LINKÖPINGS UNIVERSITET

TDP003 Projekt: Egna datormiljön

Reflektionsdokument

Författare

Arturas Aleksandrauskas, artal938@student.liu.se

INSTITUTIONEN FÖR DATAVETENSKAP

Höstterminen 2016
Version 1.0

2016-10-17

1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.0	Skapad dokumentet och byggt upp strukturen	161017

2 Reflektion

446 nesled Att sätta upp om hur vi skulle göra allt var inte lätt. Från början fram till idag är det mycket oklar vad som ska göra och hur det ska gå till! Nya krav dyker upp från ingenstan och olika specifikationer för arbetet finner man på hemsidan som säger emot varandra.
Då det har varit svårt att få en överblick av projektet så tog jag detta projekt på stort alvar. Vi kollade shemat och löst ett problem i taget, som till slut gav oss en produkt som vi kan precentera.

2.1 Arbets gången

2.1.1 Gitlab

Vi började med att sätta oss för att sätta upp Gitlab. För att båda ska ha tillgång till de senaste arbetet. Det är bra att bryta ner problem när man är två för då kan bålla med problemen 483-484 cc (gitlab är ett ända stort problem).

2.1.2 Photoshop

Deadline för LoFi var det första arbetet som skulle in. Vi satte oss ner tre dagar innan för att få en förståelse över vad som skulle göras. Efter att ha kollat i genom föreläsnings antäckningar, ritade vi upp vår mall på papper ut efter kraven och gjorde en snabb skiss på papper! Det var svårt att veta om vi har fått med allt, så Joakim skissade upp det i Photoshop och skickade in det. Joakim har tidigare erfarenhet av Photoshop så jag överlämnade finslipningen av LoFi till honom. Bortkastad tid att göra det tillsammans.483-484 cc

2.1.3 API

Joakim på eget hand sate upp en mall för hur api:t skulle se ut, villka funktioner som behövs och komentarer över vilken indata som kommer in och vilken ska ut. Tillsammans kom vi fram till att jag ska ta ansaret för api:t då jag har mer koll på python3 och Joakim har tidigare jobbat mycket med HTML och CSS så han skulle ansvara för precentations lager.
Då projektet är ganska olkart för oss är det bäst att vi ska jobba till våra styrkor. och genom att dela upp ansvaret så är det inte lika mycket jobb man måste lägga ner på projektet. Så länge man håller koll på sina ansvarsområden och är ärlig med hur man ligger till!
Jag tyckte att det funkade bra den uppdelningen. Vi jobbade 70% tillsammans sen delade vi upp de andra! ätt bra exempel är 11 oktober då jag kände att jag kan inte få ihop api:t med Flask. då fick jag stor hjälp av Joakim för att få en förståelse för metoder hur han använde flask och HTML vad han behöve för att det ska funka och hur det satt ihop. Så att tillsammans fick ihop api:t med flask och HTML. med den förståelsen kunde jag vid ett senare tillfälle bygga om vår intag av data från servern från POST till GET. even om jag inte byggde upp det så hade förståelse hur det funkade ihop.

2.1.4 Flask

Hela råa basen för Flask byggde Joakim upp. Det innebär tomma funktioner till alla sidor som han kände han behövde för att representera HTML han hade byggt upp med botstrap ut ifrån LoFi. Där efter satt vi tillsammans det som behövdes för att få ihop att data så att vi kunde använda rendet_template. Innan vi satte oss ihop och började jobba tilsamman med flask var jag ganska vilse i hur vi lå till. Efter 11 oktober har jag känt att jag har förståelse för alla dela i projektet. Jag känner mig tygg i mitt ansvarsområde men jag jag går gärna in ich fixa bugga om vi får problem med HTML och flask. tar bara lite längre tid. -Jag vill se en fördel och en nackdel med hur ni arbetat.

2.2 Hur delades arbetet upp

Områden	Ansvarig	Vem jobbade på det
API	Arturas	Arturas, Joakim
Data_test	Arturas	Arturas
LoFi	-	Arturas, Joakim
Photoshop	Joakim	Joakim
Flask	Joakim	Joakim,Arturas
HTML5	Joakim	Joakim, Arturas
CSS3	Joakim	Joakim

2.3 Resultatet av uppdelningen

Uppdelnin av arbetet gjorde att mängden tid som vi la ner på projektet var mycket liten och vi var effektiva. men det gav mycket ansvar till oss, att gå genom varandras arbete och hur grunden är uppbyggd, i CC nämner Steve McConnell det är onödigt att jobba ihop när man bygger upp grunden. För då blir det lätt att den ena kollar och den andra skriver, eller att man har onödiga diskutioner över hur upplägget ska vara. Det är bättere att bara jobba ihop vid svåra och mer komplexa problem! (Steven C. McConnell,2004:s483) Vid problemlösning satt vi och parprogramera och då var en i gruppen mycket insatt i systemet vi jobbade på och satt vid sidan om och den andra kodade och kom in med nytänkande idéer.

2.4 Erfarenheter

2.4.1 Parprogramera

Parprogramering kan vara väldigt kraftigt värktyg men det funkar inte för alla. Jag kan inte parprogramera som det är tänkt. Då jag har dyslexi tar det lång tid för mig att skriva, jag stavar fel och det går åt mycket tid för att rätta mina fel och det blir ingen miljö jag vill jobba i. Jag har inte problem att sitta vid sidan om och komma med input och att lösa problem. Som Steven C. McConnell nämen i CC *'Don't let pair programming turn into watching'*, då ha jag ett anvar att vara mycket på läst vid parprogramerings passen. Jag ser fördelar med parprogramering men bara vid utmanade moment.

2.4.2 Dela upp arbeter i mindre problem

För att få lättare förståelse för hela systemet delade vi upp det i mindre områden, samt områden vi jobbade delade vi upp i små del problem. Kravspecifikationen för API:t och Flask hjälpte mycket för att få en dra ADT men för förståelesen skull så skapade vi hjälpfunktionenr till de specifierade funktionerna för att få en bättre överblick och förståelse av systemet, som Steven C. McConnell rekomenderar (Steven C. McConnell,2004:s445).Ett exempel då Vi använde det är i search funktion i API:t.

2.4.3 lämna över det problemen till ansvargrupper

Att dela upp arbetet kändes som bra idé för att minska mängden arbete som måste genomföras per person. Det satte mig i en mer defensiv programeringstil (Steven C. McConnell,2004:186) då den data som kommer ut står jag ansarig för. Jag var tvungen att abstrahera funtionerna och vara noga med villken indata jag kunde få in och villken data som skulle lämnas over till Flask fårn API:t. Det kom med svårigheter då vi skulle kombinera det med parprogramering i vissa delmoment, för det tod tid för Joakin att sätta sig in i systemet. Samtidigt som det tid en bra stund för mig att förstå sig på Joakims arbete.

2.4.4 uppdelning av arbete ger mindre övergripande syn över systemet

Under under stora delar av projektet var jag orolig att vi ligger efter, att vi inte skulle få ihop vår produkten i tid. Men när man jobbar från olika håll i ett projekt måste man bara lita på att din partner är ärlig mot dig om hur det går.

2.4.5 erfarenheter av hur det är att samarbeta

Det viktigaste när man sammarbetar är att vara ärlig mot varandra! hur arbetet ligger till och våga säga att man har gjort fel för att ta sig an problemen snabbt (Steven C. McConnell,2004:s826). Det är inte lätt att säga att 'jag kan inte/jag förstår inte' så det är viktigt att ha en bra relation till sin labbpartner, att visa förståelse för varandra.

3 Projektet som helhet

Jag finner projektet i helhet en bra erfarenhet. Projektet var littet och man hade mycket utrymme för misstag och tid för att börja om med delar av projektet. Under projektets gång till kom det missförstång när det gäller att förstå sig på projektet och sen arbets partner. Om hur viktigt det är att börja i tid. För man kommer inte göra rätt från början.
Att göra klart projektplan i början av projektet ökar förståelsen för projektet och det ska jag ta med mig till närta projekt.
Att bygga upp en bra struktur och dela upp projektet och börja beta av problemen i små delarar. Samt att dela upp projektet i ansvarsgrupper så att alla i gruppen känner att de fyller en viktig roll i projektet. Vilket jag tror ökar sammanhålningen i gruppen.

4 Referens lista

Steven C. McConnell (2004).Code Complete -2nd ed.Microsoft Press. Redmond, Washington 98052-6399