1. Entity Framwork Core

EF core là một Object-relational mapper, truy xuất DB bằng LINQ và hỗ trợ lưu bền vững data vào DB.

Thêm nugetpackage:

- dotnet add package Microsoft.EntityFrameworkCore.SqlServer
- dotnet add package Microsoft.EntityFrameworkCore.InMemory

2. The DBContext

Để làm việc với DBContext, ta cần phải inherite DBContext. Class này chứa các properties đại diện cho tập hợp các entity của application.

DBContext tự động bị disposed bởi garbage collection. Hoặc sử dụng using/try-finally.

Trong Web application, hãy sử dụng một DBContext instance per request

Nếu context instance được tạo bởi DI container, thường thì container có nhiệm vụ disposed context instance.

Context không hỗ trợ thread-safe. Context ngốn ram nếu sử dụng lâu dài.

Context đóng mở connection mỗi một query, có thể gây ra vấn đề hiệu năng, xem xét sử dụng Connection property.

DBContext phải có constructor dùng DbContextOptions là argument. DBContextOptions dùng cho việc Configuring EF Core dưới đây

3. Configuring EF Core

Thêm DBcontext Service trong ứng dụng kết nối MSSQL:

```
builder.Services.AddDbContext<DBContextSubClass>(
        options => options.UseSqlServer(
                builder.Configuration.GetConnectionString("DefaultConnection")));
```

Thêm DBcontext Service trong ứng dụng sử dụng in-memory database:

builder.Services.AddDbContext< DBContextSubClass >(

options => options.UseInMemoryDatabase());

4. Fetching and Storing Data

Các câu lệnh như: **Where, OrderBy, Select, Include**, … đến cuối cùng sẽ tạo thành một IQueryable object. Câu lệnh này chỉ được thực thi khi nó được gọi bởi các lệnh như: **ToListAsync, FirstOrDefault, …**

EF core track (theo dõi) các thay đổi của các entities mà nó lấy từ persistence (bộ nhớ bền vững). Để save các thay đổi của entities, gọi lệnh **SaveChangesAsync** trong DBContext. Thêm hoặc Xóa các entities chỉ được thực hiện trên DbSet property, để lưu bền vững cần them **SaveChangesAsync**.

Một ví dụ về EF core LINQ:

```
// read
var brandItems = await _context.CatalogBrands
    .Where(b => b.Enabled)
    .OrderBy(b => b.Name)
    .Select(b => new SelectListItem {
        Value = b.Id, Text = b.Name })
    .ToListAsync();

// create
var newBrand = new CatalogBrand() { Brand = "Acme" };
_context.Add(newBrand);
await _context.SaveChangesAsync();

// read and update
var existingBrand = _context.CatalogBrands.Find(1);
existingBrand.Brand = "Updated Brand";
await _context.SaveChangesAsync();

// read and delete (alternate Find syntax)
var brandToDelete = _context.Find<CatalogBrand>(2);
_context.CatalogBrands.Remove(brandToDelete);
await _context.SaveChangesAsync();
```

5. Explicit Loading, Lazy Loading and Eager Loading

Khi EF core truy xuất entities, nó tải tất cả properties của entity đó, còn các relational properties sẽ được set thành null.

Eager Loading: Load relation properties cùng với entities (sử dụng Include)

Cần tránh hai loại loading sau trong web application vì nó sử dụng nhiều round trip tới db hơn.

Lazy Loading: Load entities. Và khi ứng dụng cần thì load thêm các relational propeties

Explicit Loading: khá giống lazy loading nhưng lazyload sẽ không load tự động như lazy loading, nó cần gọi Load method.