Fagprøve – David Hasselø

Logg

Dag 1: Planlegging

Denne dagen bestod det meste av planlegging for gjennomføringen av prosjektet. Jeg skrev om min tolkning av oppgaven, funksjonelle krav, drøfting av løsninger, framdriftsplan, valg av teknologier og design.

Dag 2: Gjennomføring

Forarbeid

I dag har jeg startet med selve utviklingen av prosjektet. Først og fremst startet jeg med å sette opp prosjektstrukturen. Jeg initialisere git repo og node, lastet ned nødvendige dependencies basert på teknologiene jeg valgte og strukturerte mapper.

Registrering, login og autentisering

Ettersom forarbeidet var gjort startet jeg med det første punktet i forhold til framdriftsplanen som var å skrive CRUD operasjoner for brukere. Jeg startet først med å opprette databasen med tabeller for brukere, konkurranser og turer. Deretter implementerte jeg brukerregistrering med sikker passord-hashing via bcrypt. Dette er noe jeg har gjort i en tidligere oppgave så jeg tok utgangspunkt i oppsettet derfra. Registreringen ble testet gjennom en enkel html formskjema der jeg sjekket på navn, etternavn, e-post og passord. Derfra verifiserte jeg at all brukerdata blir lagret på riktig måte i users-tabellen og i tillegg at passord blir hashet og saltet. Etter registreringen var på plass er det naturligvis viktig at de skal kunne logge inn med registrert bruker. Jeg implementerte derfor innlogging med JWT-token for autentisering. Når begge disse var på plass lagde jeg også JWT-middleware for å beskytte ruter som krever innlogging, som hindrer at brukere uten tilgang kan gjøre spesifikke endringer i appen.

Tur-håndtering

Ett av de funksjonelle kravene var muligheten til at brukere skal kunne registrere, redigere og slette turer. Jeg startet med registrering av turer der brukere kan legge til turer med et par beskrivende felter som dato, sted og hvem som deltok. Jeg valgte et modal design for å registrere turer for å gi en ren og enkel oversikt over de ulike feltene. Jeg

implementerte en sjekk gjennom JWT for oppretting av turer for brukere der den sjekker om de er logget inn, dette er for at jeg skal enkelt spore hvilken bruker som eier hvilke turer. For å knytte en tur til en spesifikk konkurranse valgte jeg å legge til enda et felt der brukere velger konkurranse i form av en drop liste. Ettersom brukeren har fylt inn all nødvendig informasjon blir turen registrert og knyttet opp mot valgt konkurranse som igjen gjør det enkelt å spore når jeg kommer til den delen hvor det skal vises aggregerte data over turer. Etter jeg var sikker på at all backend logikk fungerte som det skulle lagde jeg et enkelt frontend design for å hente dataen og visualisere det på siden.

Når dagen kom til slutt følte jeg at jeg hadde en god start og ligger i tråd med planen. Neste steg er å opprette CRUD operasjoner for tours og competitions, enkel frontend, tilgangskontroll for admin på competitions og forhåpentligvis starte på aggregert rangering.

Dag 3: Gjennomføring

I dag har jeg implementert og testet fullverdige CRUD-operasjoner for både turer (tours) og konkurranser (competitions). Dette inkluderer:

Opprettelse, visning, redigering og sletting av turer:
Brukere kan nå registrere nye turer, se sine egne turer, samt redigere og slette dem via et brukervennlig grensesnitt. Alle operasjoner bruker fetch mot backend-APIet.

CRUD for konkurranser:
Det er lagt til funksjonalitet for å opprette, vise, redigere og slette konkurranser. Det er innført logikk som sikrer at kun én konkurranse kan være aktiv om gangen. Her møtte jeg på et problem der jeg ikke hadde tatt stilling til hvordan competitions api-et skulle se ut i planleggingen, men heldigvis var den ikke så mye annerledes enn api-et for tours, men definitivt noe jeg burde tatt stilling til uansett.

Enkelt brukergrensesnitt:
UI-et bruker Bootstrap og modaler for å forenkle oppretting og redigering, uten behov for å navigere bort fra siden.

Sikkerhet og rollebasert tilgang:
Kun brukere med admin-rettigheter får tilgang til å opprette, redigere og slette konkurranser. Vanlige brukere får ikke tilgang til disse funksjonene.
I tillegg er det lagt inn sjekk slik at det ikke er mulig å opprette mer enn én konkurranse av gangen.

Dataintegritet og bruk av fetch:
All datakommunikasjon skjer via fetch mot REST API, og det er gjort grundige sjekker både på frontend og backend for å sikre at bare gyldige forespørsler blir gjennomført.

Dag 4: Gjennomføring

I dag la fokuset i å få gjort ferdig konkurranser og deretter aggregert rangering. Jeg fortsatte med å se på tilgangskontroll for konkurranser og historikk for tidligere vinnere av ferdige konkurranser. Konklusjonen ble å legge til et oppslagsfelt i oversikten over konkurranser der brukere kan velge mellom tidligere konkurranser. Da de velger en konkurranse vil de kunne se oversiktlig informasjon som premien, dato leaderboardet over hvem som vant. Her møtte jeg på et problem der alle konkurranser var satt som aktiv siden det ikke var en sjekk på dato, som er det jeg implementerte for å sette en konkurranse som inaktiv og dukke opp i historikken. Ettersom konkurranser var ferdigstilt, startet jeg på et «leaderboard», som viser alle innloggende brukere, med rangering, fullt navn og antall turer som blir lagt sammen per bruker gjennom en sql COUNT() spørring. Dette tilsvarer den aggregerte oversikten over turer som gir grunnlag for «lotto» logikken som skal videre innføres.

Denne dagen møtte jeg på en del forhindringer som gjorde at jeg stoppet opp litt, men når jeg fikk rangeringen på plass følte jeg at jeg var tilbake på spor og etter planen. Det neste som gjenstår er å implementere lotto-trekningen der innsats belønnes for brukere, som jeg så vidt fikk begynt på i dag.

Dag 5: Gjennomføring

I dag hadde jeg som hovedmål å ferdigstille all kjernefunksjonalitet, og jeg gikk rett i gang med å utvikle lotto-/flaxlodd-funksjonen for konkurransen.

Jeg begynte med å sette opp en egen backend-route og controller for loddtrekning, hvor jeg måtte sørge for at hver tur gir ett lodd i trekningen slik at brukere med flere turer får større vinnersjanse. Jeg brukte en “weighted random”-algoritme for å gjøre trekningen rettferdig. Her støtte jeg på noen utfordringer hvor jeg måtte ha en måte å definere en konkurranse som avsluttet når en trekning skjer. Dette løste jeg ved å legge til is_archived kolonnen i competitions tabellen som en enkel indikator på om en konkurranse skal vises som aktiv eller havne i visningen for historikk, og gjør det også enklere å skille i databasen.

Videre jobbet jeg med å forbedre frontend for å vise både aktiv konkurranse og konkurransehistorikk på en ryddig måte, blant annet med select/dropdown for arkiverte

konkurranser slik at det er enkelt å se gamle vinnere og premier. Jeg la også til ekstra funksjoner for å kunne hente enkelt-turer til redigering, og flyttet gjentakende frontend-logikk over i en felles common.js-fil for bedre vedlikeholdbarhet.

I tillegg la jeg til en Docker-compose.yml og init.sql slik at det skal være enkelt for sensor å kjøre opp alt fra bunnen av uten manuelle grep.

Jeg har også ryddet i repoet, slettet unødvendige filer og forbedret styling slik at alt følger Omega 365 sin brand manual.

Etter alt dette er jeg på god vei, og det som gjenstår til i morgen er testing og små finishing touches for å forbedre brukeropplevelsen og deretter dokumentasjon.

Dag 6: Gjennomføring og dokumentering

Jeg startet dagen med en større opprydding og refaktorering av frontend-koden. En viktig endring var å samle felles funksjonalitet og hjelpefunksjoner i en ny common.js-fil, slik at gjenbrukbar kode som håndtering av API-kall, feilmeldinger, og tilgangsstyring ikke dupliseres. Dette har forbedret ytelse og vedlikeholdbarhet betraktelig.

For å styrke brukeropplevelsen la jeg til:

- Lastetilstander på universelle API-kall.
- Universell feilhåndtering.
- Bedre visuell respons på knapper og inputfelt.
- Tilgangskontroll for navigasjonselementer, slik at brukere som ikke er innlogget ikke får tilgang til knapper eller lenker de ikke skal se.

Videre har jeg gjort endringer i konkurranseloggikken, blant annet en fiks der vinneren nå vises med navn i stedet for ID. Jeg forbedret også visningen av konkurransehistorikk og designet elementer som popup-bokser og modalvinduer med sweetalert2 og font-awesome for en mer profesjonell og intuitiv stil.

Jeg gjorde forbedringer på konkurransesiden:

- Mulighet for å legge til beskrivelse på konkurranser.
- Validering og inputkontroll ved oppretting og redigering.
- Ekstra felt for konkurranser som ikke er bundet til turer, samt mulighet for å sette konkurranser som avsluttet (arkivert).
- JWT-token-eksiprasjonshåndtering ble forbedret med utvidelse i middleware.

I tillegg har jeg laget egne CSS-filer for typografi, farger, knapper, forms og annet, strukturert etter formål, og fulgt Omega 365 sin brand manual gjennom hele stilen.

Mot slutten av dagen begynte jeg på system- og faglig dokumentasjon. Jeg dokumenterte blant annet:

- Oppbygging av backend og frontend.
- Beskrivelse av databasen og ER-modell.
- Brukerroller og tilgangsstyring.
- Lotto-funksjonaliteten og hvordan loddvekt fungerer.
- Universell struktur for API-kall og hvordan feil håndteres.

Målet i morgen er å fullføre hele dokumentasjonen og levere oppgaven komplett og ryddig. Jeg ligger godt an i forhold til planen og føler systemet nå er både stabilt, effektivt og brukervennlig.