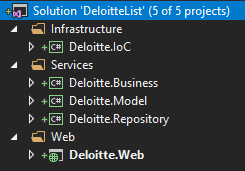
* Why certain approaches were used, why others were not selected:

Not exactly but I just separated by items checked and not checked

* Any design patterns used:

This is the pattern I use to work, separate by:

* Infrastructure
  + DI – Dependency Injection
* Services
  + Business – Business rules
  + Model – All ViewModels, Enum
  + Repository – Comunication with DB
* Web
  + Front-end



* Anything extra you would have done given more time:

First of all, sorry I delayed so much already to delivery it. But I hope you like, but I was wondering if it would be my daily task I would add these items below:

* Validations on front end (JS)
* OWIN Authentication instead passing user through url
* Logger in each class
* Mapper class to map DB and ViewModel
* Good comments in each method

And I think I should try something using AngularJS + Api to be faster and easier to build the front end and also a couple of unit test to make sure it’ll work before we release it to live

* Anything else you feel we should know:

I think it was a good experience to build something like that, using DB in memory, we also use CacheMemory for a couple of static information in our internal applications (Clubtravel) to be a faster application.