- All the Entities are called model, now if we don’t wanna interact with this directly, we must find a way to interact with its abstraction. That’s when ViewModel comes into play.

- Then we have to convert from ViewModel to Model before passing to repository 🡺 AutoMapper

- dbo.AspNetUsers: Danh sach nguoi dung

- dbo.AspNetRoles: Phan quyen

- dbo.AspNetUserRoles: Mapping giua 2 cai tren (m:n) 🡺 New table

Chap 7: A master refactor changing from Identity User to Employee just to satisfy a single mapping line code on Leave Allocation control. Also this chapter allows us to learn about refactoring in C# when business requirement

Chap 11:

Ví dụ, trong MVC có code, ta thay đổi Entities như các trường, ta sẽ phải sửa cả controller với những trường thêm này

🡺 MVVM: ta bind thẳng vào ViewModel, chứ không thông qua Model (Entity) nữa, không còn phải bind các field vào trong từng hàm của Controller

🡺 ViewModel là trung gian

Repository Pattern:

Nếu như để Data Access Code ở trong Controller, mỗi khi có chỉnh sửa yêu cầu như hiển thị ra danh sách thỏa mãn các tiêu chí, ta sẽ phải copy code query này (Where(…)) khắp các nơi 🡺 Repository Pattern tạo ra các repo có các hàm, sau này gọi đến các hàm ở nhiều nơi thì ta chỉ cần chỉnh sửa 1 nơi là hàm gốc trong Repo là áp dụng thay đổi lên tất cả mọi nơi một cách đồng nhất

🡺 Modify theo yêu cầu nghiệp vụ thì chỉ cần 1 nơi, đó là function trong repo, để controller chỉ gọi function này thôi, chứ không áp thẳng code vào controller nữa

UOW

Repository Pattern sẽ trở nên phiền phức khi ta cần nhiều tham chiếu đến nhiều repository trong một controller, mỗi lần ta cần thì lại phải add thêm một repository vào controller 🡺 UOW: Một base repo tổng quan hơn nữa, một repo thực thi chứa đầy đủ tất cả các table ta cần truy cập luôn.

🡺 Bớt phiền toái khi ném cả đống repo vào từng controller chỉ vì một function cần đến nó.