Prosjektbeskrivelse

Webapplikasjonen WebChess av Kristian Stang

Innhold

Bakgrunn

Under høstsemesteret 2017 hadde vi en obligatorisk oppgave i objektorientert programmering gitt av Bjørn Kristoffersen som gikk ut på å programmere deler av en sjakkapplikasjon i Java. Den omfattet kun implementering av et sjakkbrett, samt noen få av spillets brikker, og at disse brikkene skulle flytte til ulike felter på brettet dersom dette ville vært et lovlig trekk.

Som både sjakk- og programmeringsinteressert fikk jeg fort sansen for dette, og etter endt obligatorisk oppgave satt jeg igjen med en idé til videreutvikling av denne applikasjonen, der jeg implementerer en fullt fungerende sjakkapplikasjon.

Applikasjonen som skal utvikles i dette prosjektet vil være en fullstendig sjakkapplikasjon med online flerspillerfunksjonalitet, der to spillere kan spille sjakkpartier over web.

Utstyr og behov

Til dette prosjektet trengs ikke noe annet utstyr enn det som inngår i normalt utviklingsarbeid. Applikasjonen vil utvikles i sin helhet i språket JavaScript på plattformen Node.js. Den vil bruke en rekke åpne biblioteker både på tjener og klient. Som utviklingsmiljø/-editor benyttes Microsoft sitt Visual Studio Code. I tillegg vil applikasjonen benytte MySQL som databasetjener, for lagring av spillhistorikk og eventuelt brukere. Docker vurderes som alternativ for utplasseringsplattform.

I utviklingsarbeidet benyttes eget, privat utstyr.

Mål

Resultatmål

Hovedmålet for arbeidet med dette prosjektet er å lage en fullverdig webapplikasjon der to spillere kan spille sjakk med hverandre over internett. Dette innebærer en rekke delmål:

- Brukere skal lett kunne velge mellom å opprette nye spill og å bli med en venn ved hjelp av en kode eller en URL han/hun har fått av vedkommende.
- Brukere skal til enhver tid ha en oversikt over stillingen gjennom en oversiktlig visning av brettet og brikkene, skal kunne se hvilke trekk som er lovlige for vedkommende, og skal kunne se all annen relevant informasjon om partiet ved et enkelt øyekast.
- Brukere skal kunne flytte brikker både ved hjelp av å klikke på brikkene og deretter på feltet som brikken ønskes flyttet til, og ved hjelp av klikk-og-dra-funksjonalitet.
- Applikasjonen skal støtte alle sjakkregler, og holde styr på og kommunisere til spillerne informasjon om blant annet sjakk, sjakk matt, uavgjortsituasjoner (3 repetisjoner, 50-trekksregelen, patt, mm.) og regler angående vinn/tap/uavgjort etter tidskontroll.
- Applikasjonen skal støtte mange ulike sjakkspill pågående samtidig.
- Applikasjonen skal overbringe all informasjon mellom spillerne i sanntid, slik at brukerne får en best mulig brukeropplevelse.

Effektmål

Når det gjelder effektmål egner prosjektoppgaven seg utmerket til å tilegne seg kunnskap om, og lære å bruke, noen av de mest relevante og populære rammeverkene for webutvikling i 2019 (spesifiseres under neste punkt).

Et viktig effektmål er derfor at utvikler får god trening innenfor disse teknologiene og tankemåtene, og på sikt bli bedre på å utvikle mer avanserte webapplikasjoner, da kanskje spesielt med tanke på sanntidsinformasjonsutveksling. Også arbeid med testing av slike applikasjoner blir veldig sentralt.

Til tross for at prosjektgruppa består av kun en person, vil arbeidet følge Kanban-metoden, ettersom kravene kan komme til å endres noe underveis i prosjektet, og en smidig metode vil takle dette godt. I forhold til Scrum foretrekkes Kanban, ettersom det er mer naturlig å basere seg på oppgaver enn å dele opp arbeidet i tidsbolker i dette prosjektet. Prosjektet vil gi god erfaring med Kanban i praksis.

Et mer direkte effektmål vil være å gi sjakkspillere en mulighet til å veldig enkelt starte spill med hverandre ved hjelp av et enkelt klikk, men det finnes allerede mange gode alternativer der ute.

Hvordan angripe problemet

Applikasjonen kan logisk deles opp i tre ulike delapplikasjoner:

- **Sjakkmotoren**

  Sørger for logikken i sele sjakkspillet og inneholder implementasjonen av et sjakkparti. Denne modulen inkluderer klasser for sjakkpartier, sjakkbrett og sjakkbrikker. Den vil inneholde og ta vare på stillingen i et sjakkparti, hvem som trekker, hvor mye tid hver spiller har til rådighet, og andre detaljer som rent teknisk vedgår sjakkpartiet. Hvordan brikkene individuelt skal oppføre seg og hvordan de kan flytte er også implementert her, og denne modulens funksjonalitet henger tett sammen med reglene for sjakk.

  Sjakkmotoren vil utvikles som en egen selvstendig Node.js-modul, og tenkes tatt i bruk av webtjeneren som et bibliotek etterhvert som brikker skal flyttes og andre funksjoner skal tas i bruk. Den vil utvikles objektorientert, med klasser for spill, brett og ulike brikker, og det vil ikke bli behov for bruk av andre biblioteker i utviklingen av denne modulen. Gjennom å være en uavhengig og fullverdig sjakkmodul, betyr det også at sjakk-motoren kan benyttes av andre utviklere til andre formål og i forbindelse med egne applikasjoner.

- **Webtjeneren**

  Webtjenerens hovedfunksjon er å ta i bruk sjakkmotoren og holde styr på alle sjakkpartiene som er satt i gang. Samtidig skal den holde styr på alle spillerne som er logget inn ved hjelp av sessions, og koordinere spillerne med deres pågående partier. Etter hvert som trekk utføres på klientsiden vil webtjeneren gjøre kall av ulike slag til sjakkmotoren, og deretter sende informasjon om ny posisjon og brettoppstilling tilbake til de involverte klientene. Tjeneren skal også kunne utføre databaseoperasjoner der nødvendig, og kan betraktes som kjernen i applikasjonen med tanke på kommunikasjon.

  Webtjeneren skal også utvikles som en Node.js-applikasjon, med en serverimplementasjon av GraphQL som både grunnlag for tjeneren og som et bindeledd mellom klientene. I tillegg til å fungere som et bindeledd vil GraphQL sørge for sanntidsinformasjonsflyt mellom tjener og klienter. Tjeneren skal også ta i bruk Sequelize, et populært ORM til Node.js, for kommunikasjon med MySQL-databasen for blant annet å lagre resultater ved endte partier.

- **Webklienten**

  Webklienten vil være en svært tynn webklient, som kun har som oppgave å vise frem sjakkbrettet og all annen relevant informasjon til brukeren. Den skal tilby trekkfunksjonalitet gjennom klikk-og-dra-prinsippet, eller ved at man kun klikker på en brikke og deretter klikker i feltet man ønsker å flytte til. Spillerne vil kunne få popups når bestemte spillhendelser skjer, som f.eks. at tiden er ute, at en spiller har vunnet, eller at partiet har endt uavgjort.

  Webklienten vil utvikles som en enkel React.js-applikasjon på Node.js, og vil i tillegg til dette rammeverket bruke Apollo Client som en GraphQL-klient som et bindeledd mellom klienten og tjeneren. Apollo Client bruker WebSockets bak kulissene, noe som gjør at spillerne på denne måten enkelt vil kunne sende og motta data i sanntid, noe som er avgjørende for en god brukeropplevelse ved bruk av applikasjonen.

Utfordringer og kritiske faktorer

Ettersom dette er et rent utviklingsprosjekt der de fleste aktivitetene er knyttet til programmering og design av en webapplikasjon, finnes det ikke noen særlig nevneverdige eksterne kritiske faktorer i forbindelse med prosjektets risiko. Applikasjonen avhenger for eksempel ikke av noen eksterne aktører, ekstern data eller utstyr som skal leveres og installeres.

Det finnes allikevel noen utfordringer og kritiske faktorer det vil være naturlig å trekke frem og kommentere når det gjelder utførelsen av dette prosjektet.

- **Tidsbruk**
  Ettersom det er klare tidspunkter for både milepæler og prosjektfullførelse, vil det være svært viktig å ha en god plan over tidsbruken i prosjektet, og følge denne nøye. Det er fort gjort å havne på etterskudd om man ikke planlegger tidsbruken godt.

- **Gruppestørrelse og prosjekt uten oppdragsgiver**
  Ettersom prosjektet vil utføres på egenhånd av en person, og det ikke eksisterer noen direkte oppdragsgiver for prosjektet, kan det bli vanskelig å få nødvendig tilbakemelding på applikasjonen underveis. I tillegg kan mangel på oppdragsgiver gjøre at utviklingsarbeidet går tregere enn nødvendig, gjennom å ikke ha den jevnlige oppfølgingen en oppdragsgiver normalt vil gi. Eksterne testere og samtalepartnere kan være en løsning.

- **Mange tredjepartsbiblioteker**
  Applikasjonen planlegges utviklet ved hjelp av mange ulike og store tredjepartsbiblioteker, noe som gjør at det også vil være krav om å sette seg inn i og forstå disse. Dette kan føre til ekstra tidsbruk før det kan gjøres arbeid, og i verste fall gjøre at noen biblioteker rett og slett blir uaktuelle i lengden. Det vil være lurt å finne alternativer, eller gå for en enklere selvimplementert løsning dersom det skulle bli for vanskelig.

Fremdrift og budsjett

Budsjett

Noe tradisjonelt budsjett er ikke relevant for dette prosjektet. Det er allikevel interessant å legge frem et overslag over tidsbruken som beregnes i arbeidet med prosjektet. Prosjektet antas å gå over 17 uker.

Navn	Timer per uke	Totalt antall timer
Kristian Stang	20	340

Milepælsplan

Nr.	Oppgave	Ferdig	Beskrivelse
1	Planlegging og innledende arbeid	05.02.2019	Planlegging av utviklingsarbeidet. Valg av arbeidsmetode. Valg av plattform og språk.
2	Utvikling av sjakkmotoren	05.02.2019	Sjakkmotoren er ferdigutviklet. Sjakkpartier kan spilles gjennom manuelle metodekall uten GUI. Dokumentasjon er påbegynt.
3	Utvikling av webtjeneren	12.03.2019	Webtjeneren er ferdigutviklet. Sjakkpartier kan spilles gjennom manuelle trekk i GraphQL-hjelpe-middelet «GraphQL Playground». Dokumentasjon er påbegynt.
4	Utvikling av webklienten	30.04.2019	Webklienten er ferdigutviklet. Sjakkpartier kan spilles i GUI. Applikasjonen er ferdigstilt når det gjelder det praktiske. Dokumentasjon er påbegynt.
5	Dokumentasjon av applikasjonen	07.05.2019	Applikasjonsdokumentasjonen og prosjektdokumentasjonen er ferdig. Finpussing av applikasjonen.
6	Avslutning av prosjektet	07.05.2019	Innlevering av prosjektrapporten. Innlevering av kildekode.