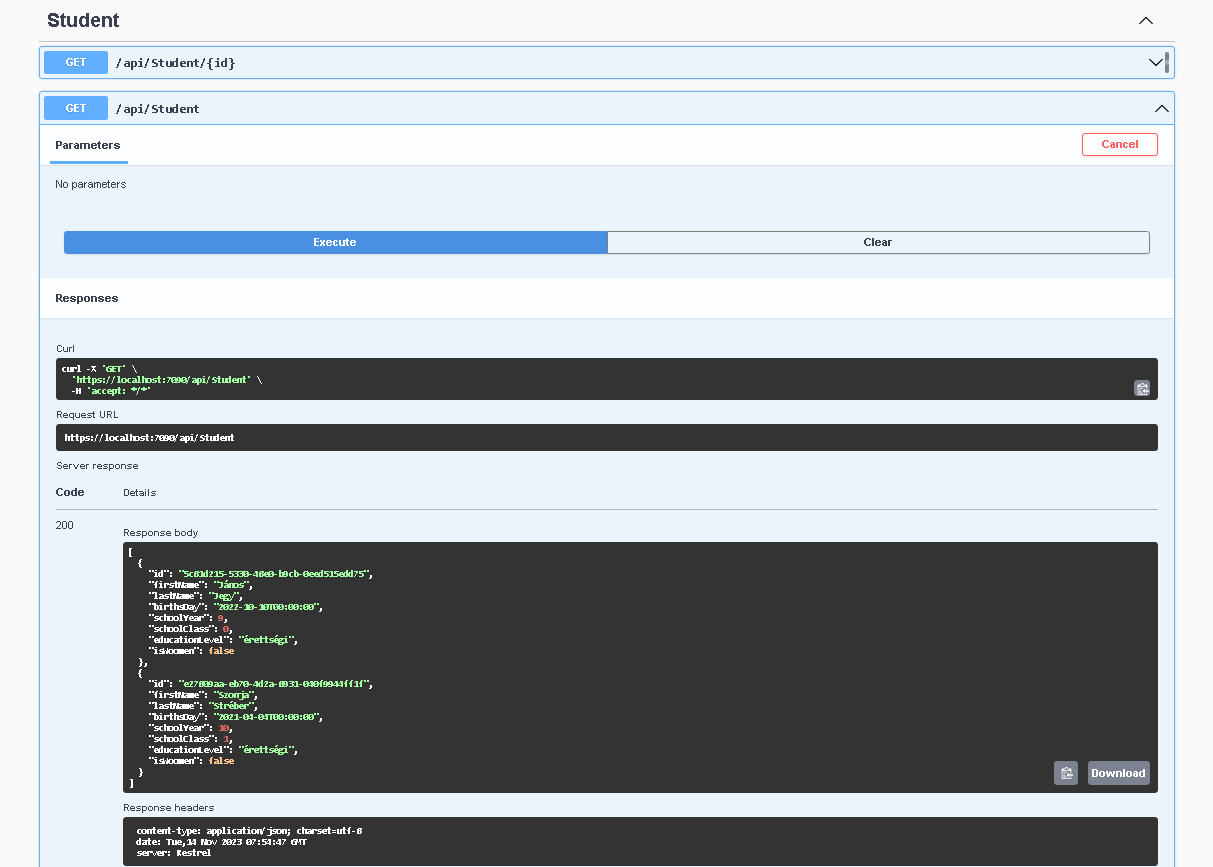
}

## Tesztelés

Backend nyitó képernyő:



Összes tanár lekérése



Az id megadása után egy adott tanár adatainak lekérdezése:

