Tre lags applikation

- Data access layer
- Business layer
- Presentation
- Evt. web service api

Simpelt eksempel (disclaimer)

- Afhængighederne nedad i lagene bryder **Dependency Inversion Principle** (DIP), en del af SOLID principperne
- Det kan diskuteres, om modellen fra data access layer må boble op i business layer.
- For et andet eksempel, se f.eks. https://medium.com/aspnetrun/layered-architecture-with-asp-net-core-entity-framework-core-and-razor-pages-53a54c4028e3

Data access

- Tilgår databasen.
- I vores eksempel vha. Entity Framework
- Udstiller metoder til at tilgå databasen
- Gøres i dette eksempel vha repository-klasser med static metoder. Dette burde gøres med et interface og en factory til at levere implementation (eller et dependency injection framework), men dette er fravalgt aht simplicitet

Problemer

- Senere, når vi kører applikationen, vil vi opdage, at app.config tilsyneladende ikke bliver læst fra datalaget. I hvert fald anvendes entity framework oplysningerne ikke.
- Dette skyldes, at konfiguration normalt lægges ved den startende applikation (altså presentation laget)

Business layer

- Tilgår Data access layer
- Tilføjer f.eks. validering, mere kompleks håndtering, kombination af flere resultater fra databasen
- Håndterer formodentlig ikke transaktioner, dvs komplekse opdateringer af databasen sker i data access layer.
  Alternativt skulle business layer kunne styre transaktioner via metoder i data access layer. Kig f.eks. på UnitOfWork pattern.

Presentation layer

- Presentation henviser selvfølgelig til brugergrænseflade, men under dette punkt burde også være f.eks. web services.
- Der kan godt være flere presentations, f.eks. en WPF, en Web (ASP.NET MVC), en(flere) webservice(s)

Problem: EF-referencer

- Filen EntityFramework.SqlServer.dll bliver ikke kopieret til de højere liggende lags bin-folder, når man builder. EntityFramework.dll bliver godt nok kopieret.
- Det er et emne til diskussion, om dette er en fejl eller ej.

Problem: EF-referencer

- 3 mulige løsninger:
  - I det øverste projekt (presentation) laves en reference til denne fil
    Den kan f.eks. findes under
    <Solution-folder>\packages
    \EntityFramework.6.4.4\lib\net45\EntityFramework.SqlServer.dll
  - I det øverste projekt installeres Entity framework i NuGet manager.
  - I data laget laves et kald til funktionalitet i dette assembly, f.eks.:

```
private void dummy()
{
  string result =   System.Data.Entity.SqlServer.SqlFunctions.Char(65);
}
```

  metoden skal ikke kaldes.

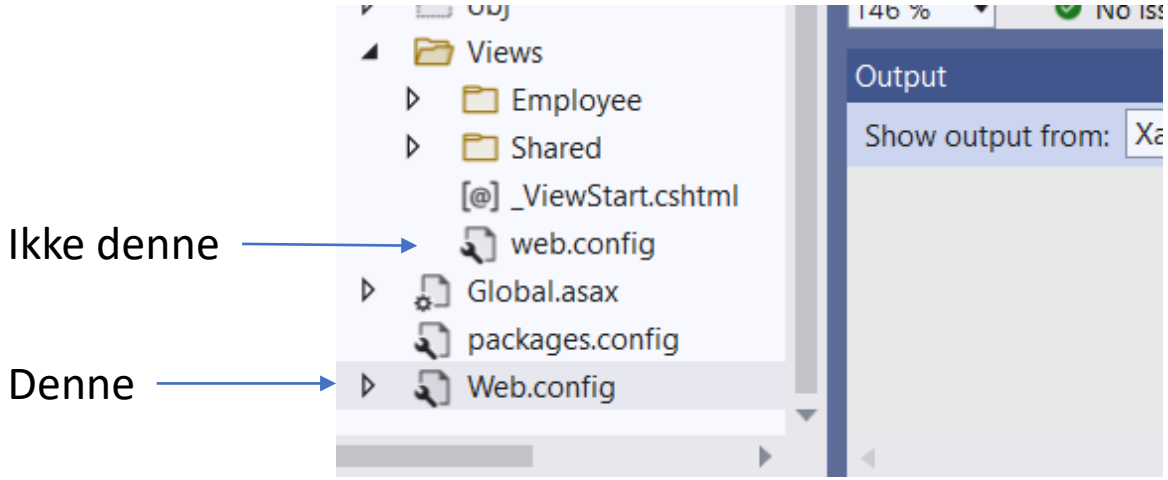
Alle tre løsninger er ret grimme, det står frit for at vælge en.

Problem: konfiguration

- Det er naturligt, og fristende, at lægge konfigurationen af databaseadgang i dataaccess laget. Det drejer sig om
  - `<section name="entityFramework"`
  - hele `<entityFramework>` delen
  - `<connectionStrings>` delen
- Det er imidlertid sådan, at det er det assembly, eksekveringen er startet fra, der skal konfigureres (hvis man vil gøre det automatisk, det kan også styres)
- Dvs. hvis man har lavet flere assemblies (som vi har), og hele applikationen startes fra en WPF-applikation (som vi gør indtil videre), er det dennes app.config, der skal indeholde denne konfiguration

Problem: konfiguration

- Hvis det startes via en MVC webapplikation (som vi gør senere) skal det konfigureres i dennes web.config.
- Dem er der desværre to af. Brug den, der ligger i roden af applikationen:



Problem: konfiguration

- Så det meste af indholdet fra app.config i data access laget skal flyttes til en anden konfigurationsfil.