.Core: Tao Database: Cu phap

[Table("PbPersons")]

public class Person : FullAuditedEntity

{

public const int MaxNameLength = 32;

public const int MaxSurnameLength = 32;

public const int MaxEmailAddressLength = 255;

[Required]

[MaxLength(MaxNameLength)]

public virtual string Name { get; set; }

[Required]

[MaxLength(MaxSurnameLength)]

public virtual string Surname { get; set; }

[MaxLength(MaxEmailAddressLength)]

public virtual string EmailAddress { get; set; }

}

**.EntityFramework: thêm Dbset**

public class PhoneBookDbContext : AbpZeroDbContext<Tenant, Role, User>

{

public virtual IDbSet<Person> Persons { get; set; }

//...other entities

public PhoneBookDbContext()

: base("Default")

{

}

//...other codes

}

.**migration: Lưu dấu và tạo method tạo Person.**

.**migration.Seed: Data Seed cho ứng dụng.**

. **.ApplicationShare: Tạo Interface và DTOs**

public interface IPersonAppService : IApplicationService

{

ListResultDto<PersonListDto> GetPeople(GetPeopleInput input);

});

public class GetPeopleInput

{

public string Filter { get; set; }

}

[AutoMapFrom(typeof(Person))]

public class PersonListDto : FullAuditedEntityDto

{

public string Name { get; set; }

public string Surname { get; set; }

public string EmailAddress { get; set; }

}

.Application: tạo method dùng interface

public class PersonAppService : PhoneBookAppServiceBase, IPersonAppService

{

private readonly IRepository<Person> \_personRepository;

public PersonAppService(IRepository<Person> personRepository)

{

\_personRepository = personRepository;

}

public ListResultDto<PersonListDto> GetPeople(GetPeopleInput input)

{

var persons = \_personRepository

.GetAll()

.WhereIf(

!input.Filter.IsNullOrEmpty(),

p => p.Name.Contains(input.Filter) ||

p.Surname.Contains(input.Filter) ||

p.EmailAddress.Contains(input.Filter)

)

.OrderBy(p => p.Name)

.ThenBy(p => p.Surname)

.ToList();

return new ListResultDto<PersonListDto>(ObjectMapper.MapTo<List<PersonListDto>>(persons));

}

}.MapTo<List<PersonListDto>>(persons));

}

}Tạo quyền

var phoneBook = pages.CreateChildPermission(AppPermissions.Pages\_Tenant\_PhoneBook, L("PhoneBook"), multiTenancySides: MultiTenancySides.Tenant);

phoneBook.CreateChildPermission(AppPermissions.Pages\_Tenant\_PhoneBook\_CreatePerson, L("CreateNewPerson"), multiTenancySides: MultiTenancySides.Tenant);

Sử dụng quyền

[AbpAuthorize(AppPermissions.Pages\_Tenant\_PhoneBook\_CreatePerson)]

public async Task CreatePerson(CreatePersonInput input)

{

//...

}

Thêm bảng mới liên hệ với bảng cũ.

Làm quá trình tạo bảng Phones, và trong bảng PbPersons thì có thêm dòng sau

[Table("PbPersons")]

public class Person : FullAuditedEntity

{

//...other properties

public virtual ICollection<Phone> Phones { get; set; }

}

[AutoMapFrom(typeof(Person))]

public class PersonListDto : FullAuditedEntityDto

{

public string Name { get; set; }

public string Surname { get; set; }

public string EmailAddress { get; set; }

public Collection<PhoneInPersonListDto> Phones { get; set; }

}

[AutoMapFrom(typeof(Phone))]

public class PhoneInPersonListDto : CreationAuditedEntityDto<long>

{

public PhoneType Type { get; set; }

public string Number { get; set; }

}

Trong method GetPeople ta có

public ListResultDto<PersonListDto> GetPeople(GetPeopleInput input)

{

var persons = \_personRepository

.GetAll()

.Include(p => p.Phones)

.WhereIf(

!input.Filter.IsNullOrEmpty(),

p => p.Name.Contains(input.Filter) ||

p.Surname.Contains(input.Filter) ||

p.EmailAddress.Contains(input.Filter)

)

.OrderBy(p => p.Name)

.ThenBy(p => p.Surname)

.ToList();

return new ListResultDto<PersonListDto>(ObjectMapper.MapTo<List<PersonListDto>>(persons));

}