MVVM - Model-View-ViewModel

1. Po co w folderze entities mamy klasy POCO np. Customer, a następnie jeszcze Customer.ViewModels?

Na przykład Blip.Entities.Customer nie zawiera Blip.Entities.Customer.ViewModels. Zobaczymy to w akcji, gdy spojrzymy na metody kontrolera w projekcie Blip.Web. Działania kontrolera będą miały dostęp do modeli widoku, ale nie do samych bytów. Zapobiega to dokonywaniu przez błędne działanie kontrolera zmian bezpośrednio w jednostce, a tym samym bezpośrednio w pamięci trwałej. Może to mieć istotne zalety w zakresie bezpieczeństwa.

1. Kontrolki danych z klasy modelu w MVC. Jak powiedzieć że prop id to właśnie id, albo że dana jest wymagana itp. do utworzenie obiktu modelu, który potem będzie rzutowany na DB, wiec musi mieć te atrybuty/wymagania szzególne.

Zapewnia to klasa Systemowa .net : System.ComponentModel.DataAnnotations

[Key]

[Column(Order = 1)]

public Guid CustomerID { get; set; }

1. **Co sprawia, że nasze rozwiązanie Model-View-ViewModel (MVVM) jest pragmatyczne?**

To że zarówno jednostki podstawowe, jak i modele widoków są dostępne w warstwie internetowej, więc

* możliwe jest utrzymanie danych w formularzach internetowych dla akcji CRUD bezpośrednio przy użyciu klasy podstawowej encji.

W projektach, w których istnieje znaczna liczba tabel baz danych zawierających referencje lub dane „sprawdzające”

* może to wyeliminować wiele powtarzających się kodów, które nie służą przekonującym celom.

1. Jak to u mnie wygląda:

| Project | Components |
| --- | --- |
| Web | Controllers, Views |
| Entities | Models, View Models |
| Data | Data Context, Repositories |
| Common | Utility functions, Enumerations |

1. W projekcie w którym chcielibyśmy oddzielić DB od web przechodząc przez view model w akcjach kontrolera, to wtedy web miałby ref do data, ale do entities już nie (do bazowych klas modelu, do ich propert)
2. To w jakim stopniu to moje rozw. zabezpiecza dane? Odp. sobie gadając z Szymonem. Ale odp. jest gdzieś powyżej, że przez to ze mamy viewmodel w innym folderze to i tak nie działamy bezpośrednio na bytach? Jak to zobaczyć naocznie? Przetestować czy faktycznie tak jest?.

Odp.:

W klasie kontrolera odwołujemy się tylko do using Blip.Entities.Customer.ViewModels;

więc nie musi wiedzieć o samych entities ani o innych view models niż te związane z obiektem Customer.

1. Pojedynczy projekt solucji, czy multi = oba bazują na repository wzorcu projektowym, jedyna różnica jest w ref między projektami, przez co namespaces.