Eksamen for PG6301 Web utvikling og API design, 2022

Dette er en praktisk eksamen som skal evaluere din mestring av teknologiene og konseptene som har blitt gjennomgått i faget: React, Express, Mongodb, Heroku, Jest, OpenID Connect og Web Sockets. At det er en praktisk eksamen innebærer at du evalueres basert på hva du lager og hvordan det fungerere og ikke på teoretisk kunnskap du har tilegnet deg.

Læringsmålene eksamen skal demonstrere er som følger:

1. Lage en app med parcel, express, concurrently, prettier, Jest
2. Sette opp en fungerende React app med React Router, håndtering av loading state og feilhåndtering
3. Sette opp en fungerende Express app inkludert Routes i egen fil
4. Kommunikasjon mellom klient og server med GET og POST inkludert feilhåndtering
5. Deployment til Heroku
6. Lagring, henting og endring av data i Mongodb
7. Login med OpenID Connect (både Google og Active Directory)
8. Web Sockets
9. Test coverage på 50-70% eller bedre dokumentert med Github Actions

- For å oppnå A må alle 9 av disse være dekket
- For å oppnå B må 8 av disse være dekket
- For å oppnå C må 6-7 av disse være dekket
- For å oppnå D må 4-5 av disse være dekket
- For å oppnå E må 2-3 av disse være dekket og applikasjonen må enten kjøre på Heroku eller `npm install && npm test && npm start` må kjøre uten feil og gi en brukbar applikasjon

Om du kun gjenskaper koden fra forelesningen teller det som et halvt poeng. Om du løser punktet delvis teller det delvis. Om du har mestret noe, men har fått problemer med å få det til å virke på eksamen kan du dokumentere det i README.md for å få delvis uttelling. Vesentlig mangler i kodestil, ustabil oppførsel eller fraværende eller problematisk CSS layout trekker ned.

Innleveringen skal være en ZIP-fil med din kildekode som eksportert fra Github som et vedlegg i Wiseflow.

README-fila skal innehold link til Github repository slik at sensor kan se Actions-rapport og til Heroku slik at programmet kan testes ut

Viktig:

- Eksamen må besvares i Github Classroom med følgende link: https://classroom.github.com/a/d9U-EtNf
- Lever en ZIP-fil (ikke RAR, 7z eller tilsvarende). Den beste måten er å utvikle på Github og velge "Code" > "Download ZIP" i ditt repository på github.com
- README-fila på Github og i ZIP-fila må inneholde link to Github repository og Heroku-applikasjonen
- Genererte filer som `.node_modules`, `.parcel-cache`, `dist` og `coverage` må ikke være sjekket inn på Github eller i ZIP-fila
- Les oppgavebeskrivelsen grundig og sjekk at du ikke har oversett noen funksjonelle krav
- Du trenger ikke å løse alle krav for å få en ok karakter, prioriter hvilke krav du vil løse og dokumenter eventuelle forenklinger du har utført i README.md

Samarbeid under eksamen:

Du kan be andre om hjelp under eksamen, men du må angi kode du ikke har skrevet eller du har delt med andre i README.md om det ikke skal regnes som plagiat. Du vil bli evaluert på egen kode du leverer og bør angi hvilken kode du ikke selv har utviklet

Oppgavebeskrivelse

Du skal implementere en webapplikasjon som lar brukere lese, legge inn og oppdatere nyhetsartikler.

Nettsiden skal ha tre type brukere: Anonyme brukere som kun kan se overskrifter, registrerte brukere som kan lese nyhetsartiklene og redaksjonelle brukere som kan legge inn nye artikler og redigere sine egne artikler.

Det anbefales at du starter med å lage en plan for hvordan du skal løse innleveringen. Du kan legge planen i README-fila om du vil

Funksjonelle krav:

- [ ] Anonyme brukere skal se nyhetsaker når de kommer til nettsiden. Legg inn noen nyhetssaker for å demonstrere
- [ ] Når en ny sak publiseres, skal alle brukerne få se den nye saken umiddelbart. Bruk websockets for å sende oppdateringer
- [ ] Brukere kan logge seg inn. Det anbefales at du implementerer at brukerne logger seg inn med Google, men andre mekanismer er også akseptabelt
- [ ] En bruker som er logget inn kan se på sin profilside (userinfo fra Google)
- [ ] Brukere skal forbli logget inn når de refresher websiden
- [ ] En bruker som er logget inn kan klikke på en nyhetssak for å se detaljene om nyhetssaken. Detaljene skal inkludere en nyhetskategori, overskrift, tekst og navn på den som publiserte den
- [ ] "Redaksjonelle brukere" kan logge seg inn med Active Directory. Det må fungere å logge seg inn med en Active Directory på skolens AD (`domain_hint=egms.no`)
- [ ] Redaksjonelle brukere kan publisere nye nyhetsartikler
- [ ] Nyhetsartikkel skal inneholde en kategori valgt fra en nedtrekksliste (`<select>`), tittel (`<input>`) og tekst (`<textarea>`)
- [ ] Dersom noen allerede har publisert en nyhetsartikkel med samme tittel skal serveren sende HTTP status kode 400 og en feilmelding

- [ ] Brukeren skal forhindres fra å sende inn en nyhetsartikkel som mangler kategori, tittel eller tekst
- [ ] En redaksjonell bruker skal kunne redigere en artikkel de selv har publisert
- [ ] Alle feil fra serves skal presenteres til bruker på en pen måte, med mulighet for brukeren til å prøve igjen

Må-krav til teknisk løsning

- [ ] Besvarelsen skal inneholde en README-fil med link til Heroku og test coverage
- [ ] `npm start` skal starte server og klient. Concurrently og parcel anbefales
- [ ] `npm test` skal kjøre tester. Testene skal ikke feile
- [ ] Koden skal ha konsistent formattering. Prettier og Husky anbefales
- [ ] Nettsidene skal ha god layout med CSS Grid (Holy Grail layout) og horisontal navigasjonsmeny. Brukeren må kunne navigere overalt uten å bruke "back" eller redigere URL
- [ ] Serveren validerer at brukeren er logget inn
- [ ] Innleveringen skal være i form av en ZIP-fil. Maks størrelse på fila er 1MB
- [ ] Artikler skal lagres i MongoDB
- [ ] Applikasjonen skal deployes til Heroku
- [ ] Testene skal kjøre på Github Actions

Bør-krav til teknisk løsning

- [ ] Github Actions bør beregne testcoverage. Testdekningen bør være over 50%. Bruk `collectCoverageFrom` for å inkludere *alle filer*. Kun genererte filer som `coverage` og `dist` skal ekskluderes
  - Vi har fått en rabattkode som gjør det mulig å benytter coveralls. Du kan bruke denne eller `jest-coverage-report-action`
- [ ] Brukeren ser kun menyvalg som de har tilgang til
- [ ] Brukere som går til en side de ikke har tilgang til blir bedt om å logge inn
- [ ] Brukere bør alltid se listen over artikler når de navigerer seg rundt på sidene