Fagprøve – David Hasselø

Egenvurdering

Sammendrag

I denne egenvurderingen reflekterer jeg over min gjennomføring av fagprøveprosjektet. Dokumentet inneholder mine egne vurderinger av hva som fungerte bra, hvilke utfordringer jeg støtte på, hvordan jeg løste dem, og hva jeg kunne gjort annerledes. Jeg diskuterer også bruk av tid, valg av teknologier og arbeidsmetoder, samt kvaliteten på det endelige resultatet. Egenvurderingen viser min evne til kritisk tenkning, læring og forbedringspotensial, og danner et grunnlag for videre utvikling som IT-utvikler.

David Hasselø

David.hasselo@omega365.com

Innhold

1. Planlegging og arbeidsprosess

Gjennomføringen gikk for det meste etter planleggingen, men det var noen plasser jeg trengte å improvisere, men ingen drastiske endringer. Det var små ting som å endre litt på tabelloppsettet for å ta hensyn til at en konkurranse skulle være markert som arkivert. En av de større avvikene jeg hadde var at jeg opprinnelig skulle ha et slags dashboard design, men endte opp med å holde det simpelt med en hovedside med knapper som fører brukeren til ønsket funksjon, dette mener jeg holder appen mer renslig og tradisjonell. Grunnen til at jeg hadde dette med i planleggingen var fordi jeg tenkte dette var den beste løsningen slik at brukere enkelt kunne åpne appen å få all informasjon de trengte, men når jeg implementerte det var jeg ikke særlig fornøyd med hvordan det endte og gikk for den nåværende løsningen. I tillegg manglet jeg noen API-kall når det kommer til konkurranser som jeg ikke tok særlig stilling til.

2. Gjennomføring

Gjennomføringen føler jeg gikk meget bra, og jeg ble positivt overrasket over hvor raskt jeg kom i gang allerede første dag. Oppsett av prosjektet, inkludert server oppsett, databasekonfigurasjon og grunnleggende backend, gikk lett og ga meg en god start. Jeg opplevde at jeg hadde god kontroll på arbeidsflyten, og det var motiverende å se at de ulike delene av løsningen falt på plass etter hvert som jeg jobbet.

Det jeg fikk til spesielt godt var å strukturere kodebasen på en ryddig og modulær måte, slik at det ble enkelt å utvide med nye funksjoner etter hvert. Jeg er også fornøyd med hvordan jeg fikk til samspillet mellom frontend og backend, og at jeg klarte å lage universelle hjelpefunksjoner som gjorde koden mer gjenbrukbar og oversiktlig. Bruk av Docker gjorde det enkelt å holde utviklingsmiljøet stabilt, og jeg opplevde få problemer med oppsett og kjøring av tjenestene.

3. Teknologivalg og verktøy

Jeg mener de verktøyene jeg valgte for å løse denne oppgaven var optimalt for min bruk og jeg hadde ingen avvik fra planen når det kom til det.

For backend brukte jeg Node.js og express, som gir en fleksibel struktur for å lage REST-API-er. Node er godt egnet til raske og skalerbare webtjenester, og er noe jeg har jobbet med før, så node var derfor et soleklart alternativ som ga meg få problemer under prosessen.

PostgreSQL brukte jeg som relasjonsdatabase. Fordelen med den er at den er gratis, stabil, god støtte i Node.js og har riktig mengde kompleksitet for prosjektet mitt da den passer både små og store apper. Alternativet var å bruke SQL server som database, siden det er det jeg jobber mest med i mitt yrke. Ulempen med SQL server som er noe jeg tok stilling til i planleggingen er at den er veldig tung å kjøre og har mye mindre støtte for deployment i docker og railway i forhold til postgreSQL. I tillegg er den ganske lisensbelagt og har en ganske tungvint installeringsprossess. Derfor er jeg veldig fornøyd at jeg valgte postgreSQL, og har ikke hatt problemer med det. En ulempe jeg støtet på er at jeg ikke er særlig kjent til pgAdmin, som er ett av administreringsverktøyet til postgreSQL, og det tok litt tid å bli trygg. Det jeg lærte av å implementere database i en slik fullstack applikasjon er hvordan man automatisk setter opp en init.sql for automatisk opprettelse av tabeller, pgadmin-server.json for automatisk config av databasen og hvordan de alle kan håndteres gjennom docker for å gjøre jobben til neste utvikler mye enklere.

4. Kvalitet på arbeid og resultat

Løsningen oppleves som både stabil og brukervennlig. Stabiliteten skyldes at applikasjonen er bygget med velprøvde teknologier som Node.js, Express og PostgreSQL, og at det er lagt vekt på feilhåndtering både på frontend og backend. Brukervennligheten er ivaretatt gjennom et responsivt design med Bootstrap, tydelige tilbakemeldinger til brukeren, og en enkel og intuitiv navigasjon. Universelle hjelpefunksjoner og loading states gir en jevn brukeropplevelse, og alle hovedfunksjoner er tilgjengelige fra menyen uansett hvilken side man er på.

Jeg er spesielt fornøyd med at hele løsningen dekker alle funksjonelle krav, fra brukerhåndtering til lotto-trekning, og at leaderboardet med aggregerte turer fungerer sømløst sammen med konkurransehistorikken. At trekningen faktisk belønner innsats på en rettferdig måte, og at alle data oppdateres automatisk uten manuell oppfriskning, er også et resultat jeg er stolt av.

5. Testing og dokumentasjon

Jeg valgte å bruke et strukturert testskjema i Excel for å sikre at alle funksjonelle krav i applikasjonen ble grundig testet og dokumentert. Skjemaet er gruppert etter hovedfunksjoner i løsningen, med sjekkbokser for hvert krav og felt for merknader og utfyller. Dette gjorde det enkelt både for meg selv og en kollega å gå systematisk gjennom funksjonaliteten og gi tilbakemeldinger.

I tilbakemeldingene fra kollegaen var alle sjekkboksene for de funksjonelle kravene sjekket, noe som indikerer at alt er funksjonelt på brukernivå. I tillegg fikk jeg flere innspill på forbedringspotensialet. Blant annet kom han med disse forbedringene og feilstillingene:

- Passordregler for å sikre minimumslengde
- Egen «legg til ny tur»-knapp på hovedsiden
- Ved redigering huskes ikke valgt dato for turer
- Kunne sikret at bruker ikke får legge inn turer etter dagens dato
- Kan knytte gamle turer og fremtidige turer utenfor konkurranse-tidspunkt til aktiv konkurranse
- Ved redigering huskes ikke valgt dato for konkurranser
- At man kanskje ikke trenger å vise brukere med 0 turer registrert for konkurransen
- Tabellvisning kan forbedres

Link til testksjema Fagprøve Testskjema - David

Noen av disse forbedringspunktene var jeg allerede oppmerksom på, mens andre ville jeg kanskje ikke oppdaget uten ekstern testing. Dette viser at testskjemaet og gjennomgangen sammen med en kollega ga god effekt, og bidro til å belyse flere muligheter for videreutvikling og økt brukervennlighet, som er noe jeg ville hatt fokus på dersom jeg skulle videreutvikle appen.

6. Refleksjon og læring

Gjennom arbeidet med StikkUt-appen har jeg fått verdifull erfaring med å bygge en komplett fullstack-applikasjon fra bunnen av. Jeg har lært mye om hvordan man strukturerer et prosjekt med tydelig separasjon mellom frontend, backend og database, og hvordan man bruker Docker Compose for å forenkle oppsett og kjøring av flere tjenester samtidig. Jeg har også fått bedre forståelse for sikkerhet i webapplikasjoner, spesielt rundt autentisering med JWT og håndtering av sensitive data. I tillegg har jeg blitt tryggere på å bruke moderne JavaScript (async/await), og på å lage gjenbrukbare komponenter og hjelpefunksjoner for å holde koden ryddig og oversiktlig.

Jeg har blitt mer bevisst på at jeg er god til å strukturere prosjekter, og til å finne løsninger på problemer som oppstår underveis. Jeg ser også at jeg har blitt flinkere til å skrive lesbar og gjenbrukbar kode. Samtidig har jeg erfart at jeg kan bli enda bedre på planlegging, og at jeg noen ganger bruker for mye tid på detaljer i stedet for å prioritere det viktigste først. Hvis jeg skulle løst en slik oppgave i framtiden hadde jeg lagt vekt på å planlegge tidsbruk og arbeidsoppgaver på en litt mer omfattende måte, og i tillegg lagd en to-do list for alle arbeidsoppgaver slik at jeg kan enkelt planlegge og sette av tid til gjennomføringen.

7. Bruk av tid

Når det gjelder bruk av tid, opplevde jeg at jeg lå foran skjema i starten av gjennomføringen. Etter hvert som jeg støtte på utfordringer, og enkelte funksjonelle krav krevde mer tid til feilsøking og testing, ble fremdriften noe forsinket. Målet mitt fra planleggingen var å alltid ligge litt foran, slik at jeg kunne unngå stress og heller bruke god tid på kvalitetssikring og dokumentasjon.

I praksis brukte jeg mye tid på selve utviklingen av appen, spesielt fordi det dukket opp flere forbedringspunkter og nye ideer underveis. Dette gjorde at dokumentasjonsarbeidet til slutt måtte gjennomføres mer intensivt på slutten. Jeg ser i etterkant at det er lett å undervurdere hvor lang tid det tar å ferdigstille, teste og dokumentere et fullstendig system, spesielt når man også vil implementere ekstra funksjonalitet utover minimumskravene.

Likevel opplever jeg at jeg klarte å holde god kontroll på fremdriften, og at jeg fikk levert både funksjonell programvare og grundig dokumentasjon innen fristen. Dette har vært en nyttig erfaring for videre prosjekter, og jeg vil i fremtiden være enda mer bevisst på å sette av nok tid til testing og dokumentasjon, og vurdere om alle ekstra funksjoner er nødvendige innenfor tidsrammen.

8. Samarbeid og kommunikasjon

Jeg har for i hovedsak jobbet selvstendig med prosjektet, men jeg har fått nyttige tilbakemeldinger fra en kollega Spesielt var det verdifullt å la en kollega teste løsningen gjennom det strukturerte testskjemaet jeg laget. Han oppdaget enkelte forbedringspunkter jeg ikke selv hadde sett, og kom med konkrete innspill til funksjoner og brukervennlighet. Dette bidro til at jeg fikk et mer objektivt blikk på løsningen, og at jeg kunne gjøre justeringer for å forbedre sluttresultatet.