Projekt 1 - bibliotek

Christian E & Kenneth K

Rapport

Indledning med indhold og kort resume

I denne rapport vil vi komme ind på processen i det projekt vi er blevet stillet i undervisningen, som handler om at lave en webapplikation til at vedligeholde en XML-fil. På websiden skal det være muligt for en bruger at opdatere et “bibliotek” med bøger vha. input felter, og derefter kunne se hele biblioteket og samtidig sortere i måden bøgerne fremvises på.

Formålet med projektet går ud på at vi som studerende kan anvende de færdigheder vi har tilegnet os i fagene, på en praktisk måde, i en problemstilling som ikke er kendt af os i forvejen.

I projektet har vi primært gjort brug af vores nyligt tilegnede viden omkring XML og Node.js, men også JavaScript og andre teknologier fra tidligere semestre er inddraget i mindre grad gennem vores løsning. Derudover har vi også brugt teori som vi har lært i undervisningen, understøttet med teori i diverse relevante bøger og forums på nettet.

I løbet af rapporten vil vi komme ind på hvilke metodeovervejelser vi har gjort os, på baggrund af den stillede opgave. Derudover vil vi forklare hvordan vi har researchet for at få løst opgaven, og i vores tilfælde vil denne del mest gå med at forklare det vi har lært i undervisningen, da det er udgangspunktet for den stillede opgave, og vores research har dermed baseret sig på emner relevant for dette.

I analysefasen vil vi prøve at besvare de spørgsmål vi har stillet os selv i vores problemformulering, og analysere på hvordan vi har løst dem.

I konstruktionsfasen vil vi gå i dybden med, hvordan vi har opbygget nogle af de forskellige elementer som udgør vores webapplikation, og her vil vi primært fokusere på de læringselementer som opgaven fokuserer på, altså XML og Node.js.

Til sidst i rapporten vil vi evaluere hele processen, og komme ind på, hvilke dele af opgaven der har været mest udfordrende, samt hvilke dele var lettere at håndtere. Derudover vil vi komme ind på om der er noget vi ville have gjort anderledes, eller

prioriteret på en anden måde hvis vi skulle gøre det igen, for til sidst at afslutte med en konklusion.

Problemformulering

Hvordan sikrer vi os at vores løsning både kan modtage og validere det data som modtages, samtidig med at det er brugervenligt at indtaste nye bøger til vores database. Og hvordan sikrer vi at vores data kan blive fremvist på en overskuelig måde, selv ved forskellige sorteringsmetoder?

Research

Weekenden der startede lige efter projektets kick-off brugte vi som vores researchfase. Her læste vi begge de relevante dele af undervisningsmaterialet fra andet semester igennem. Vi kiggede også vores opgaveløsninger samt de tilhørende løsninger på dkexit igennem, for at få en så god forståelse for metoderne, vi skulle bruge, som muligt. Vi har tilmed gået tilbage til udvalgte dele af undervisningsmaterialet på dkexit hver gang vi stødte på et problem undervejs i projektforløbet. Vi har sparsomt brugt eksterne inspirationskilder, såsom W3schools, stackoverflow med mere, ikke for at bruge deres materiale direkte, men for at få en anden synsvinkel og/eller forklaring på de metoder, vi skulle anvende til vores løsning, som vi havde mest svært ved at få en grundig forståelse for.

Metoder

For at lave en omfattende løsning har vi brug for en lang række kodefiler, af forskellige typer, der hver har indhold med forskellige formål og forbindelser til hinanden. Den præcise konstruktion af vores mappes indhold kommer vi mere ind på under “konstruktion”. I dette afsnit kommer vi ind på hvilke metoder, vi vil bruge til at lave en løsning, der har de egenskaber, som projektbeskrivelsen forventer af produktet. Vi starter naturligvis med et HTML dokument, med navnet “index.html”. Da vi har valgt at lave vores løsning som en onepager, så er HTML-dokumentet her naturligvis vores forside, altså den side, som vises i browseren for brugeren. Det indeholder links til vores to CSS-stylesheets og til vores script.js-fil. script.js-dokumentet har kun tre linjer kode i sig, som bruges til at kalde på vores

server.js og router.js-dokumenter. Derudover har HTML-dokumentet også en form med inputfelter og en submit-button, som er den brugeren anvender til at indtaste info om, og oprette nye bøger. HTML-filen displayer så også de bøger, der ligger i vores XML-fil, ved navn “books.xml”.

XML-dokumentet bruges til at opbevare data om bøgerne i vores “bibliotek”. Det er opskrevet i en almen træstruktur, hvor alt indhold er omringet af et <booksLibrary>-tag. Heri findes der childelementer med de forskellige bøgers tags med indformation via plain tekst, omringet af <books>-tags. Udover bøgerne er der også et link i booksLibrary-starttagget, der henviser til vores xsd-fil, ved navn books.xsd. Vores XSD-dokument bruges til at definere alle elementer og attributter, der findes i vores XML-dokument. XSD-dokumentet skrives som et XML-schema, hvilket bruges til at beskrive strukturen af vores XML-dokument. Det er heraf XSD-navnet kommer fra; XML Schema Definition. Yderst i XSD-strukturen har vi <xs:complexType>-tags. De er elementer der opbevarer andre elementer og/eller attributter. Heri findes også <xs:sequence>-tags, som definerer den sekvens/rækkefølge, som vores elementer skal komme i. Elementerne i et <xs:sequence>-tag kan fremkomme 1 eller flere gange. De forskellige elementer er defineret via <xs:element/>-tagget. De har hver et navn og de fleste også en type, enten string, short eller long. Typen bruges til at definere præcist hvilken type indhold, der findes i elementet.

Vi bruger Node.js til at implementere backend-delen af vores løsning, til selve opdateringen af vores biblioteks indhold. Node.js er et JavaScript runtime miljø, der lader os kode både frontend og backend i JavaScript. Vi har en række js-filer til formålet, herunder handler.js, router.js og server.js. Vi bruger server.js til at starte selve serveren og modtage datainputtet fra HTML-sidens inputfelter, som så sendes videre via body-variablen til router.js, der sender dataen videre til vores handler.js-fil. Her indsættes den validerede data så i vores XML-fil; books.xml, som nævnt tidligere og vises derigennem på index.html i browseren, hvis altså serveren er aktiv.

Analyse

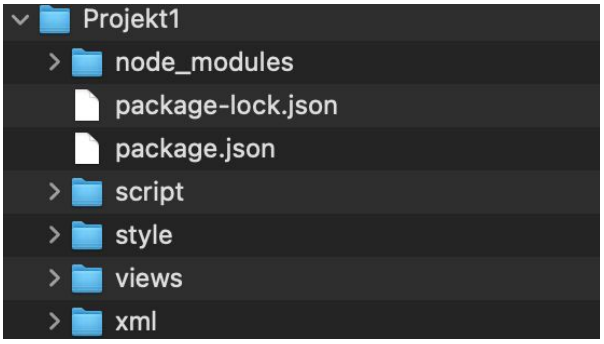
I første del af vores problemformulering ville vi sikre at vores løsning både kan modtage data, samtidig med at det er brugervenligt og bliver valideret. Hvis man kigger på vores løsning i konstruktions afsnittet, har vi forsøgt at gøre indtastningen af nye bøger meget simpel og ligetil, for derved at skabe brugervenlighed ved at eliminere eventuelle misforståelser som brugeren kan have. Valideringsprocessen i vores løsning, består på nuværende tidspunkt kun af de typer som vi har tildelt vores input felter i HTML'en. Dvs. at hvis man fx forsøger at indtaste bogstaver i felterne hvor der kræves tal, vises det med rødt at indtastningen er "ulovlig", og derfor skal man ændre det. Hvis vi havde haft tid til at implementere rigtig validering skulle der både være mulighed for at hvert input felt ville fortælle med en tekst, hvad der var galt i indtastningen, samtidig med at man ikke skulle have muligheden for at tilføje en bog til biblioteket, medmindre den rette syntaks var overholdt.

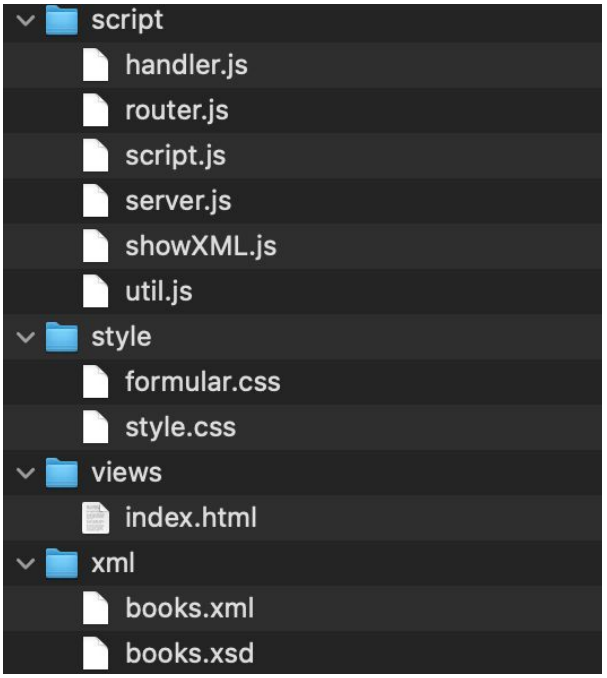
I anden del af vores problemformulering har vi stillet os selv et spørgsmål, omkring hvordan vi sikrer at vores data fremvises på en overskuelig måde, både som standard og ved forskellige sorteringsmetoder. Som det første har vi forsøgt at vise alle bøgerne i biblioteket lige ved siden af felterne hvor man tilføjer en ny bog, og hver gang en ny bog tilføjes, opdateres biblioteket ved siden af. At fremvise bøgerne ved forskellige sorteringsmetoder har vi heller ikke haft tid til, grundet problemer og tidspres, men fremvisningsmetoden skulle være den samme uanset om man ønsker at sortere efter udgiver, forfatter eller andet.

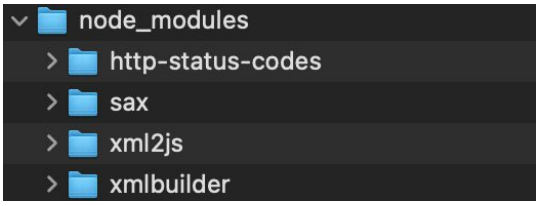
Konstruktion

Konstruktionen af vores projekt kan ses på de tre nedenstående billeder. Vi har delt de forskellige filtyper op i mapper, som vist. Det er for at holde orden i løsningen og skabe overskuelighed. Øverst har vi node-moduler i node_modules. Dernæst har vi samlet alle vores scripts i script-mappen. Vi har ligeledes lagt vores stylesheets i style-mappen, xml filer i xml-mappen og vores index.html i view mappen. Det er en simpel opdeling, der gør det let for os, såvel som andre at finde præcist det man leder efter.

Som det kan ses af billederne har vi også flere filer i de forskellige mapper, dette er også noget vi har gjort for at skabe overskuelighed i vores løsning. Så i stedet for at have fx en JavaScript fil fyldt med al koden, har vi forsøgt at dele det op i filer som giver mening for den handling de skal udføre. Dette gælder ikke kun for JavaScript filerne, men også for vores stylesheets og filer relateret til XML. Udover at det skaber overskuelighed både for os selv, vil det også være med til at gøre det mere overskueligt hvis der kom en ny udvikler der skulle skabe sig et overblik over en løsning. Samtidig giver det også mulighed at bruge de forskellige filer som moduler, hvis man har brug for at genbruge nogle funktionaliteter på et senere tidspunkt.







Evaluering af proces

Udviklingsprocessen af webapplikationen har vist sig at være mere besværlig og udfordrende end vi havde regnet med, og derfor har vi heller ikke nået at færdiggøre alle de elementer som vi havde håbet på. Dette afspejler sig i vores løsning, både i form af designet men også i de funktionaliteter, som vi gerne ville have haft at løsningen skulle indeholde. Dette har været med til at gøre vores proces til en lidt kedelig oplevelse, da vi havde håbet på at opgaven var lettere at gå til for os.

Konklusion

Hvis vi tager udgangspunkt i vores problemformulering og den opgave der er blevet stillet, kan vi konkludere at vi desværre ikke nåede helt i mål med vores løsning. Som beskrevet i evalueringen, har dette været en blanding af at det har været meget sværere for os end hvad vi havde forventet, samtidig med mangel på tid ift. at få det udfærdiget sådan som vi gerne ville have løsningen skulle være. De nuværende restriktioner har selvfølgelig hindret optimalt gruppearbejde og forståelsen for den hjælp vi har fået, men alt i alt er vores løsning ikke fuldstændig tilfredsstillende.

Referencer

1. Wexler, Y. (2019) *Get Programming With Node.Js*. Shelter Island, NY: Manning Publications Co
2. http://dkexit.eu/webdev/site/index.html (n.d.) *Web Development* [online] available from <http://dkexit.eu/webdev/site/index.html> [18 February 2021]
3. Atta (n.d.) *How to Edit an XML File with Node.Js* [online] available from <https://attacomsian.com/blog/nodjs-edit-xml-file> [18 February 2021]

4. w3schools.com (n.d.) *XML Schema Element* [online] available from <https://www.w3schools.com/xml/schema_schema.asp> [18 February 2021]