Utvikling av en Frontend-applikasjon og Testing av API (Del 2 av prosjektoppgaven)

Dette dokumentet beskriver del 2 av prosjektoppgaven, som bygger videre på APIet utviklet i del 1.I del 2 jobbet jeg for å utvikle en frontend-applikasjon i Blazor som fungerer som brukergrensesnitt for dette APIet. Målet med frontend-applikasjonen var å gi brukerne en enkel måte å administrere dokumenter og mapper på, med autentisering og sikkerhet via JWT-token. I tillegg implementerte jeg tester for å sikre at APIet fungerer som forventet, og gjorde enkelte forbedringer i APIet for å tilrettelegge for frontendens behov. Selv om tidsbruken ble en utfordring, klarte jeg å levere en fungerende løsning jeg er fornøyd med.

Autentisering:
Frontend-applikasjonen bruker JWT-token for autentisering. Ved innlogging lagres tokenet i nettleserens localStorage, og dette tokenet brukes i Authorization-headeren for alle API-kall. Dette sikrer at kun autoriserte brukere får tilgang til mapper og dokumenter.

Mapper:
Mapper implementeres som en hierarkisk struktur der brukerne kan opprette, redigere og slette mapper. Mapper kan også inneholde undermapper, og jeg brukte rekursive metoder for å vise denne strukturen i frontend-applikasjonen.

Dokumenter:
Dokumenter er knyttet til mapper, og brukerne har mulighet til å opprette, redigere og slette dem. En utfordring som gjenstår, er at opprettede dokumenter ikke vises i applikasjonen. Dette skyldes at jeg ikke rakk å implementere funksjonaliteten for å oppdatere visningen etter opprettelse.

Brukergrensesnitt og navigasjon:
Frontend-applikasjonen bruker knapper for å navigere mellom mapper og vise tilhørende dokumenter. Jeg la også til funksjoner for å logge ut og returnere til innloggingssiden.

Endringer i APIet for å støtte frontend:

For å støtte funksjonaliteten i frontend-applikasjonen, ble det gjort endringer i API-et. Spesielt ble en ny endpoint implementert: GetFoldersWithDocuments. Dette endepunktet returnerer en liste over mapper sammen med tilhørende dokumenter, slik det ble demonstrert i videoen.

Testing av APIet:

Jeg brukte xUnit for enhetstesting av APIet for integrasjonstester. Testene fokuserte på følgende:

- Autentisering: Sørge for at brukere med riktige legitimasjoner kan logge inn og motta en gyldig token.

- CRUD-operasjoner: Validerte opprettelse, oppdatering og sletting av mapper og dokumenter.

Refleksjon

Denne oppgaven har vært svært lærerik, men også krevende. Jeg undervurderte hvor mye tid det ville ta å implementere frontend-applikasjonen, spesielt når det kom til feilhåndtering og testing. Likevel føler jeg at jeg fikk en god forståelse for hvordan Blazor fungerer som frontend-rammeverk og hvordan API-integrasjon kan struktureres. Jeg oppnådde en godt fungerende integrasjon mellom frontend og API. Autentisering med JWT fungerte godt, og jeg klarte å bygge en fleksibel løsning for mapper og dokumenter. Jeg brukte mye tid på debugging for å sikre at API-kall fungerte som forventet.
Jeg kunne optimalisert applikasjonen ved å redusere antall API-kall og implementere lokal caching av data. Jeg ville også brukt mer tid på å forbedre brukergrensesnittet, særlig for å gjøre applikasjonen mer intuitiv for sluttbrukeren.

Konklusjon

Oppgaven ga meg verdifull erfaring i hvordan man utvikler en applikasjon fra backend til frontend. Jeg lærte mye om autentisering, datavisualisering og API-integrasjon. Selv om tidsstyringen kunne vært bedre, er jeg fornøyd med resultatet og hvordan prosjektet reflekterer min innsats og læring.