## Software arkitektur 3 lags modellen

Da arkitekturen af projektet skulle besluttes, som skulle stå for at få opbygget et system, som ville give et godt og hurtigt overblik over systemet, blev der valgt at bruge lag delte model. Grunden til at valget faldt på den var at ved den lag delte model er der en grovkornet gruppering af klasser og pakker. Hvorved at de klasser der befinder sig i ét lag har et sammenhængende ansvar et vigtigt aspekt i systemet i den logiske separation der er lavet i projektet. Ved at gruppere systemet på denne måde, kan der hurtigt findes ud af hvor der skal ændres ting hvis der er noget der skal ændres. Organiseringen af lagene, er således at et højere lag må kalde et lavere lag, men et lavere lag må ikke kalde et højere lag. Den lag delte model der er valgt at tage udgangspunkt i er den 3 lag delte model. Der består af, DAL (Data access layer), BL (Buisness layer) og PL (Presentation layer). Et eksempel på hvordan et normalt scenarie kunne se ud, kan ses på Figur 1, som viser hvad der sker når en bruger indtaster i et felt der kan autofuldføre. Først ændre brugeren på teksten på GUI, derefter kalder GUI’en ned på GUI viewmodel at den skal udfylde listen, den kalder så ned på BL, som derefter kalder ned på DAL, at den skal lave et database udtræk, ud fra den indtastede tekst.



Figur 1: Sekvens diagram for Autofuldførelse, med lag opdeling

Hver af disse lag kan så bestå af flere klasser og pakker, som det kan ses på FIGURE, som viser et package diagram for systemet.

FIGURE INDSÆT OPDATERET PACKAGE DIAGRAM

Fordele ved at der er valgt at bruge en lag delt model; SRP (Single Responsibility Principle), som fås ved at der kommer en separation af applikations specifikke egenskaber fra generelle egenskaber. Derudover kommer der en separation af højniveaus handlinger fra lavniveaus. Koblingen og afhængighederne formindskes deraf mellem de forskellige klasser, samt der kommer høj samhørighed. En af de vigtigste ting ved at bruge 3 lags modellen er dog at der er muligheden for at genbruge kode, sådan at man for eksempelvis kan genbruge de nederste lag af kode til en applikation med en anden brugergrænseflade. Det gøres endnu mere simpelt ved at de forskellige lag er implementeret med interfaces, som simplificere en ændring eller udskiftning af dele af systemet. Ved at lave den logiske segmentering mellem de forskellige lag er klarheden af koden øget, for andre der skulle ønske at arbejde videre med koden.