

ASP.NET Active Dynamic Web Pages

יחסי גומלין

עבודה עם ViewModels

אחד לרבים

```
✓ public class SongAlbumsViewModel
{
    4 references
    public Song Song { get; set; } = new Song() { Title = string.Empty };
    2 references
    public List<Album> Albums { get; set; } = [];
}
```

עבודה עם ViewModels

אחד לרבים

```
// GET: Songs/Create
```

0 references

```
public async Task<IActionResult> Create()
{
    //SHOW A Dropdown of albums:
    var albums = await context.Albums.ToListAsync();

    var vm = new SongAlbumsViewModel() { Albums = albums};

    return View(vm);
}
```

עבודה עם ViewModels אחד לרבים

@model SongAlbumsViewModel

1

```
<form asp-action="Create">
  <div asp-validation-summary="ModelOnly" class="text-danger"></div>
  <div class="form-group">
    2 <label asp-for="Song.Title" class="control-label"></label>
      <input asp-for="Song.Title" class="form-control" />
      <span asp-validation-for="Song.Title" class="text-danger"></span>
    </div>

    <div class="form-group">
      <select name="Song.albumId">
        3 <option value="">Single</option>
        4 @foreach (var album in Model.Albums)
          {
            <option value="@album.Id">@album.Name</option>
          }
      </select>
    </div>

    <div class="form-group">
      <input type="submit" value="Create" class="btn btn-primary" />
    </div>
</form>
```

Single - לא שייך לאלבום

עבודה עם ViewModels אחד לרבים

[HttpPost]

[ValidateAntiForgeryToken]

0 references

```
public async Task<IActionResult> Create(SongAlbumsViewModel vm)
{
    if (ModelState.IsValid)
    {
        context.Add(vm.Song);
        await context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(vm);
}
```

עבודה עם ViewModels

אחד לרבים

Details

```
public async Task<IActionResult> Details(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var song = await context.Songs.FindAsync(id);

    if (song == null)
    {
        return NotFound();
    }
    song.Album = await context.Albums.FindAsync(song.AlbumId);
    return View(song);
}
```

עבודה עם ViewModels

אחד לרבים

Details

```
<dl class="row">
  <dt class="col-sm-2">
    @Html.DisplayNameFor(model => model.Title)
  </dt>
  <dd class="col-sm-10">
    @Html.DisplayFor(model => model.Title)
  </dd>

  <dt class="col-sm-2">
    @Html.DisplayNameFor(model => model.Album)
  </dt>
  <dd class="col-sm-10">
    @{
        var albumName = Model.Album?.Name ?? "Single";
    }
    @albumName
  </dd>
</dl>
```

עבודה עם ViewModels

אחד לאחד – ה Scaffold עושה עבודה טובה – אבל צריך שיפור:

צריך להציג רק מדינות שאין להן עיר בירה כדי לשבץ אחד לאחד
(שלא יצא אחד לרבים)

```
// GET: Capitals/Create
```

```
0 references
```

```
public async Task<IActionResult> Create()
```

```
{
```

```
var countries = await context.Countries.Where(c => c.Capital == null).ToListAsync();
```

```
ViewData["CountryId"] = new SelectList(countries, "Id", "Name");
```

```
return View();
```

```
}
```

```
public class Capital
```

```
{
```

```
[Key]
```

```
11 references | 0 exceptions, - live
```

```
public int Id { get; set; }
```

```
[Required, MinLength(1), MaxLength(30)]
```

```
12 references | 0 exceptions, - live
```

```
public required string Name { get; set; }
```

```
//One to One Relationship - Foreign Key:
```

```
8 references | 0 exceptions, - live
```

```
public int CountryId { get; set; }
```

```
//Navigation Properties:
```

```
9 references | 0 exceptions, - live
```

```
public Country? Country { get; set; }
```

נשנה לאופציונלי

עבודה עם ViewModels

רבים לרבים – הצגת סרטים עם זאנר

```
// GET: Movies
```

3 references

```
public async Task<IActionResult> Index()
{
    var Movies = await context.Movies.Include(m => m.Genres).ToListAsync();
    return View(Movies);
}
```

עבודה עם ViewModels

רבים לרבים – הצגת סרטים עם זאנר

Views/Movie/Index.cshtml

```
@model IEnumerable<Movie>
```

```
<tbody>
```

```
    @foreach (var item in Model)
    {
```

```
        <tr>
```

```
            <td>
```

```
                @Html.DisplayFor(modelItem => item.Title)
```

```
            </td>
```

```
            <td>
```

```
                @{
```

```
                    var allGenres = string.Join(", ", item.Genres.Select(g=>g.Name));
```

```
                }
```

```
                @allGenres
```

```
            </td>
```

```
            <td>
```

```
                <a asp-action="Edit" asp-route-id="@item.Id">Edit</a> |
```

```
                <a asp-action="Details" asp-route-id="@item.Id">Details</a> |
```

```
                <a asp-action="Delete" asp-route-id="@item.Id">Delete</a>
```

```
            </td>
```

```
        </tr>
```

```
    }
```

```
</tbody>
```

עבודה עם ViewModels

רבים לרבים – הצגת סרטים עם זאנר

בגלל המיגרציה שביצענו – המערכת לא מוצאת את ה"טבלה"
GenreMovie

פתרונות

- (1) להשאיר את שם הטבלה המקורי ולא להתערב בתהליך
- (2) נדרוס את המתודה OnModelCreating במחלקה DbContext
ונגדיר את יחסי הגומלין בעצמנו:

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);

    modelBuilder.Entity<Movie>().HasMany(m=>m.Genres)
        .WithMany(g=>g.Movies).UsingEntity<MovieGenres>();
}
```

```
public class MovieGenres
{
    0 references
    public int MoviesId { get; set; }
    0 references
    public int GenresId { get; set; }
}
```

עבודה עם ViewModels

רבים לרבים – הצגת סרטים עם זאנר

בגלל המיגרציה שביצענו – המערכת לא מוצאת את ה"טבלה"
GenreMovie

עוד דרך לפתרון:
נוסיף את המחלקה הבאה למודלים שלנו ול DbContext ונבצע מיגרציה:

```
[PrimaryKey("MoviesId", "GenresId")]  
1 reference  
public class MovieGenres  
{  
    0 references  
    public int MoviesId { get; set; }  
    0 references  
    public int GenresId { get; set; }  
}
```

עבודה עם ViewModels

רבים לרבים – שינוי שם לטבלת רבים לרבים:

```
public class Lec5Context(DbContextOptions<Lec5Context> options) : DbContext(options)
{
    11 references
    public DbSet<Country> Countries { get; set; } = default!;
    7 references
    public DbSet<Capital> Capitals { get; set; } = default!;
    7 references
    public DbSet<Song> Songs { get; set; } = default!;
    8 references
    public DbSet<Album> Albums { get; set; } = default!;
    7 references
    public DbSet<Movie> Movies { get; set; } = default!;
    9 references
    public DbSet<Genre> Genres { get; set; } = default!;
    0 references
    public DbSet<MovieGenres> MovieGenres { get; set; } = default!;
}
```

```
[PrimaryKey("MoviesId", "GenresId")]
1 reference
public class MovieGenres
{
    0 references
    public int MoviesId { get; set; }
    0 references
    public int GenresId { get; set; }
}
```

נמחק את תיקיית Migrations
נריץ ב package manager shell את הפקודות הבאות:
Drop-Database
Add-Migration InitDB
Update-Database

עבודה עם ViewModels

רבים לרבים – יצירת סרט עם בחירה מרחבה של זאנרים

```
public class MovieWithGenreViewModel
{
    5 references
    public Movie Movie { get; set; } = new Movie { Title = string.Empty };
    7 references
    public List<SelectGenre> Genres { get; set; } = [];
}
```

```
2 references
public class SelectGenre
{
    3 references
    public int Id { get; set; }
    4 references
    public required string Name { get; set; }

    3 references
    public bool IsSelected { get; set; }
}
```

עבודה עם ViewModels

רבים לרבים – יצירת סרט עם בחירה מחובה של זאנרים

```
// GET: Movies/Create
```

0 references

```
public async Task<IActionResult> Create()
{
    var genres = await context.Genres.ToListAsync();
    var vm = new MovieWithGenreViewModel()
    {
        Genres = genres.Select(g => new SelectGenre()
        {
            Id = g.Id,
            Name = g.Name,
            IsSelected = false
        }).ToList()
    };
    return View(vm);
}
```

עבודה עם ViewModels

רבים לרבים – יצירת סרט עם בחירה מרובה של זאנרים

הטריק פה – לולאת for i
בלולאה אנחנו מוסיפים input לכל אחת מהתכונות במערך של הזאנרים

חלק מהתכונות נסתרות מהמשתמש – אבל אנחנו רוצים שהן ישלחו יחד עם המידע מהטופס

```
<form asp-action="Create">
  <div asp-validation-summary="ModelOnly" class="text-danger"></div>
  <div class="form-group">
    <label asp-for="Movie.Title" class="control-label"></label>
    <input asp-for="Movie.Title" class="form-control" />
    <span asp-validation-for="Movie.Title" class="text-danger"></span>
  </div>
  @for (var i = 0; i < Model.Genres.Count; i++)
  {
    <div class="form-group">
      <label class="control-label" >@Model.Genres[i].Name</label>
      <input asp-for="Genres[i].IsSelected" />
      <input asp-for="Genres[i].Name" type="hidden" />
      <input asp-for="Genres[i].Id" type="hidden" />
    </div>
  }
  <div class="form-group">
    <input type="submit" value="Create" class="btn btn-primary" />
  </div>
</form>
```


עבודה עם ViewModels

רבים לרבים – יצירת סרט עם בחירה מרובה של זאנרים

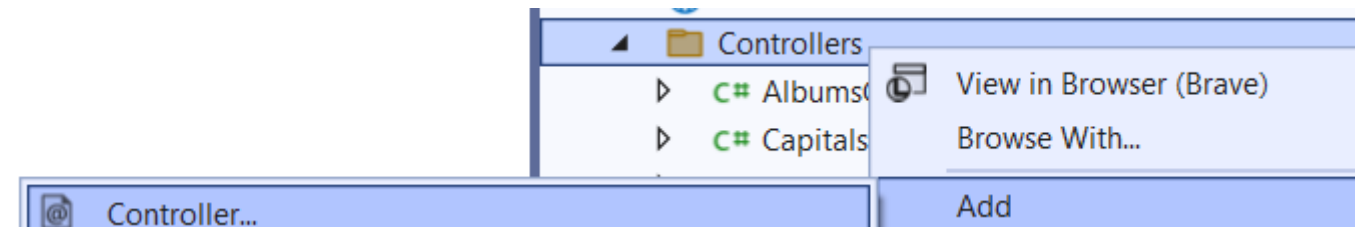
```
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Create(MovieWithGenreViewModel vm)
{
    if (ModelState.IsValid)
    {
        var genres = vm.Genres
            .Where(vg => vg.IsSelected)
            .Select(vg => new Genre() { Name = vg.Name, Id = vg.Id })
            .ToList();

        genres.ForEach(g => context.Attach(g));

        vm.Movie.Genres = genres;
        context.Add(vm.Movie);
        await context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }
    return View(vm);
}
```

הגשה של API מפרוייקט MVC קיים

מעולה אם חצים ליצור API עם עמודים לדוקומנטציה



Add New Scaffolded Item

▲ Installed

▲ Common




API

▲ MVC

Controller

View

Razor Component

	API Controller - Empty	API Controller with actions, using Entity Framework by Microsoft v1.0.0.0 An API controller with REST actions to create, read, update, delete, and list entities from an Entity Framework data context.
	API Controller with read/write actions	
	API Controller with actions, using Entity Framework	

שיעור בית:

נתונה דיאגרת ERD

צרו דטה-בייס מתאים לדיאגרמה בעזרת Entity Framework

כלומר יש ליצור מחלקה עבור Post

מחלקה עבור User

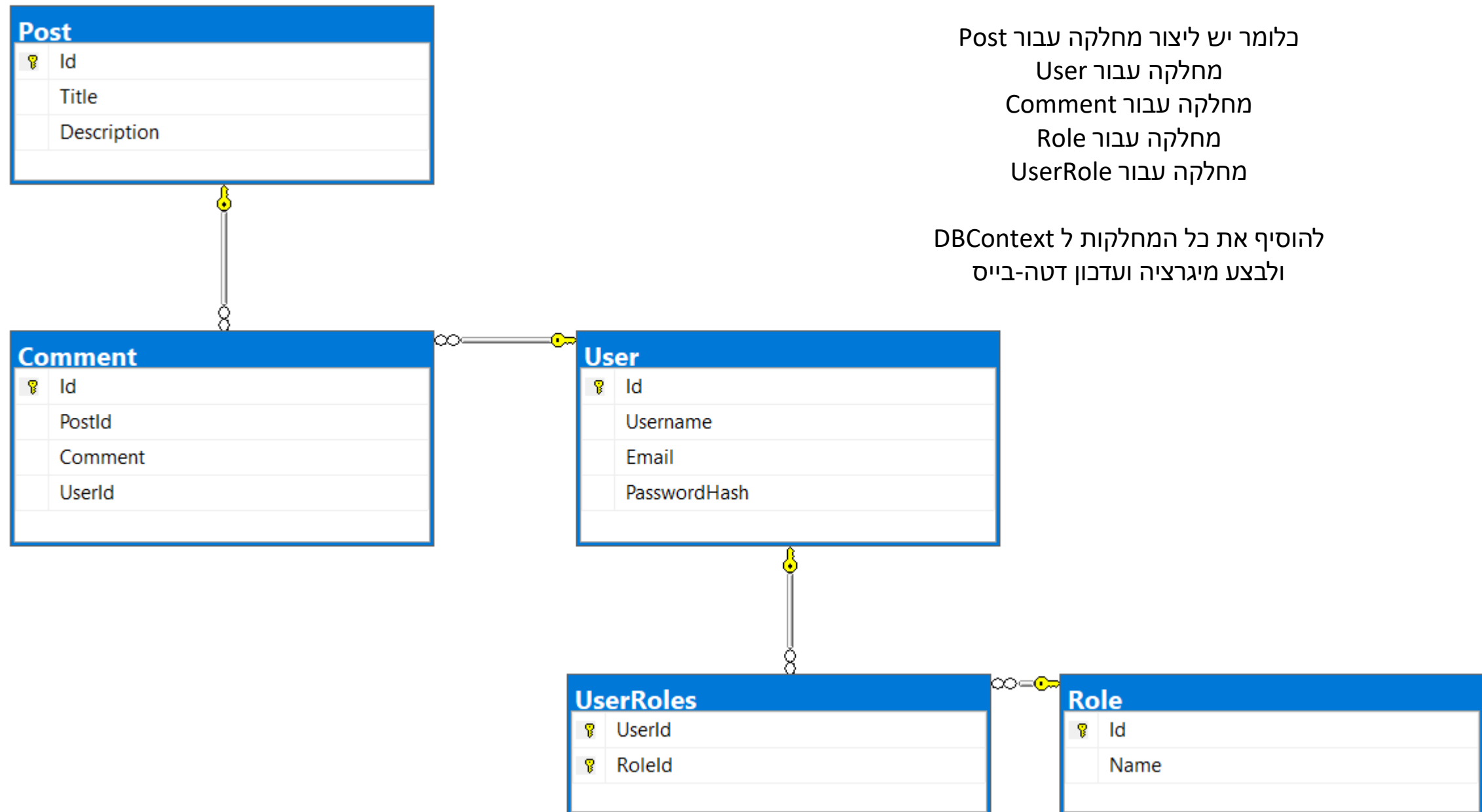
מחלקה עבור Comment

מחלקה עבור Role

מחלקה עבור UserRole

להוסיף את כל המחלקות ל DbContext

ולבצע מיגרציה ועדכון דטה-בייס



רעיונות לתרגול נוסף:

צרו Controller ליצירת כתבה
הציגו את הכתבה עם התגובות שלה
ואפשרות להוספת תגובה בתחתית עמוד Index