יצירת פרויקט עם חיבור לDB

* ליצור database בSQL-
* ליצור פרויקט מסוג web application API asp.net core
* ליצור פרויקט מסוג Class library .core -DAL BL DTO
* ליצור קונטרולרים לכל טבלה
* ליצור אינטרפייסים ב-DTO וב-DAL
* להוסיף בקובץ appsettings.json

אחרי "\*" פסיק ואז:

"ConnectionString": {

"ProductsConnection": "Data Source=.;Initial Catalog=products;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False"

}

את הdata source להעתיק מהקונקשן סטרינג

התקנות – tools -> NuGet Package Manager -> console….

להתקין על הפרויקט הראשי - Microsoft.EntityFrameworkCore

להתקין על הDAL:

1. Microsoft.EntityFrameworkCore.SqlServer
2. Microsoft.EntityFrameworkCore.Tools
3. Microsoft.EntityFrameworkCore.Design
4. Microsoft.EntityFrameworkCore

להריץ את הפקודה על הפרויקט DAL ולהגדיר לפני זה set as startup project:

Scaffold-DbContext " Data Source=.;Initial Catalog=products;Integrated Security=True;Connect Timeout=30;Encrypt=False;TrustServerCertificate=False;ApplicationIntent=ReadWrite;MultiSubnetFailover=False" Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models

להוסיף project references:

על הפרויקט הראשי – BL, DTO

על הDTO- - DAL

על הDAL- – כלום

על הBL – DTO, DAL

יצירת פרויקט אנגולר:

פתיחת קובץ חדש:

ב cmd להריץ: npm install @angular/cli -g

לפתוח את הcmd על הניתוב של התקיה החדשה שיצרתי

ng new example (example -שם הפרויקט)

לפתוח את התקיה החדשה ב visual studio code

לעשות: npm i **אם** עוצרים באמצע את ההתקנות.

להתקין בשביל שרת: express,

כדי להריץ את הפרויקט: npm run ng s

קובץ proxy לחיבור לשרת:

להוסיף קובץ במיקום [src](https://github.com/ShoshyAidelman/GPS_Angular/tree/main/src)/proxy.config.json

ובתוכו לכתוב:

|  |
| --- |
| { |
|  | "/api/\*":{ |
|  | "target":"http://localhost:8000/", |
|  | "source":"false" |
|  | } |
|  | } |

לקחת את כתובת השרת מקובץ הסי-שארפ :

launchSettings<- applicationUrl

הניתוב:

/Api/\* כשם שניתן ב-C# בקובץ launchSettings ב launchUrl

כמו כן לזכור לשנות בקובץ launchSettings:

sslPort:0

בפונקציות נרשם גם כן בכל פעם הניתוב ללא המילה API

APPModule Angular

|  |
| --- |
| @NgModule({ |
|  | declarations: [ |
|  | AppComponent, |
|  | // ApartmentTypeComponent-all components declare here |
|  | ], |
|  | imports: [ |
|  | BrowserModule, |
|  | AppRoutingModule, |
|  | HttpClientModule, |
|  | BrowserAnimationsModule |
|  | ], |
|  | providers: [ApartmentService],-all service declare here |
|  | bootstrap: [AppComponent] |
|  | }) |
|  | export class AppModule { } |

APP.SERVICE angular

|  |
| --- |
| @Injectable() |
|  | export class AppService{ |
|  | constructor(private \_http: HttpClient) { |
|  | console.log( |
|  | this.\_http.get('/apartmentType')-הניתוב לפי מה ששהוחלט בC#- |
|  | ) |
|  | } |

בסרביס של הקומפוננטה:

|  |
| --- |
| @Injectable() |
|  | export class ApartmentService{ |
|  |  |
|  | constructor(private \_http: HttpClient) { } |
|  | * getApartmentTypes():Observable<ApartmentType[]>{ |
|  | return this.\_http.get<ApartmentType[]>('http://localhost:8000/apartmentType'); |
|  | } |
|  | } |

לשים את שם הפונקציה והיא תואמת לשם הפונקציה בC#

הסוג מהמודל

שם הסרבר

ניתוב

בקומפוננטה בקובץ TS:

|  |
| --- |
| @Component({ |
|  | selector: 'app-apartment-type', |
|  | templateUrl: './apartment-type.component.html', |
|  | styleUrls: ['./apartment-type.component.scss'] |
|  | }) |
|  | export class ApartmentTypeComponent { |
|  |  |
|  | constructor(private \_apartmentTypeService:ApartmentService) { |
|  | this.\_apartmentTypeService.getApartmentTypes().subscribe(data=>{ |
|  | console.log(data); |
|  | }) |
|  | } |
|  |
|  |
|  | } |