Novi Projekat

1. Napravi folder Model
2. Napravi BaseModel koji će sadržati Id, CreateDate, DeleteDate i UpdateDate attribute
3. Napravi ostale Modele koji će svi naslijediti BaseModel
4. Konfiguriši EF i napravi DataContext Kako konfigurisati Entity Framework (EF) i napraviti DB
5. Napravi folder Repository
6. Napravi podfolder BaseRepository u folderu Repository
7. Napravi generični interface IBaseRepository i definiši sve CRUD funkcije u njemu
8. Napravi generičnu klasu BaseRepository i implementiraj interface IBaseRepository
9. Registruj BaseRepository za svaki Model u Program.cs kao Transient: builder.Services.AddTransient<IRepositoryBase<ImeModela>, RepositoryBase<ImeModela>>();
10. Konfiguriši AutoMapper
11. Napravi folder Services
12. Napravi za svaki Model jedan folder koji se zove **ImeModela**Service

Kako instalirati biblioteke

1. Desni klik na Projekat > Manage NuGet Packages…
2. U gornjem Tab-u klikni na Browse
3. Pomoću Search-a pronađi željenu biblioteku i odaberi je
4. Klikni na install

Kako konfigurisati Entity Framework (EF) i napraviti DB

1. Instaliraj potrebne biblioteke (Kako instalirati biblioteke):
    a. Microsoft.EntityFrameworkCore
    b. Microsoft.EntityFrameworkCore.Design
    c. Microsoft.EntityFrameworkCore.SqlServer
    d. Microsoft.EntityFrameworkCore.Tools
2. Napravi novi folder u projektu (Naziv može biti DataAccess)
3. Napravi novi interface i nazovi ga IDataContext
4. U interface unesi sljedeće attribute / funkcije:
    a. int SaveChanges(); **=> Funkcija za da preslika promjene u DB**
    b. void Seed(); **=> Funkcija za ubrizgavanje default vrijednosti u DB**
    c. DatabaseFacade Database { get; } **=> Atribut preko kojeg možemo pristupiti samoj DB**
5. U istom folder napravi klasu DataContext
6. Naslijedi DbContext klasu iz EF biblioteke i interface IDataContext
7. Napravi konstruktor

8. Proslijedi DbContextOptions preko konstruktora u baznu klasu koju si naslijedila:
   `public DataContext(DbContextOptions options) : base(options) { }`
9. Napravi za svaki Model, iz kojeg želiš da napraviš tabelu, jedan atribut tipa
   `DbSet<T>public DbSet<User> Users { get; set; }`
10. Dodaj connection-string u appsettings.json
11. Dodaj DbContext u Program.cs i proslijedi mu connection string kao opciju
    ```
    builder.Services.AddDbContext<DataContext>(x =>
    {x.UseSqlServer(builder.Configuration.GetConnectionString("Development"));
    })
    ```
12. Registruj DataContext kao Transient u Program.cs, da bi ga mogla ubrizgavati pomoću Dependency Injection
    `builder.Services.AddTransient<IDataContext, DataContext>();`
13. Napravi novu migraciju koja se zove Init (inicijalna migracija) Kako napraviti novu migraciju i okinuti je
14. Okini update baze podatke

Kako napraviti novu migraciju i okinuti je

1. Klikni na Tools u gornjem Toolbar-u u VS
2. Stavi kursor na NuGet Package Manager
3. Odaberi Package Manager Console, otvorit će se konzola u donjem djelu
4. U konzoli unesi komandu:
   `Add-Migration StaviImeMigracijeOvde`
5. Izvrši promjene u bazi tako što ćeš okinuti komandu u konzoli:
   `Update-Database`

Kako konfigurisati AutoMapper

1. Instaliraj potrebne biblioteke (Kako instalirati biblioteke):
   a. AutoMapper.Extensions.Microsoft.DependencyInjection
   b. AutoMapper
2.