Asp.NET Core

ViewModel & DTO Yapılanması

AutoMapper Library

ViewModel Nedir?

- ViewModel, temelde iki farklı senaryoya karşılık sorumluluk üstlenen ve biz yazılım geliştiricilerin işini kolaylaştıran operasyonel nesnelerdir.
  - 1. Senaryo

    OOP yapılanmasında bir modelin kullanıcıyla etkileşimi neticesinde kullanılan ve esas datanın memberlarını temsil eden ve süreçte ilgili model yerine veri taşıma/transfer operasyonunu üstlenen bir nesnedir.
  - 2. Senaryo

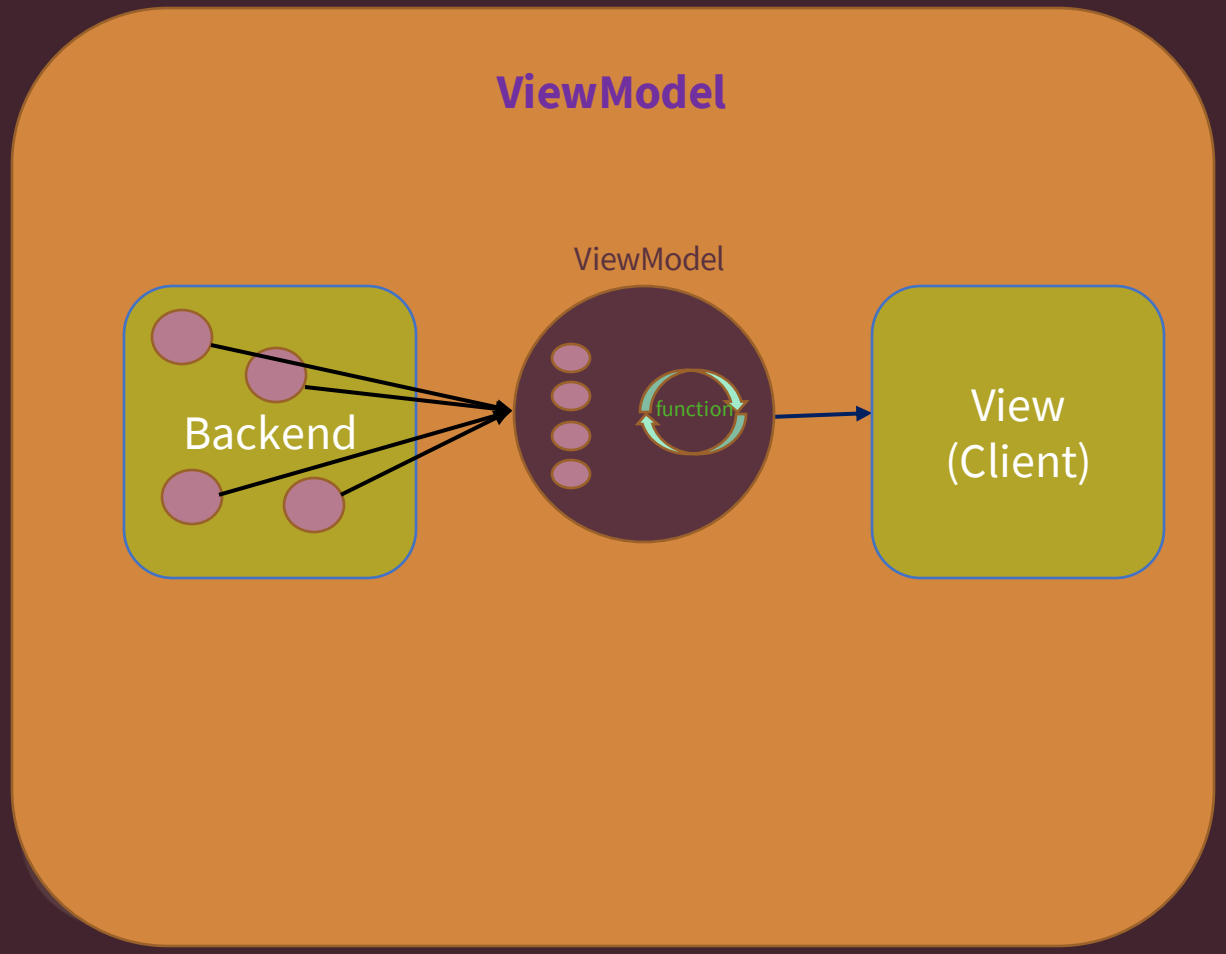
    Birden fazla modeli/değeri/veriyi tek bir nesne üzerinde birleştirme görevi gören nesnedir.

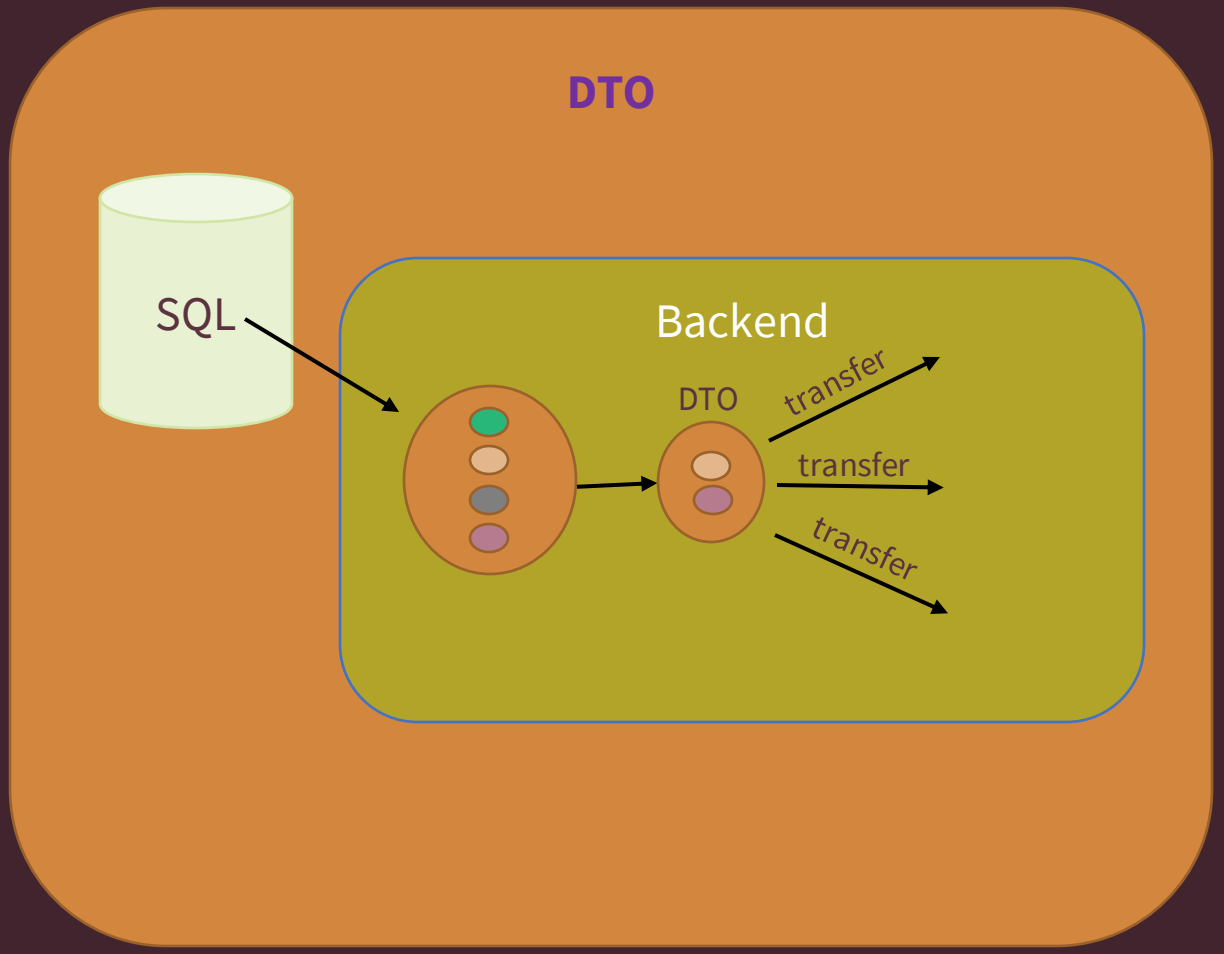
DTO Nedir?

- DTO(Data Transfer Object)
- Herhangi bir davranışı olmayan ve uygulamanın çeşitli yerlerinde yalnızca bir veri tüketimi ve iletimi için kullanılan, veritabanındaki herhangi bir verinin transfer nesnesidir/karşılığıdır/görünümüdür.

ViewModel	DTO
• Kullanıcıya sunulacak verinin view'e uygun/view'in beklediği şekilde tasarlanmış modelidir. • Veriyi görünüm/sunum/presentation için anlamlı hale getirir. • İşlevsel fonksiyonlar(metot) barındırabilir. • İçerisinde bir veya birden fazla DTO temsil edebilir. • DTO'ya nazaran daha karmaşıktır.	• Bir verinin(genellikle veritabanından gelen verinin) transfer modellemesidir. Transfer edilecek olan ilgili verideki sadece ihtiyaç olunan dataları temsil eder. • Görünüm/sunum/presentation için kullanılabilir lakin bunun dışında uygulamanın herhangi bir katmanında çeşitli veri tüketimi ve transferi içinde kullanılmaktadır. • Herhangi bir fonksiyonellik barındırmaz. • Salt veriyi temsil eder.

Temel Amaç


ViewModel
ViewModel
Backend
function
View
(Client)


DTO
SQL
Backend
DTO
transfer
transfer
transfer

Senaryo 1

Bir Model'in View'de ki Etkileşimine Uygun Parçasını Temsil Etme

Kullanıcıya sunulan hiçbir veri direkt olarak
veritabanındaki entity türünden olmamalıdır.
Bu tarz durumlarda ViewModel kullanılmalıdır.

Senaryo 2

Birden Fazla Nesneyi Tek Bir Nesneye Bağlama

Sözleşme/Kontrat Mantığı Nedir?

- Backend'de üretilen bir verinin client'a gönderilmesi için tasarlanan ViewModel o işlemin sözleşmesi/kontratı olmaktadır.
- Haliyle Backend'den gelecek datayı client'ın uygun formatta karşılayabilmesi için kesinlikle o türden bir nesne oluşturması gerekecektir.

ViewModel'lar da Validation Durumları

- Kullanıcıdan alınan veriler iş kuralı gereği kontrol edilirler. Bizler bu kontrollere validation diyoruz.

Kullanıcılardan gelen veriler kesinlikle veritabanı tablolarının karşılığı olan entity modelleri olmamalıdır! ViewModel olarak alınmalıdır! Ve tüm validation'lar bu ViewModel nesneleri üzerinde gerçekleştirilmelidir.

ViewModel'ı Entity Model'e Dönüştürme

- Kullanıcıdan gelen dataları ViewModel ile karşıladıktan sonra bu ViewModel'da ki verileri veritabanına kaydetmek isteyebiliriz. Bu durumda, bu verileri Entity Model'a dönüştürmemiz gerekecektir. Bunun için aşağıdaki yöntemlerden herhangi biri kullanılabilir:
    - Manuel Dönüştürme
    - Implicit Operator Overload İle Dönüştürme
    - Explicit Operator Overload İle Dönüştürme
    - Reflection İle Dönüştürme
    - AutoMapper Kütüphanesi İle Dönüştürme