Szűrés és keresés

- Filtering in ASP.NET Core Web API - Code Maze (code-maze.com)
- Searching in ASP.NET Core Web API - Code Maze (code-maze.com)

Mi az a szűrés? (filtering)

A szűrés egy olyan mechanizmus, amellyel az eredmények valamilyen kritérium megadásával nyerhetők ki. Sokféle szűrőt írhatunk, hogy az eredményeket az osztály tulajdonságának típusa, értéktartomány, dátumtartomány vagy bármi más alapján kapjuk meg.
A szűrés megvalósításakor mindig korlátozza a kérésben megadható előre meghatározott opciók készlete. Például küldhetünk dátumértéket a tulajdonosi fiók típusának lekérdezéséhez, de nem sok sikert fogunk elérni.
Az elülső oldalon a szűrést általában jelölőnégyzetek, rádiógombok vagy legördülő gombok formájában valósítják meg. Ez a fajta megvalósítás csak azokra a lehetőségekre korlátozza Önt, amelyek egy érvényes szűrő létrehozásához rendelkezésre állnak.
Vegyünk például egy autóértékesítő weboldalt. A kívánt autók szűrésekor ideális esetben választani szeretne:

- Autógyártó mint kategória egy listából vagy egy legördülő menüből.
- Autómodell egy listából vagy egy legördülő listából
- Új vagy használt autó rádiógombokkal?
- A város, ahol az eladó található, legördülő menüpontként.
- Az autó ára egy beviteli mező (numerikus)

Mi az a keresés? (searching)

Kétségtelen, hogy szinte minden internetes weboldalon látott már keresőmezőt.
Könnyű megtalálni valamit, ha ismerjük a weboldal szerkezetét, vagy ha egy weboldal nem olyan nagy. De ha a számunkra legmegfelelőbb témát szeretnénk megtalálni, vagy ha nem tudjuk, hogy mit fogunk találni, esetleg először látogatunk meg egy nagyméretű weboldalt, akkor valószínűleg keresőmezőt fogunk használni.
Az alapvető keresést nem olyan nehéz megvalósítani, de ha félvállról vesszük, a keresési funkció használhatatlan lehet.
A keresés egyik felhasználási esete az lenne, hogy egy tulajdonost a neve alapján keressünk meg.

Feladatunk

A feladat megoldása közben a repository mintát fogjuk kiegészíteni újabb rest api elérésekkel:

- Diákok szűrése születési éveik alapján (keressük azon diákokat akik születése éve egy adott intervallumba esik).
- A keresésre a diák teljes nevét fogjuk használni, és megkeressük azon diákokat akiknek neve illeszkedik a beírt mintára.

A feladat megoldása

Előkészületek

A szűrés és keresés paramétereit egy osztályba fogjuk eltárolni, ez most a `StudentQueryParameters` metódus lesz, amelynek elkészítjük a DTO változatát, és a konvertálását az osztály és DTO osztály között.

```
public class StudentQueryParameters
{
    public uint MinYearOfBirth { get; set; }
    public uint MaxYearOfBirth { get; set; } = (uint)DateTime.Now.Year;

    public bool ValidYearRange => MaxYearOfBirth > MinYearOfBirth;

}
```

Dto

```
    public class StudentQueryParametersDto
    {
        public uint MinYearOfBirth { get; set; }
        public uint MaxYearOfBirth { get; set; } = (uint)DateTime.Now.Year;
    }
```

Extension:

```
public static class StudentQueryParametersExtension
{
    public static StudentQueryParametersDto ToDto(this StudentQueryParameters parameters)
    {
        return new StudentQueryParametersDto
        {
            MaxYearOfBirth = parameters.MaxYearOfBirth,
            MinYearOfBirth = parameters.MinYearOfBirth
        };
    }

    public static StudentQueryParameters ToStudentQueryParameters(this StudentQueryParametersDto parameters)
    {
        return new StudentQueryParameters
        {
            MinYearOfBirth = parameters.MinYearOfBirth,
            MaxYearOfBirth = parameters.MaxYearOfBirth
        };
    }
}
```

A szűrést a StudentRepo-ba fogjuk elkészíteni, ezért az IStudentRepo interface amely repository mintától öröklődik egy új művelettel fog kiegészülni:

```
public IQueryable<Student> GetStudents(StudentQueryParameters parameters);
```

A szűrés kódja

A StudentRepoba írjuk meg a szűrés kódját:

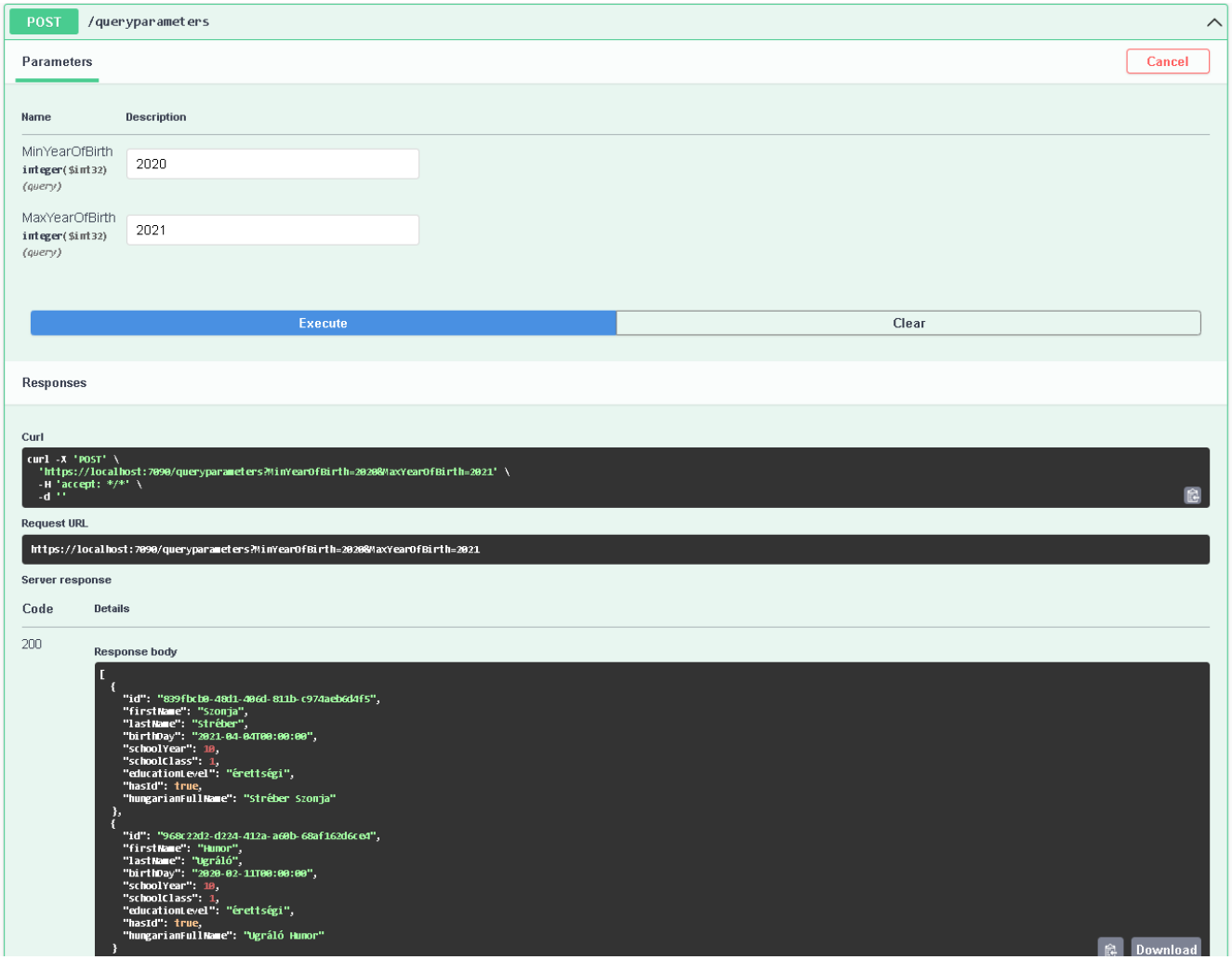
```
        public IQueryable<Student> GetStudents(StudentQueryParameters parameters)
        {
            return FindByCondition(student => student.BirthDay.Year >= parameters.MinYearOfBirth
                                              && student.BirthDay.Year <= parameters.MaxYearOfBirth
                                   )
                                   .OrderBy(student => student.HungarianFullName);
        }
```

A szűrt adatokat teljes név alapján rendezzük!

A RestApi kérés sem lesz bonyolult:

```
[HttpPost("/queryparameters")]
public async Task<IActionResult> GetStudents([FromQuery] StudentQueryParametersDto dto)
{
    StudentQueryParameters parameters = dto.ToStudentQueryParameters();
    if (!parameters.ValidYearRange)
    {
        ControllerResponse response = new ControllerResponse();
        response.AppendNewError("A születési év maximuma nagyobb kell legyen a születési év minimumánál!");
        return BadRequest(response);
    }
    else
    {
        if (_studentRepo is null)
        {
            ControllerResponse response = new ControllerResponse();
            response.AppendNewError("A diákok szűrése születési év alapján nem lehetséges");
            return BadRequest(response);
        }
        else
        {
            List<Student> result = await _studentRepo.GetStudents(parameters).ToListAsync();
            return Ok(result);
        }
    }

}
```

Van folytatás!!!



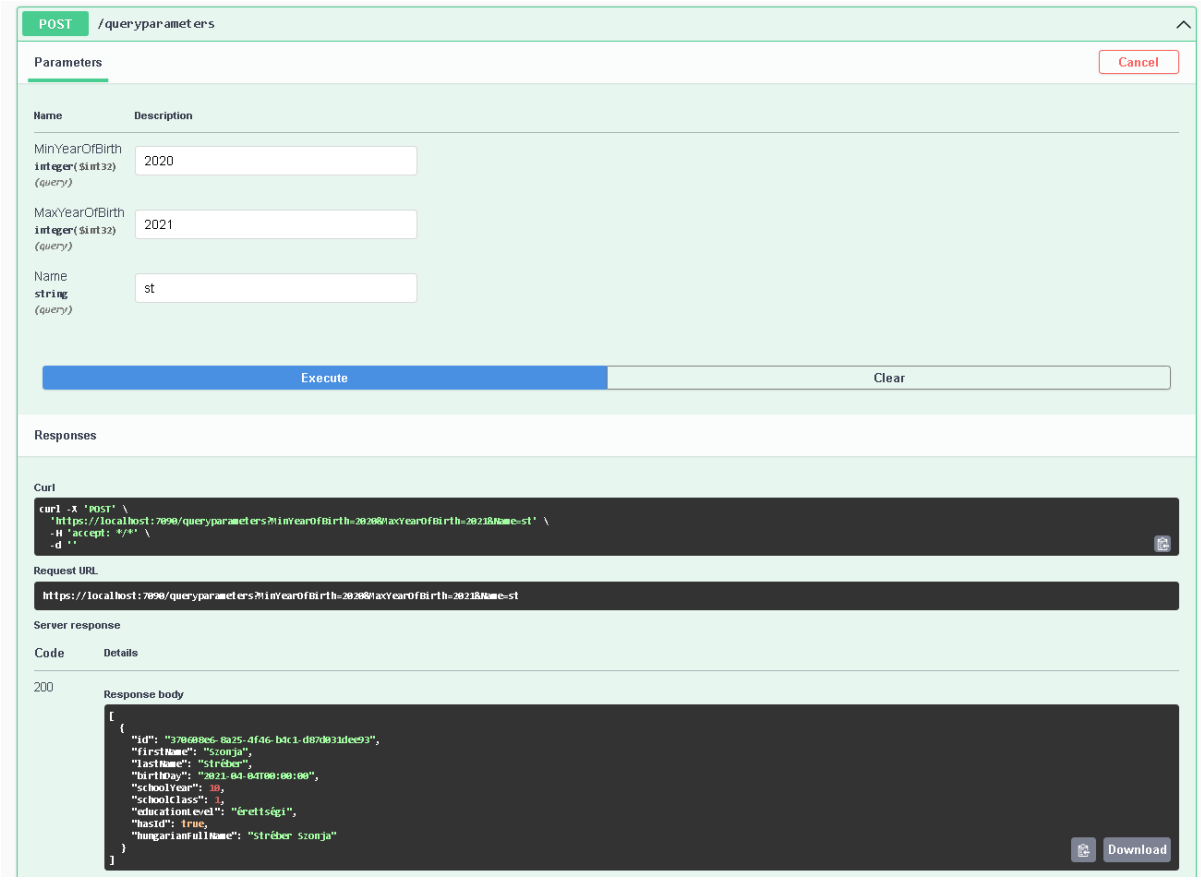
A rendezés `(OrderBy(student => student.HungarianFullName))` akkor működik jól, ha a `HungarianFullNam` változónak van set metódusa is!

```
        public string HungarianFullName
        {
            set { }
            get => $"{LastName} {FirstName}";
        }
```

A keresés kódja

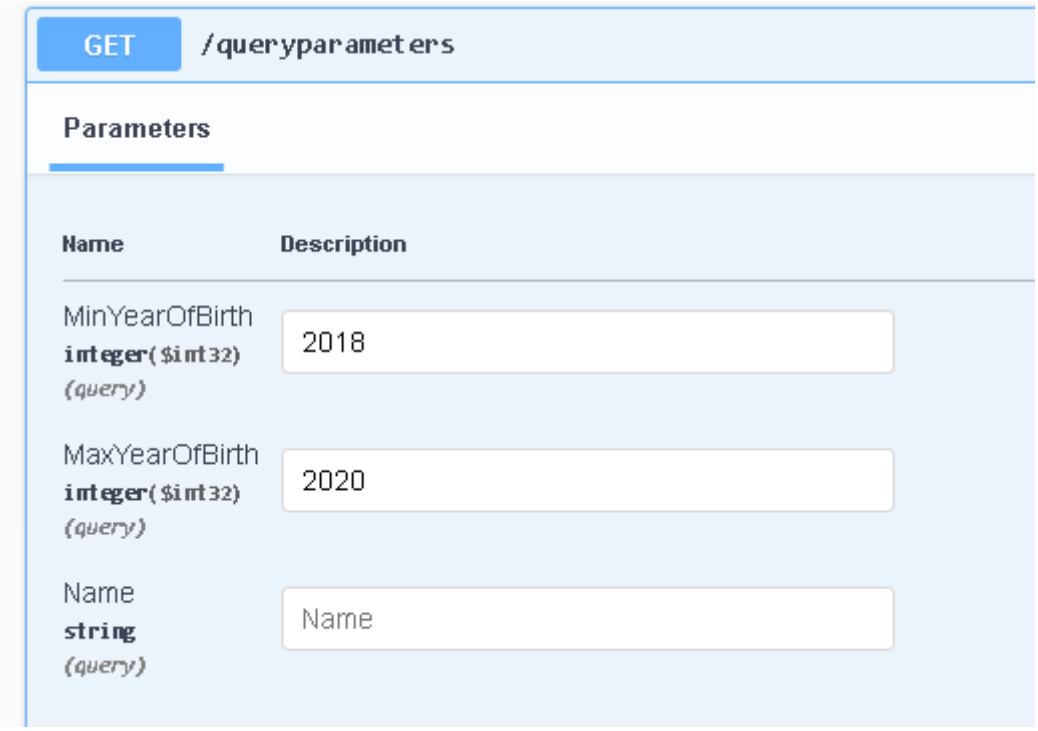
A keresés során keressük, hogy van-e olyan teljes név amelyben a keresett névrészlet található!

```
public IQueryable<Student> GetStudents(StudentQueryParameters parameters)
{
        IQueryable<Student> filteredStudent = FindByCondition(
                  student => student.BirthsDay.Year >= parameters.MinYearOfBirth
                && student.BirthsDay.Year <= parameters.MaxYearOfBirth
                   )
                  .OrderBy(student => student.HungarianFullName);

        SearchByStudentName(ref filteredStudent,parameters.Name);
        return filteredStudent;
}

private void SearchByStudentName(ref IQueryable<Student> students, string studentName)
{
   if (!students.Any() || string.IsNullOrEmpty(studentName))
   {
        return;
   }
   students=students.Where(student => student.HungarianFullName.ToLower().Contains(studentName.Trim().ToLower()));
}
```

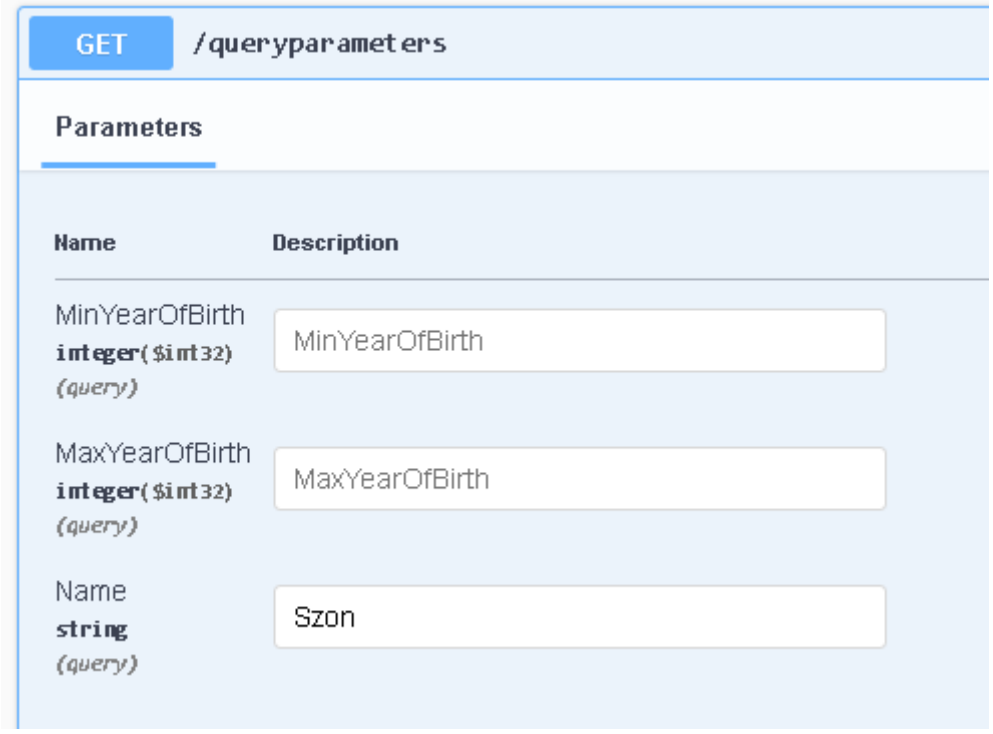


Tesztelés

Szűrés születési évszámokra



Keresés név részletre



Keresés és szűrés