Oppgave 1

|  |
| --- |
| På hvilken måte har *refactoring* med smidig utvikling å gjøre? Gjør rede for hva *refactoring* er. Vurder i hvilken grad dere har gjort *refactoring* i case-oppgaven deres. |

**Refactoring smidig utvikling:**

* Raskere leveranser
* Stadig lesbar kode
* Enklere for andre å fortsette på eller fullføre kode.
* Bedre flyt i utviklingen

Når en jobber med smidig metodikk er det en viktig del med stadig refaktorisering av kode, for at kode skal være enkel å forstå dersom man må endre på kode selv eller om andre må fortsette på din kode, og om andre koder videre i applikasjon og som da trenger denne koden som for eksempel en funksjon for å gjøre annen kode/program. Og ved bygging av større applikasjoner sparer man på refaktorisering av kode stadig/ofte etter som man utvikler for å oppretholde en god og lesbar struktur, da det kan bli vanskelig å gå tilbake på et senere tidspunkt etter mye annen kode er lagt inn i applikasjonen å se tilbake på tidligere kode som er vanskelig å lese da den ikke er refaktorisert. Og det vil da gjøre det lettere å samarbeide på samme prosjekt og levere fungerende applikasjoner oftere til kunde som smidig utvikling går ut på.

**Refactoring egen case-oppgave:**

I egen case-oppgave har jeg gjort refactoring ettersom jeg har programmert. Det i den grad at jeg fikk til de funksjonene og annen kode jeg programmerte til å utføre det jeg ønsket å oppnå og deretter refaktorerte jeg ved å endre navn og satte inn rett design i henhold til plan. Når jeg nærmet med ferdig på min side så fant jeg ut at det var en dårlig ide å ha flere globale variabler og jeg måtte da refaktorere noe på nytt ved å bake disse variablene inn i funksjonene og/eller endre noe på koden uten å ødelegge logikken eller funksjonen i applikasjonen.

Samt etter at vi var kommet enda lengre på prosjektet måtte jeg gjøre noen endringer i forbindelse med css på min web side da en annen sin endring skapte ubalanse i min applikasjon da måtte jeg finne feil og utbedre min applikasjon slik at den skulle fungere/se bra ut i henhold til plan selv med disse endringene og eventuelt fremtidige endringer.

|  |
| --- |
| Forklar hva *continuous integration* er og hva verdien er av å drive med det. |

Med continuous integration menes stadig å teste ny kode opp mot eksisterende kode for å kunne oppdage feil så tidlig som mulig og da kunne utbedre feilene. Og feilene som kan oppstå vil da bli desidert mindre om man tester oftere. Men dette er for kode som ikke er i bruk på en web applikasjon, men som er under utvikling. Altså en kode eller kodebase som ikke er i bruk direkte fra kunde eller andre brukere.

Så konklusjon om verdien er da at en oppdager en feil utført av en selv eller andre som jobber på prosjektet tidlig og den som har ført feilen får da i oppgave å fikse/løse feilen tidlig i prosessen. Som gjør at en i større grad kan levere et produkt i henhold til avtale/kontrakt i god tid samt i god kvalitet.

I vår case-oppgave pushet vi hele tiden opp til hoved kode i reposetorien, hvor vi da så om noe vi hadde gjort endret noe som allerede fungerte eller om det stod som det hadde gjort og vi fikk da se verdien av det selv da det fra tid til annen oppsto konflikter med eget arbeid eller andres og en fikk da fokus på å rette opp dette med en gang istedenfor å oppdage det mye senere.

|  |
| --- |
| Gjør rede for *closed-window rule* og *time-boxing*. Drøft i hvilken grad dere fulgte disse prinsippene i case-oppgaven deres. |

I close-window rule konseptet hvor man stenger av for å utføre endringer i hoved koden i en vis periode. Dette gjorde vi i den grad når vi skulle ha fremvisning av produktet for kunde sjekket vi først om alt fungerte som det skal av det vi skulle vise frem og deretter satte vi en stopp for å ikke pushe opp noen nye koder opp til reposetorien. Som vi da gjorde for å sikre at ikke noen nye endringer kunne ødelegge for fremvisningen.

Time-boxing konseptet hvor man setter tidsrammer på hver oppgave benyttet vi oss i mindre grad av, men i en liten grad da vi satte opp møter med kunde og oppgave måtte vere ferdige innen det.

Når vi startet prosjektet var vi enige om å jobbe sammen på kunde møte, opptegning av skisser og da utbedringen av modellen, før vi valgte å dele opp prosjektet i flere små oppgaver som vi da delte ut rundt om i teamet i samarbeid, når disse var utdelt jobbet vi for det meste hver for oss frem til vi var ferdige med den aktuelle oppgaven vi var utdelt.

Oppgave 2b

For å fullføre oppgave 2 så la jeg til 2 endringer i modellen.

Jeg la til et felt/objekt etter app som het input og i objektet la jeg til isEdit som jeg benyttet i forbindelse med når jeg skal legge til mapper eller filer.

La også til isEdit på hvert felt i objektene i filesAndFolders som jeg benytter ved redigering av navn på mappe eller fil, samt om jeg vil slette mappen eller filen.