**TẠO WEBSITE SỬ DỤNG ASP NET CORE 6.0 & SQL SERVER**

1. **Tạo cấu trúc project website**

* src
  + application
    - CafeDevCode.Website (ASP Net Core MVC)
  + database
    - CafeDevCode.Database (Class Library)
    - CafeDevCode.Database.Shared (Class Library)
  + business
    - CafeDevCode.Logic (Class Library)
    - CafeDevCode.Logic.Shared (Class Library)
  + common
    - CafeDevCode.Utils (Class Library)
    - CafeDevCode.Common.Shared (Class Library)

1. **Khởi tạo Entities**

* Xây dựng phần database cho website
* Khởi tạo BaseEntity chứa các thời gian chỉnh sửa và cập nhật.
  + CafeDevCode.Common.Shared
    - Model
      * BaseEntity.cs
* public class BaseEntity
* {
* public string? CreateBy { get; set; }
* public DateTime? CreateAt { get; set; }
* public string? LastUpdateBy { get; set; }
* public DateTime? LastUpdateAt { get; set; }
* public bool IsDeleted { get; set; }
* public BaseEntity SetCreateInfo(string user, DateTime date)
* {
* this.CreateBy = user;
* this.CreateAt = date;
* return this;
* }
* public BaseEntity SetUpdateInfo(string user, DateTime date)
* {
* this.LastUpdateBy = user;
* this.LastUpdateAt = date;
* return this;
* }
* }
* Khai báo, khởi tạo các Entities và Configuration
  + CafeDevCode.Database
    - Configuration
    - Entities
* Thêm các nuget liên quan (Entity Framework Core & AspNetCore Identity)
  + Microsoft.EntityFrameworkCore
  + Microsoft.EntityFrameworkCore.Tools
  + Microsoft.EntityFrameworkCore.Design
  + Microsoft.EntityFrameworkCore.SqlServer
  + Microsoft.AspNetCore.Identity
  + Microsoft.AspNetCore.Identity.Stores
  + Microsoft.AspNetCore.Identity.EntityFrameworkCore

1. **Khởi tạo AppDatabase DbContext và Migrate**

* Thêm AppDatabase.cs trong CafeDevCode.Database
* public class AppDatabase : IdentityDbContext<User>
* {
* public AppDatabase()
* {
* }
* public AppDatabase(DbContextOptions<AppDatabase> options)
* : base(options) { }
* protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
* {
* optionsBuilder.UseSqlServer();
* }
* public virtual DbSet<Author> Authors => Set<Author>();
* public virtual DbSet<Category> Categories => Set<Category>();
* public virtual DbSet<Comment> Comments => Set<Comment>();
* public virtual DbSet<PlayList> PlayLists => Set<PlayList>();
* public virtual DbSet<PlayListVideo> PlayListVideos => Set<PlayListVideo>();
* public virtual DbSet<Post> Posts => Set<Post>();
* public virtual DbSet<PostCategory> PostCategories => Set<PostCategory>();
* public virtual DbSet<PostRelated> PostRelateds => Set<PostRelated>();
* public virtual DbSet<PostTag> PostTags => Set<PostTag>();
* public virtual DbSet<PostType> PostTypes => Set<PostType>();
* public virtual DbSet<SitePage> SitePages => Set<SitePage>();
* public virtual DbSet<Tag> Tags => Set<Tag>();
* public virtual DbSet<Video> Videos => Set<Video>();
* }
* Migrate database
  + Add-Migration InitialDatabase
* Khởi tạo dữ liệu mặc định Seeder
  + CafeDevCode.Database
    - Seeder
      * AppSeeder.cs
* public class ApppSeeder
* {
* public static async Task InitializeAsync(AppDatabase database,
* UserManager<User> user, RoleManager<IdentityRole> role)
* {
* database.Database.EnsureCreated();
* if (!role.Roles.Any())
* {
* await role.CreateAsync(new IdentityRole
* {
* Id = Guid.NewGuid().ToString(),
* Name = "Admin",
* NormalizedName = "ADMIN",
* });
* await role.CreateAsync(new IdentityRole
* {
* Id = Guid.NewGuid().ToString(),
* Name = "user",
* NormalizedName = "USER",
* });
* }
* if (!user.Users.Any())
* {
* var result = await user.CreateAsync(new User
* {
* Id = Guid.NewGuid().ToString(),
* UserName = "admin",
* NormalizedUserName = "Admin",
* Email = "admin@gmail.com",
* LockoutEnabled = false,
* }, "Admin@123");
* if (result.Succeeded)
* {
* var userResult = await user.FindByNameAsync("admin");
* await user.AddToRoleAsync(userResult, "Admin");
* }
* }
* }
* }

1. **Xây dựng cấu trúc tầng business**

* CafeDevCode.Logic
  + Commands
    - Data
    - Handler
  + Event
    - Data
    - Handler
  + Queries
    - Implement
    - Interface

1. **Cắt layout Admin (SbAdmin)**

* Sử dụng layout admin mẫu từ SbAdmin để làm giao diện quản trị.
  + Tải layout mẫu, thêm các resource vào project (css, js, assets…)
  + Khởi tạo layout Admin từ index.html của SbAdmin
    - CafeDevCode.Website/Views/Shared/\_AdminLayout.cshtml
    - CafeDevCode.Website/Views/Shared/\_AdminAuthenticateLayout.cshtml
    - CafeDevCode.Website/Controllers/HomeController.cs
    - CafeDevCode.Website/Views/Shared/Home/Index.cshtml
    - CafeDevCode.Website/Views/Shared/Partial/AdminLeftMenu.cshtml

1. **Xây dựng tầng business**

* Khởi tạo common sử dụng chung
  + CafeDevCode.Common.Shared/Model/BaseQuery.cs
* public class BaseQuery
* {
* public string? Keywords { get; set; } = string.Empty;
* public int? PageSize { get; set; }
* public int? PageIndex { get; set; }
* }
  + CafeDevCode.Common.Shared/Model/BasePagingData.cs
* public class BasePagingData<T>
* {
* public List<T> Items { get; set; } = new List<T>();
* public int PageIndex { get; set; }
* public int PageSize { get; set; }
* public int TotalItem { get; set; }
* public int TotalPage { get; set; }
* }
  + CafeDevCode.Common.Shared/Model/BaseCommandResult.cs
* public class BaseCommandResult
* {
* public bool Success { get; set; }
* public string Messages { get; set; } = string.Empty;
* public List<BaseError> Errors { get; set; } = new List<BaseError>();
* }
  + CafeDevCode.Common.Shared/Model/BaseCommandResultWithData.cs
* public class BaseCommandResultWithData<T> : BaseCommandResult
* {
* public T? Data { get; set; }
* }
  + CafeDevCode.Common.Shared/Model/BaseError.cs
* public class BaseError
* {
* public string? Code { get; set; } = string.Empty;
* public string? RelatedProperties { get; set; } = string.Empty;
* public string? Message { get; set; } = string.Empty;
* }
* Thêm thư viện:
  + AutoMapper
  + AutoMapper.Extensions.Microsoft.DependencyInjection
  + MediatR
  + MediatR.Extensions.Microsoft.DependencyInjection
* Xây dựng models tương ứng trong CafeDevCode.Logic.Shared
  + Detail Model
  + Summary Model
* Xây dựng Interface và Implement (business/CafeDevCode.Logic/Queries)
* Xây dựng commands request và handler tương ứng
  + Create command request
* public class CreateAuthor : Author,
* IIdentifiedCommand,
* IRequest<BaseCommandResultWithData<Author>>
* {
* public string? RequestId { get; set; }
* public string? IpAddress { get; set; }
* public string? UserName { get; set; }
* }
  + Create command handler
* public class CreateAuthorHandler : IRequestHandler<CreateAuthor, BaseCommandResultWithData<Author>>
* {
* private readonly IMapper mapper;
* private readonly AppDatabase database;
* public CreateAuthorHandler(IMapper mapper,
* AppDatabase database)
* {
* this.mapper = mapper;
* this.database = database;
* }
* public Task<BaseCommandResultWithData<Author>> Handle(CreateAuthor request,
* CancellationToken cancellationToken)
* {
* var result = new BaseCommandResultWithData<Author>();
* try
* {
* var author = mapper.Map<Author>(request);
* author.SetCreateInfo(request.UserName ?? string.Empty, AppGlobal.SysDateTime);
* database.Authors.Add(author);
* database.SaveChanges();
* result.Success = true;
* result.Data = author;
* }
* catch (Exception ex)
* {
* result.Messages = ex.Message;
* }
* return Task.FromResult(result);
* }
* }
  + Update command request
* public class UpdateAuthor : Author,
* IIdentifiedCommand,
* IRequest<BaseCommandResultWithData<Author>>
* {
* public string? RequestId { get; set; }
* public string? IpAddress { get; set; }
* public string? UserName { get; set; }
* }
  + Update command handler
* public class UpdateAuthorHandler : IRequestHandler<UpdateAuthor, BaseCommandResultWithData<Author>>
* {
* private readonly AppDatabase database;
* private readonly IMapper mapper;
* public UpdateAuthorHandler(AppDatabase database,
* IMapper mapper)
* {
* this.database = database;
* this.mapper = mapper;
* }
* public Task<BaseCommandResultWithData<Author>> Handle(UpdateAuthor request,
* CancellationToken cancellationToken)
* {
* var result = new BaseCommandResultWithData<Author>();
* try
* {
* var author = database.Authors.FirstOrDefault(x => x.Id == request.Id);
* if (author != null)
* {
* mapper.Map(request, author);
* author.SetUpdateInfo(request.UserName ?? string.Empty, AppGlobal.SysDateTime);
* database.Authors.Update(author);
* database.SaveChanges();
* result.Success = true;
* result.Data = author;
* }
* else
* {
* result.Messages = $"Can't find author ID is {request.Id}";
* }
* }
* catch (Exception ex)
* {
* result.Messages = ex.Message;
* }
* return Task.FromResult(result);
* }
* }
  + Delete command request
* public class DeleteAuthor :
* IIdentifiedCommand,
* IRequest<BaseCommandResult>
* {
* public int Id { get; set; }
* public string? RequestId { get; set; }
* public string? IpAddress { get; set; }
* public string? UserName { get; set; }
* }
  + Delete command handler
* public class DeleteAuthorHandler : IRequestHandler<DeleteAuthor, BaseCommandResult>
* {
* private readonly AppDatabase database;
* private readonly IMapper mapper;
* public DeleteAuthorHandler(AppDatabase database,
* IMapper mapper)
* {
* this.database = database;
* this.mapper = mapper;
* }
* public Task<BaseCommandResult> Handle(DeleteAuthor request,
* CancellationToken cancellationToken)
* {
* var result = new BaseCommandResult();
* try
* {
* var author = database.Authors.FirstOrDefault(x => x.Id == request.Id);
* if (author != null)
* {
* author.MaskDelete(request.UserName ?? string.Empty, AppGlobal.SysDateTime);
* database.Authors.Update(author);
* database.SaveChanges();
* result.Success = true;
* }
* else
* {
* result.Messages = $"Can't find author ID is {request.Id}";
* }
* }
* catch (Exception ex)
* {
* result.Messages = ex.Message;
* }
* return Task.FromResult(result);
* }
* }

*(Tương tự với các object khác)*

* Khởi tạo MappingProfile (CafeDevCode.Logic/MappingProfile)
* public AuthorMappingProfile()
* {
* CreateMap<CreateAuthor, Author>();
* CreateMap<UpdateAuthor, Author>();
* CreateMap<Author, AuthorSummaryModel>()
* .ReverseMap();
* CreateMap<Author, AuthorDetailModel>()
* .ReverseMap();
* }
* Khởi tạo Global (CafeDevCode.Utils/Global/AppGlobal.cs)
  + Get DateTime
* public class AppGlobal
* {
* public static DateTime SystemDateTime => DateTime.Now;
* }

1. **Xây dựng login handler, xây dựng trang đăng nhập**