**Arduini enimboldi – sviluppo back**

**Fase 1**

Il cliente si presenta al desk di Arduini con il suo mezzo e segnala un malfunzionamento, viene aperta una commessa ed associata ad essa un codice commessa.

Lato front (app)

Successivamente il cliente potrà registrarsi all’app inserendo email (che fa da chiave univoca), password, numero telefono ed eventualmente nome e cognome (non obbligatorio??) la registrazione prevede la conferma email e successivamente potrà loggarsi, prevedere anche il recupero password.

(La conferma email e da far confermare dal cliente)

A questo punto vedrà lo storico dei feedback lasciati, e potrà inserirne di nuovi.

**L’inserimento feedback** prevede innanzitutto l’inserimento del codcommessa sulla conferma del codice avverrà un check che controlla se per quel codice commessa è stato già aggiunto il feedback ed in caso contrario se esiste il codice commessa nel db del cliente, se vengono passati i controllia questo punto si potranno scegliere tra 4 livelli di valutazione di cui uno positivo, se si sceglie il positivo e si conferma una api recupererà dal database dei clienti targa, codice per matchare la sede, data esecuzione interventoecc… e salverà il feedback (il salvataggio prevederà l’inserimento di iduser,datainserimentofeedback,stato,info recuperate di cui sopra. Se il feedback è negativo si andrà avanti di un altro passaggio per chiedere come mai si da giudizio negativo e su conferma avverrà la chiamata alla api di salvataggio feedback questa volta con in più il salvataggi del campo descrizione motivo feedback negativo. (in caso di esito negativo verrà inviato un alert a backoffice) e gli id dei servizi per i quali il cliente non è soddisfatto.

Lato backoffice avremo una interfaccia così:

|  |  |
| --- | --- |
|  |  |
| Utenti App | Griglia in sola lettura con elenco utenze (Nome cognome, data registrazione, email,telefono |
| Feedback | Datainserimento, codcommessa, sede, nome utente, feedback, descrizione (in caso di negatività) |
| FeedbackNegativi | Datainserimento, codcommessa, sede, nome utente, feedback, descrizione (in caso di negatività) |
| Sedi | CRUD delle sedi. |
| Utenti operativi | CRUD per Utenti. |
| Log notifiche | In sola lettura |

I ruoli che potranno accedere sono **superadmin** (fa tutto)**admin**(vede il crud della creazione utenze operative), **manager** (vede solo le utenze a lui associata mediante targhe),**utenza operativa**.

Occorre poi prevedere un job che potrebbe essere una appconsole che recupera dal db del cliente tutte le commesse aperte supponiamo l’ultimo mese (la cui datainserimento è superiore al mese), le metchia mediante codcommessa alla tabella dei feedback

* + Data inserimento
  + codcommessa
  + Data intervento (Data\_Commessa nel DMZ)
  + ID Utente
  + ID Esito
  + ID Targa/Veicolo (id nostro)
  + ID Sede (id nostro)
  + Commento se esito non pienamente positivo

e per quelle che non trovano riscontro che si trovano nel db del cliente va a vedere, per ciascun utente, se il campo email o telmecchia, in caso affermativo manda una notifica push (solo se campo datainviofeedback non è già completato) ed inserisce dentro tabella notifica la datainvio ed idcommessa (così si evita di reinviare più volte).

Campi notifica:

* Data Inserimento
* ID User
* ID Tipo (ad ora solo Notifica)
* IdCommessa
* Messaggio

**Step per lo sviluppo**

Vedere se convivono cookies autentication e token

Implementare login, logout, (recupero pwd).

Ed in base a ruoli mostrare crud utenti (da implementare)

|  |  |
| --- | --- |
| step | ore |
| Creare progetto applicazione web |  |
| Abilitare identity su postgress | 2 |
| Aggiungere scaffoldingidentity | 2 |
| Aggiungere jvt e swagger |  |
| Vedere se le due autenticazioni convivono |  |
| Con token (login, registrazione, conferma email, reimposta password:todo qui vedere come si fa con token quindi fare dopo, logica di rinnovo token, il token scade dopo tot il client usa il token se non è più valido si rilogga e ne ottiene un altro). | 16 |
| Applicare template bootstrap |  |
| Implementare login, logout con cookies |  |
| CRUD utenti operativi (suggerimento UNO) | 8 |
| Associazione ruoli (combo) | 4 |
| Test policies su action in base a ruoli | 4 |
| CRUD sedi (recupero da tecres) | 4 |
| Inserisci feedback (con recupero dati da database cliente | 24 |
| getfeedback | 2 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| In sola lettura anagrafica feedback | 8 |
| Job | 16 |
|  | Tot 98=12 gg |

Interfacce da formattare con html:

login

Ore lavorative a casa (34)

10 creazione jwtoken ed infrastruttura identity

4 formattazione bootstrap

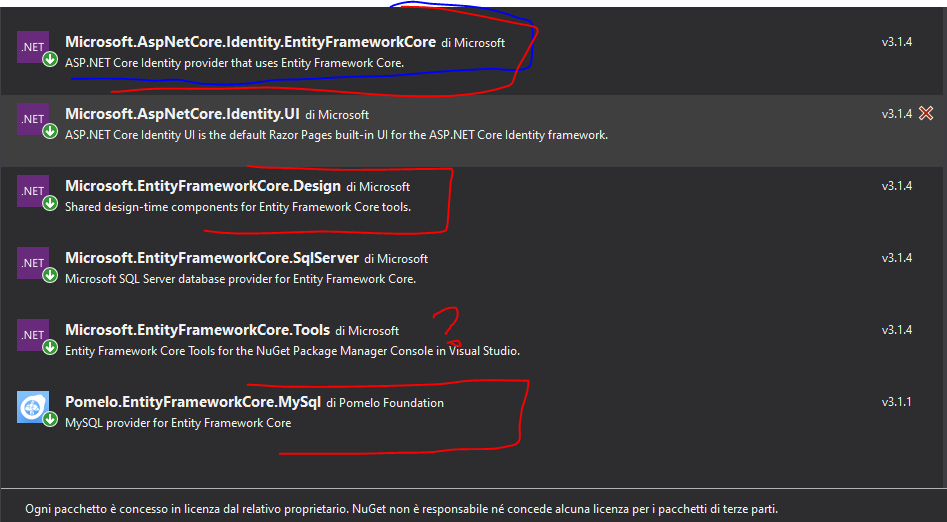
2 impostazione api

6 crud utenze (implementazione paginatore, ordinamento, filtro)

12 interfaccie varie

Creato progettomvc senza security

Aggiuntinugetpackage permigration con mariadb:



Aggiunto dbcontext:

creata in root cartella context ed aggiunto dbcontext

usingArduiniNerboldiBack.Etities;

usingMicrosoft.AspNetCore.Identity;

usingMicrosoft.AspNetCore.Identity.EntityFrameworkCore;

usingMicrosoft.EntityFrameworkCore;

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Threading.Tasks;

namespaceArduiniNerboldiBack.Context

{

publicclassArduiniDbContext : IdentityDbContext<ApplicationUser>

{

publicArduiniDbContext(DbContextOptions<ArduiniDbContext> options)

: base(options)

{

}

protectedoverridevoidOnModelCreating(ModelBuilder builder)

{

base.OnModelCreating(builder);

}

}

}

Creata cartella enitites ed aggiunto application user

Aggiunto Applicationuser

usingMicrosoft.AspNetCore.Identity;

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Threading.Tasks;

namespaceArduiniNerboldiBack.Etities

{

publicclassApplicationUser:IdentityUser

{

}

}

Registrato il dbcontext:

publicvoidConfigureServices(IServiceCollection services)

{

stringconnString = ConfigurationExtensions.GetConnectionString(this.Configuration, "DefaultConnection");

services.AddDbContext<ArduiniDbContext>(options => options

.UseMySql(connString,

mysqlOptions =>

mysqlOptions.ServerVersion(newServerVersion(new Version(10, 5, 0), ServerType.MariaDb))));

services.AddControllersWithViews();

}

E poi ancora:

services.AddDefaultIdentity<ApplicationUser>(options =>options.SignIn.RequireConfirmedAccount = true)

.AddEntityFrameworkStores<ArduiniDbContext>();

ed ancora:

app.UseAuthentication();

Aggiuntoper la migrazione automatica

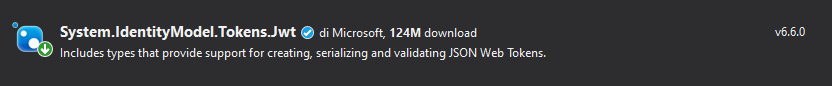
:

dbContext.Database.Migrate();

Aggiunta JWT autentication:

creata cartella providers ed aggiunto IJwtTokenProvider e JwtTokenProvider

installare seguente apcchetto per TokenValidationParameters

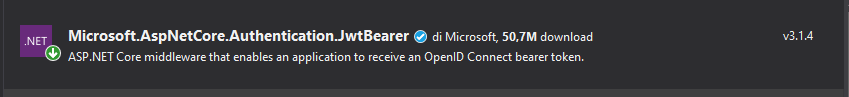


Aggiunta cartella middlewares e dentro

JwtTokenMiddleware.cs

Dentro model aggiunto

JwtToken.cs



Poi configuro in startup:

services.Configure<JwtToken>(Configuration.GetSection("JwtToken"));

services.AddSingleton<IJwtTokenProvider, JwtTokenProvider>();

services.AddAuthentication()

.AddJwtBearer(options =>

{

vartokenProvider = services.BuildServiceProvider().GetService<IJwtTokenProvider>();

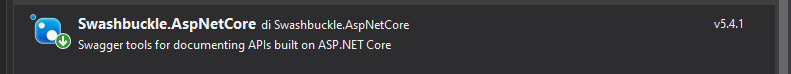
options.TokenValidationParameters = tokenProvider.GetTokenValidationParameters();

});

quindi aggiongo il tokenController:

**Installazione Swagger**

Installare pacchetto nuget



Registrare in startup:

services.AddSwaggerGen(c =>

{

//c.SwaggerDoc("v1", new Info { Title = "You api title", Version = "v1" });

c.SwaggerDoc("v1", new OpenApiInfo

{

Version = "v1",

Title = "Arduini - Nerboldi API",

Description = "Documentazione per chiamare le API",

//TermsOfService = new Uri("https://example.com/terms"),

//Contact = new OpenApiContact

//{

// Name = "Shayne Boyer",

// Email = string.Empty,

// Url = new Uri("https://twitter.com/spboyer"),

//},

//License = new OpenApiLicense

//{

// Name = "Use under LICX",

// Url = new Uri("https://example.com/license"),

//}

});

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

Description = @"JWT Authorizationheaderusing the Bearerscheme. \r\n\r\n

Enter 'Bearer' [space] and thenyour token in the text input below.

\r\n\r\nExample: 'Bearer 12345abcdef'",

Name = "Authorization",

In = ParameterLocation.Header,

Type = SecuritySchemeType.ApiKey,

Scheme = "Bearer"

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement()

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "Bearer"

},

Scheme = "oauth2",

Name = "Bearer",

In = ParameterLocation.Header,

},

new List<string>()

}

});

//varxmlFile = $"{Assembly.GetExecutingAssembly().GetName().Name}.xml";

//varxmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);

//c.IncludeXmlComments(xmlPath);

});

Quindi aggiungere il middleware prima del routing:

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.RoutePrefix = "apidocs";

c.SwaggerEndpoint("/swagger/v1/swagger.json", "Arduini - Nerboldi API V1");

});

Per abilitare i commenti:

Dentro il file di progetto inserire

<PropertyGroup><GenerateDocumentationFile>true</GenerateDocumentationFile><NoWarn>$(NoWarn);1591</NoWarn></PropertyGroup>

E successivamente:

services.AddSwaggerGen(c =>

{

//c.SwaggerDoc("v1", new Info { Title = "You api title", Version = "v1" });

c.SwaggerDoc("v1", new OpenApiInfo

{

Version = "v1",

Title = "Arduini - Nerboldi API",

Description = "Documentazione per chiamare le API",

//TermsOfService = new Uri("https://example.com/terms"),

//Contact = new OpenApiContact

//{

// Name = "Shayne Boyer",

// Email = string.Empty,

// Url = new Uri("https://twitter.com/spboyer"),

//},

//License = new OpenApiLicense

//{

// Name = "Use under LICX",

// Url = new Uri("https://example.com/license"),

//}

});

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

Description = @"JWT Authorizationheaderusing the Bearerscheme. \r\n\r\n

Enter 'Bearer' [space] and thenyour token in the text input below.

\r\n\r\nExample: 'Bearer 12345abcdef'",

Name = "Authorization",

In = ParameterLocation.Header,

Type = SecuritySchemeType.ApiKey,

Scheme = "Bearer"

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement()

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "Bearer"

},

Scheme = "oauth2",

Name = "Bearer",

In = ParameterLocation.Header,

},

new List<string>()

}

});

varxmlFile = $"{Assembly.GetExecutingAssembly().GetName().Name}.xml";

varxmlPath = Path.Combine(AppContext.BaseDirectory, xmlFile);

c.IncludeXmlComments(xmlPath);

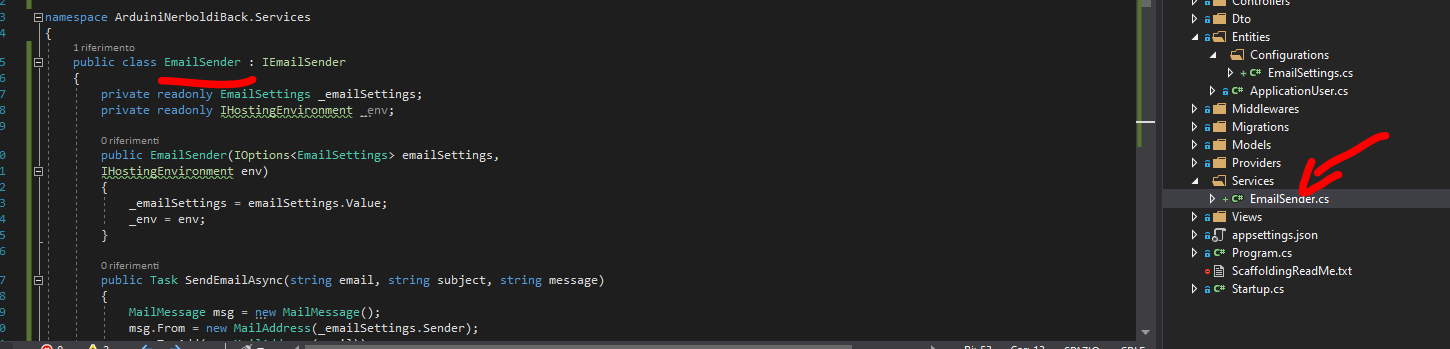
});

**Invio mail:**

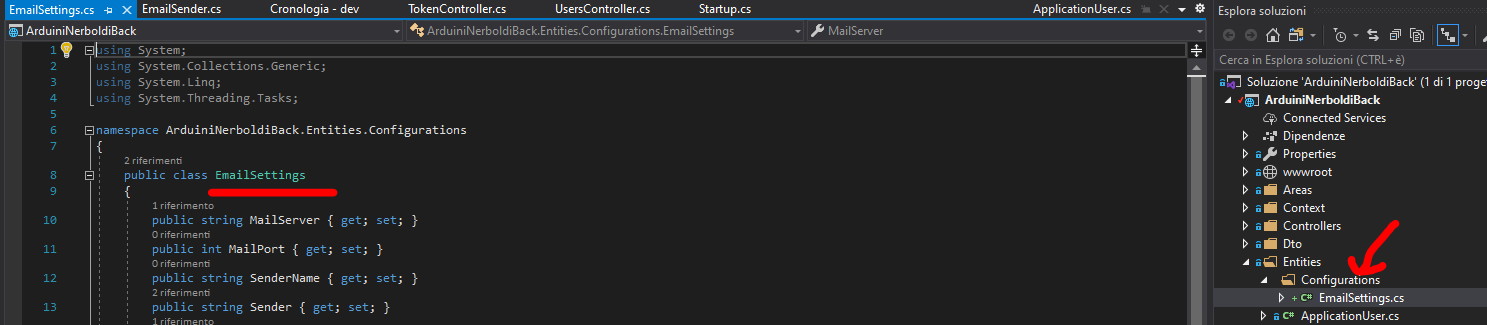
Spedizione email:

Aggiungere il servizio EmailSender che implementa IEmailSender (vedi uno)

Aggiunta la cartella services e dentro Email sender:



Aggiunta cartella configuration dentro entities ed aggiunto email setting:

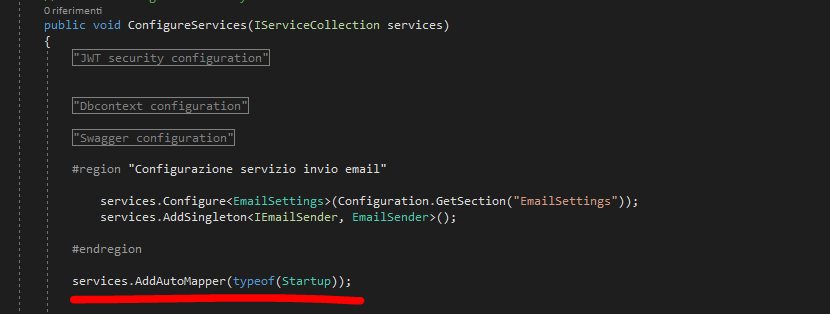


Registrare in configure services:

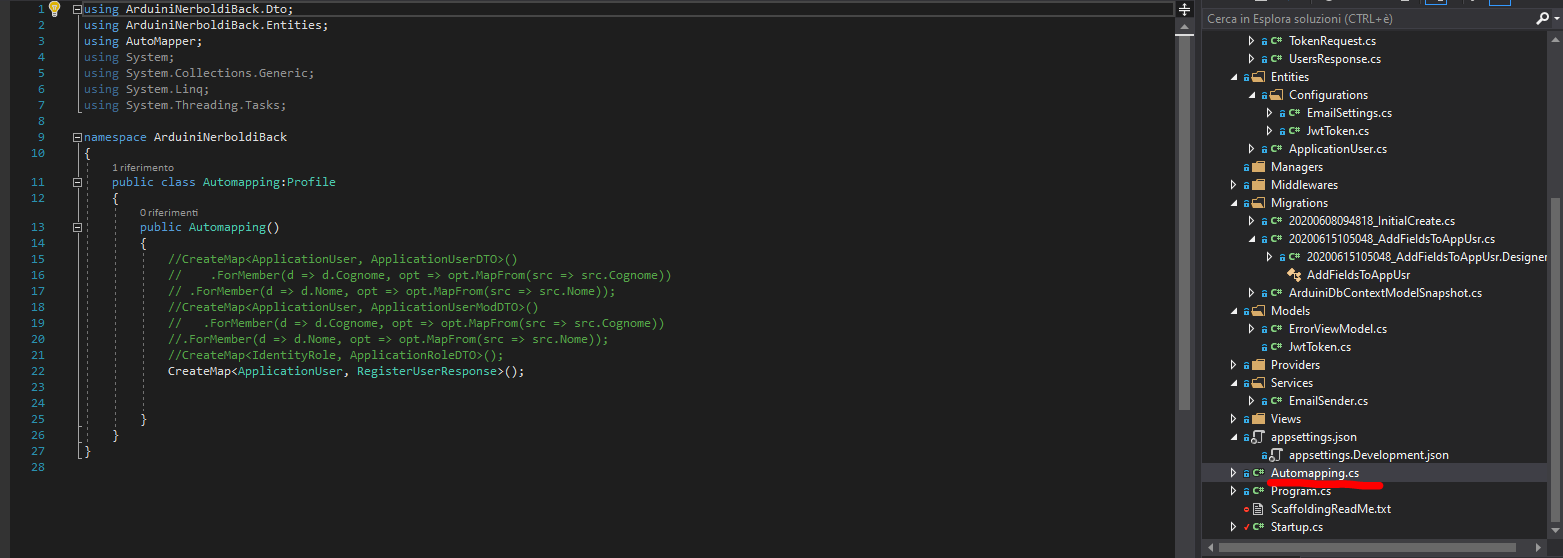
services.Configure<EmailSettings>(Configuration.GetSection("EmailSettings"));

services.AddSingleton<IEmailSender, EmailSender>();

Aggiunto dbmapper:

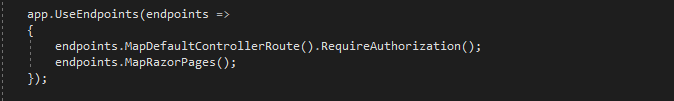


Creare file automapping.cs:



Configurare affinche venga avviata la pagina razor di login:

Commentare l’endpoint di default e sostituire con:



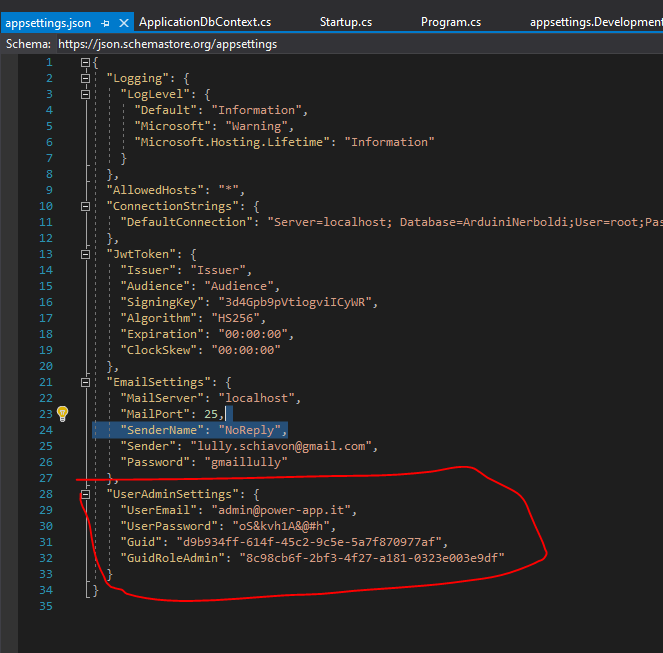
Inserire template bootstrap:

Copiata la cartella vedor e css, img, js, scss dentro wwwroot

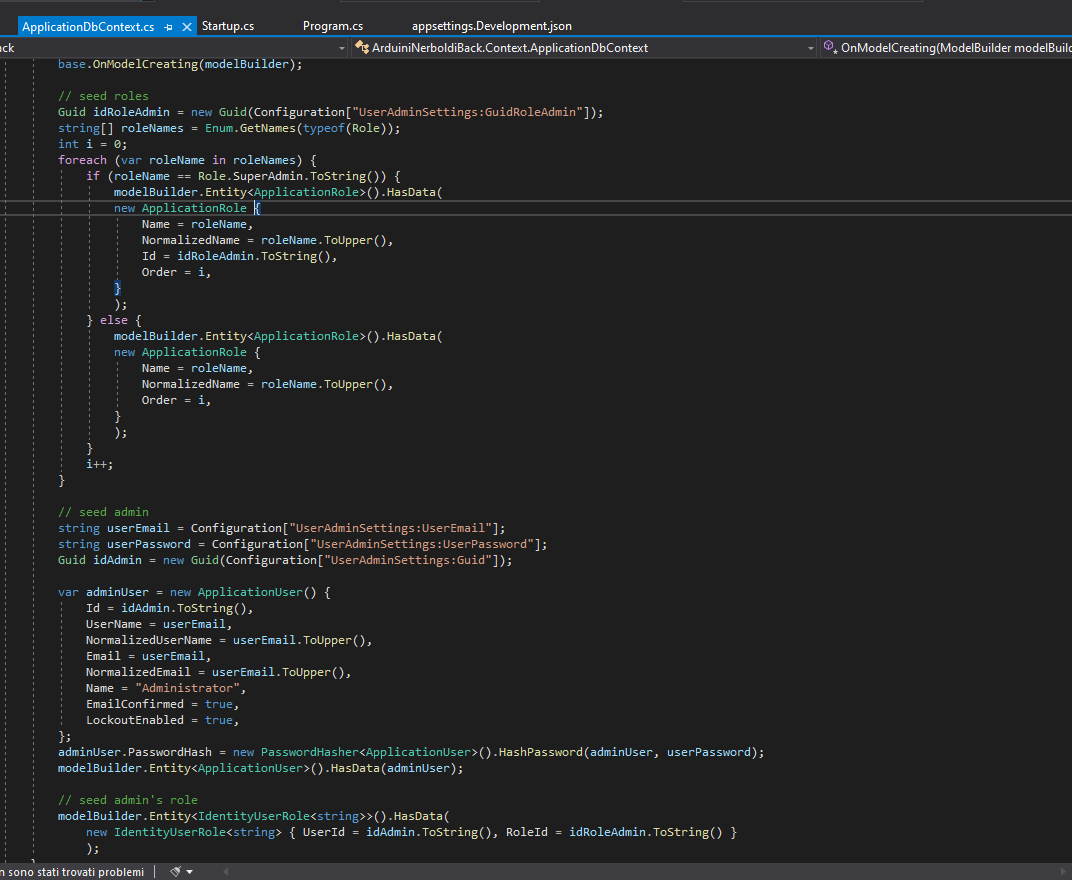
Copiato package.jos e package.json.lok e gulp.js in root

Modificare il gulpfile...

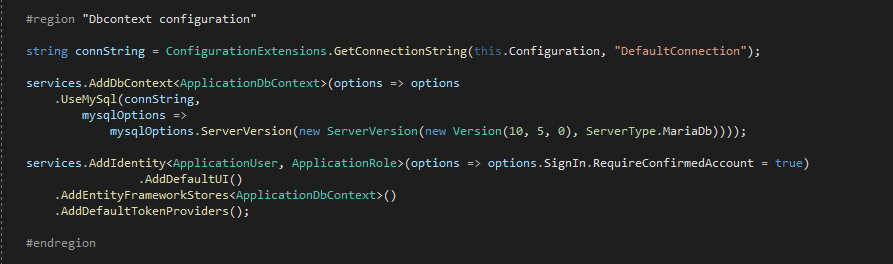
Aggiunta ruoli ed utenti nuiltin:



Aggiungiamo nll’onmodelcreating del dbcontext i ruoli basati su una enumerazione ed un utente superadmin associandolo al ruolo superadmin:



Modifichiamo in startup in configureservices la configurazione per aggiungere la gestione dei ruoli, aggiungendo un applicationrole:



**GENERAZIONE ENTITIES CUSTOM**

Generiamo quindi tutte le entities per le tabelle custom e quindi aggiungiamo i dbset nel contest e con l’apifluent aggiungiamo le relazioni per navigare i dati

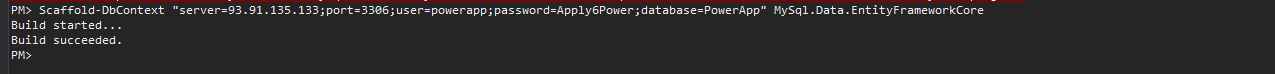
**GENERAZIONE ORM per accesso a database mysql del cliente Arduini**

Installare questo pacchetto nuget:



Lanciare la seguente istruzione da console di gestione pacchetti

**Scaffold-DbContext "server=93.91.135.133;port=3306;user=powerapp;password=Apply6Power;database=PowerApp" MySql.Data.EntityFrameworkCore**



Sistemazione delle entities nelle cartelle opportune...

CRUD (paginazione, ordinamento filtro)

Per la paginazione usiamo il cloudscribe.we.b.pagination

Includiamo in csproj i seguenti riferimenti

<**PackageReference** Include="cloudscribe.Web.Pagination" Version="2.1.\*" />

<**PackageReference** Include="cloudscribe.Pagination.Models" Version="1.0.\*" />

Aggiungiamo allo startup:

services.AddCloudscribePagination();

Aggiungiamo a \_ViewImports.cshtml il seguente testo

@addTagHelper "\*, cloudscribe.Web.Pagination"

Aggiungiamo il tag pager subito dopo il tag table

<div>

<cs-pager cs-paging-pagesize="@Model.Users.PageSize"

cs-paging-pagenumber="@Model.Users.PageNumber"

cs-paging-totalitems="@Model.Users.TotalItems"

cs-pagenumber-param="p"

cs-pager-li-current-class="page-item active"

cs-pager-li-other-class="page-item"

cs-pager-li-non-active-class="page-item disabled"

cs-pager-link-current-class="page-link"

cs-pager-link-other-class="page-link">

</cs-pager>

</div>

Aggiungiamo il codebehind nell’index del controller:

model.Users.Data = allEnititiesDTO

.Skip(offset)

.Take(DefaultPageSize)

.ToList();

model.Users.PageNumber = currentPageNum;

model.Users.PageSize = DefaultPageSize;

model.Users.TotalItems = allEnititiesDTO.Count;

Dove il model è così:

namespace ArduiniNerboldiBack.Areas.Identity.Dto

{

public class ApplicationUserListViweModel

{

public ApplicationUserListViweModel()

{

Users = new PagedResult<ApplicationUserDto>();

}

public string Query { get; set; } = string.Empty;

public PagedResult<ApplicationUserDto> Users { get; set; } = null;

}

}

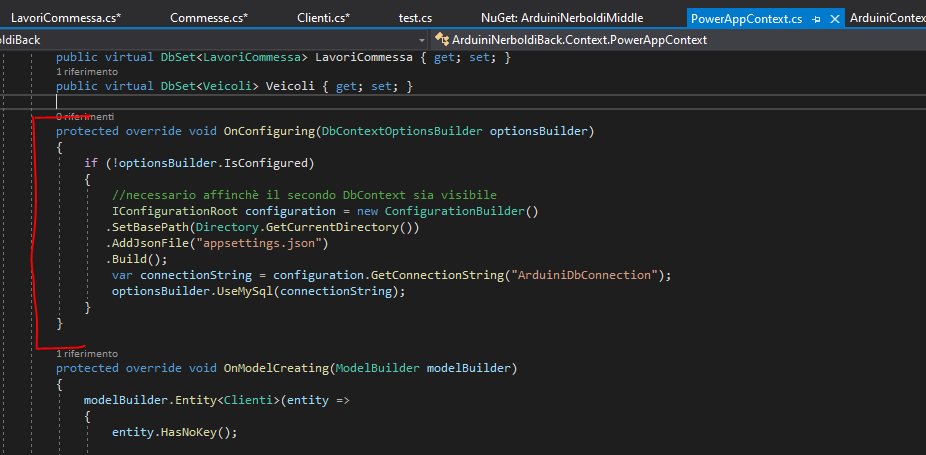
Quindi andiamo ad aggiungere ordinamenti...

PROBLEMATICA NAVIGAZIONE TABELLE IDENTITY

Attenzione quando si creano delle entitaapplication derivate da identity es applicationrole derivato da identityrole o applicationuserroleidentityuserrole per poter navigare tra i dati, questo aggiunge nel database un nuovo campo discriminator che da problemi con l’update dei dati, forzare il fatto che questo campo non venga generato aggiungendo la datanotation [NotMapped] che impedisce la creazione del discriminator.

PROBLEMATICA Utilizzo secondo dbcontext arduini:

Aggiunto questo blocco di codice dentro il dbcontextarduini:



protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

//necessario affinchè il secondo DbContext sia visibile

IConfigurationRoot configuration = new ConfigurationBuilder()

.SetBasePath(Directory.GetCurrentDirectory())

.AddJsonFile("appsettings.json")

.Build();

var connectionString = configuration.GetConnectionString("ArduiniDbConnection");

optionsBuilder.UseMySql(connectionString);

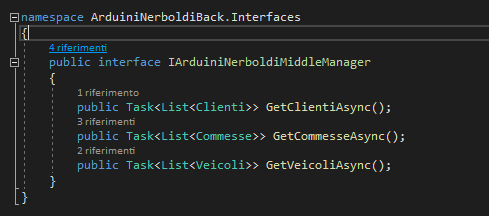
}

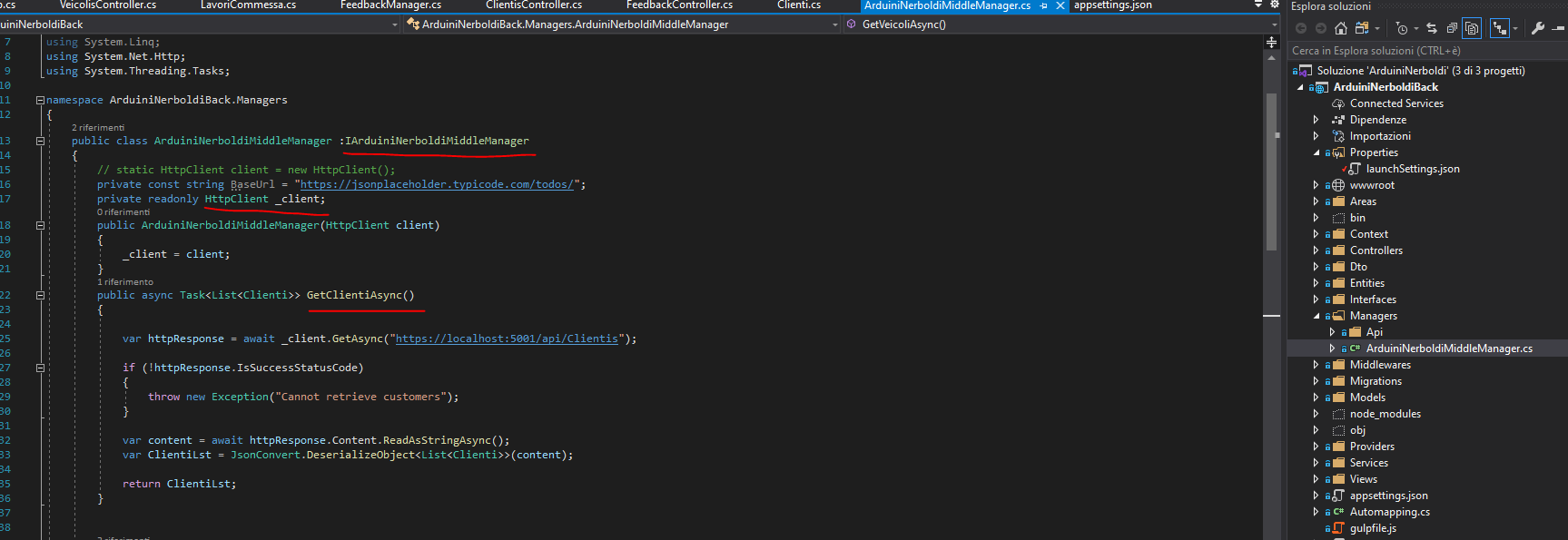
}

CREAZIONE MIDDLEWARE per accesso dati DB Cliente

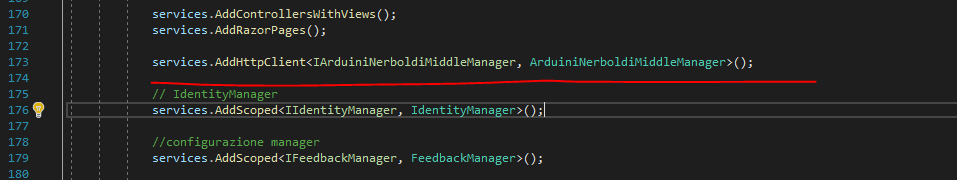
Creato nuovo progetto arduininerboldimiddle; lanciando la procedura da linea di comando creo il dbcontext approcci db first poi creo i controller api basandosi su entity framework usando gli scaffolding visual studio, spsto le entities in una libreia esterna per condividerle con l’applicativo principale.

Quindi nell’applicativo principale creo il manager per gestire le chiamate al middleware:





Poi in configureservices va registrato in questo modo:



Addhttpclient permette l’iniezione del suo componente interno al manager che è httpclient