# Projektgennemførelse og metoder

## Projektstyring

Gruppen har valgt at tage elementer fra Scrum, således at gruppen styrer vores projekt med en fornuftig iterativ model.

Gruppen har kørt sprints på 1-3 uger. Herunder elementer gruppen har anvendt:

* Taskboard
  + Gruppen har brugt et digitalt scrumboard vha. en in-house webapplikation (Redmine) skolen har stillet til rådighed. Her er det muligt at se hvilke opgaver der skal laves under nuværende sprint, samt hvem de tilhører, og hvor meget tid der er sat af til opgaven.
* Stå-op-møde
  + Mandag, onsdag og fredag har gruppen holdt et møde på 5-10 minutter. Her kommer hvert gruppemedlem ind på 3 ting; Hvad har du lavet? Hvad skal du lave? Hvilke forhindringer har du? Det har gavnet gruppen, da man hurtigt får feedback fra hvert medlem og hører om de er gået i stå, eller om der er nogen der mangler tasks.
* Retrospekt-møde
  + Under retrospekt-mødet har gruppen afsluttet nuværende sprint og planlagt næste. Der ses hvor meget gruppen nåede, og hvert gruppemedlem kommer med feedback til hvad der kan gøres bedre til næste sprint. Derefter planlægges et nyt sprint med omhu, således at opgaverne og den nominerede tid lyder realistisk ift. forhindringer.

Kort beskrivelse af hvert sprint:

|  |  |  |
| --- | --- | --- |
| **Iterationsnummer** | **Længde** | **Formål** |
| 1 | 4 dage | Kravspecifikation |
| 2 | 2 uger | De første kernefunktionaliteter implementeres/påbegyndes |
| 3 | 2 uger | De sidste kernefunktionaliteter (det første use case-udkast gruppen lavede) laves færdigt. Nogle ting refaktoreres |
| 4 | 3 uger | Udvidelser (nye use cases) blev påbegyndt. Unittests blev sat op. Databasen integreres med applikationen |
| 5 | 2 dage | Gruppen valgte at gennemføre et minisprint for at fastlægge hvad gruppen skulle lave i næste sprint. Her undersøgte gruppen blandt andet om der kunne nåes at refaktorere WPF-applikationen til at bruge MVVM eller MVC-pattern |
| 6 | 1 uge | Under dette sprint blev synkroniseringsprocessen og webapplikationen påbegyndt |
| 7 | 1 uge | Her blev Sync lavet færdig, og der blev arbejdet videre på webapplikationen |
| 8 | 1 uge | Notifikations-use casen blev lavet færdig og Sync blev koblet på WPF-applikationen. Controller og Model blev arbejdet på i webapplikationen |
| 9 | 1 uge | Meget rapportskrivning. De enkelte use cases blev uddelegeret i webapplikationen |

## UML

Til formidling af kravspecifikation og systemarkitektur, har projektgruppen valgt at anvende UML (Unified Modeling Language). Dette er valgt, for at formidle systemet bedst muligt, da UML er industristandard, simpel og intuitiv at gå til for omverdenen.

## Tidsplan

Gruppen har fulgt en tidsplan hvor der er forsøgt at sætte mere tid af til rapportskrivning end hvad vi normalt har afsat.

Se bilag xx.

## Mødestruktur

Gruppe- og vejledermøder er blevet styret ved hjælp af en mødeindkaldelse, efterfulgt af et møde med dagsorden, dirigent og referent. De administrative roller er blevet fastlagt vha. en turnusordning , hvor de forskellige roller som mødeindkalder, referent og dirigent skifter fra møde til møde. Dette er gjort for at sikre at alle gruppemedlemmer får et indblik i det administrative arbejde. For at sikre konsensus i dokumenter, er der udarbejdet skabeloner til mødeindkaldelser og referater. Møderne er blevet afholdt efter behov, med udgangspunkt i et møde ved et sprints begyndelse og afslutning. Referatet fra forrige møde er blevet gennemgået og godkendt ved hvert møde.

## Continuous Integration

Jenkins er i det følgende beskrevet, da på trods af at det ikke kom til at virke under dette projekt, er brugt mange timer på det og der er kommet en masse erfaringer ud af det.

Jenkins har haft til formål at sikre en objektiv og løbende håndtering af de tests, der er skrevet i projektet. Det kobles sammen med et git-repository og har derigennem adgang til den nyeste version af projektet. Jenkins builder selv projeket og kører de test-suites, der er lavet ved hjælp af NUnit- og NSubstitute frameworket. Det kan også sættes op til at udføre statisk analyse og coverage, hvis det ønskes.

I dette projekt er der blevet brugt lang tid på at få Jenkins til at fungere. Dette er dog ikke lykkes efter hensigten. Det git-repository, der er brugt i projektet kører på GitHub, hvilket viste sig at være et problem. Under opsætningen af Jenkins projektet skal der specificeres, hvornår Jenkins skal bygge den VisualStudio solution, der henvises til. Til omfanget af dette projekt, vil det være optimalt at bygge projektet hver gang der bliver lavet et nyt push. For at dette kan lade sig gøre med et git-repository, der kører på GitHub, skal der installeres et plugin på Jenkins-serveren. Det har ikke været muligt at finde en måde at gøre dette på, uden at prøve sig frem med opsætningen af et af de plug-ins der findes til nettop dette. Da projektgruppen ikke har direkte adgang til Jenkins serveren kunne det derfor ikke lade sig gøre.

Der blev senere udleveret et git-repository af skolen, som kører på Git-Swat og derfor ikke havde ovenstående problem. I forbindelse med udviklingen af projektets database tilgang, blev der brugt et modelling-projekt i Visual Studio. Denne type af projekt kræver at der er nogle bestemte filer i installationen af Visual Studio. Disse filer var ikke tilgængelige på Jenkins-serveren og Jenkins kunne derfor ikke bygge solution’en. Det blev forsøgt at omgå dette ved at ligge de manglende filer i en mappe i projektet, og importere dem i app.config (STUFF VAR DET HER DU FORSØGTE AT IMPORTERE DEM ?). Resultatet heraf blev at Jenkins kunne bygge projektet, men at det ikke kunne bygges lokalt i Visual Studio.

Da underviseren, der står for Jenkins lagde den manglende fil der, hvor Jenkins søgte efter den, kunne projektet bygges på Jenkins og lokalt. På dette tidspunkt var størstedelen af de ønskede test-suites skrevet og kørt, og grundlaget for at bruge Jenkins var ikke længere tilstrækkeligt.

Det ovenstående beskriver processen med Jenkins i forhold til WPF applikationen. Jenkins skulle efter planen også have været brugt til at teste web applikationen. Her manglede Jenkins-serveren ligesom ved modelling-projektet en fil, for at kunne bygge web-applikationer. Tidligere erfaringer viste, at det ikke kunne betale sig at forsøge at løse dette problem, uden selv at have adgang til serveren, og Jenkins blev derfor droppet.