PROJEKT 2 - BIBLIOTEK

Rapport

19. MARTS 2021

IBA PB WEB
Christian Ehlers & Kenneth Kristensen

Indledning

I denne rapport vil vi komme nærmere ind på processen i det projekt vi er blevet stillet, i forbindelse med undervisningen og vores projektuge. I projektet skal vi lave en webapplikation til vedligeholdelse af en biblioteksdatabase. På websiden skal det være muligt for en bibliotekar at oprette bøger til bibliotekssamlingen, samt oprette hvor mange kopier der er tilgængelige af hver bog. Derudover skal det være muligt at tilføje personer, eller lånere, til databasen, hvor de skal oprettes med kodeord som skal krypteres. Det skal desuden også være muligt for lånere, at låne og reservere bøger som er tilgængelige, samtidig med at bøgerne kan afleveres igen efter udlånsperioden er udløbet. Hvis afleveringsdatoen overskrides med mere end 30 dage, skal personen også tildeles en bøde.

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 vil vi primært gøre brug af Node.js og Express sammen med MongoDB, men også JavaScript og andre teknologier vi har lært eller gjort brug af i forbindelse med undervisningen. Vi vil også gøre brug af teori fra vores undervisningen, og fra relevante bøger eller forums på nettet.

Vi vil i løbet af rapporten komme ind på hvilke metodeovervejelser vi har gjort os, på baggrund af den opgave vi har fået stillet. 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 forsøge at besvare de spørgsmål vi har stillet os selv i vores problemformulering, for at analysere på hvordan vi har prøvet at løse dem.

I konstruktionsfasen vil vi gå mere i dybden med opbygningen af nogle af de forskellige elementer som udgør webapplikationen, og vores primære fokus vil her være på Express og MongoDB, som er de læringselementer opgaven fokuserer på.

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 kan vi med det udleverede repository, der indeholder en Express-applikation som udgangspunkt, lave et website der indeholder fyldestgørende funktionalitet til en biblioteksløsning for både bibliotekar og bruger?

Research

I researchfasen vi lavede, før vi begav os ud i konkret kodning, gik vi igennem vores undervisningsmateriale på dkexit.eu og læste relevante udsnit af bogen “Get Programming With Node.js”. Det var for at repetere de dele af undervisningen, som vi mente kunne være brugbare i vores løsning til projektet. Eksempler på dette er dkexit’s løsning, der hedder “Example A.157”. Eksemplet brugte vi som udgangspunkt til vores egen routing fil; index.js. Da vi fik udleveret et repository, som var udgangspunktet for opgaven, brugte vi figurer i bogen til at forstå mappestrukturen deri, og hvad de forskellige filers formål var, samt hvad vi skulle tilføje/justere for at tilpasse det til vores projekt og dets indhold. Researchfasen fandt sted de første par dage i projektet og blev lavet hver for sig, hvorefter vi samlede os og evaluerede, hvad vi var kommet frem til. Ud fra vores research lavede vi så en tidsplan, som findes under referencer, punkt 3/4, og gik så til arbejde med kodning.

Metode

Da vores biblioteksprojekt skal omfatte mange funktionaliteter, har vi brug for at dele vores indhold op i forskellige mapper og filtyper. Mange funktionaliteter og mange forskellige indholdssider bliver hurtigt uoverskueligt, så ved at inddele projektet i overordnede kategorier kan vi holde styr på hvilke funktionaliteter vi bruger til at holde styr på vores bøger, hvilke der bruges til lånere osv. I konstruktionsfasen vil vi komme nærmere ind på den præcise opbygning af vores mapper.

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. Som det første i dette projekt fik vi udleveret et remote repository som vi skulle klone, for at få adgang til et Express “skelet” af løsningen. Skelettet vi har fået adgang til, indeholder kun en basis opsætning af en Express applikation, samt en backup af en database samt dets collections. Normalt vil vi starte med at bygge en forside op, med menupunkter til de forskellige undersider vi gerne vil have, men i dette projekt var det først vigtigt at få lavet en restore af databasens backup for

at få adgang til indholdet i den. Da Express som standard har lavet en forside til os, har vi valgt kun at bruge den som en indgang til de andre sider vi har brug for i vores projekt. Vores forside indeholder derfor lidt tekst, samt en navigationsmenu til sider som bruges til hhv. at kunne se de bøger som ligger i databasen, at kunne tilføje nye bøger til vores database, at kunne se hvilke lånere eller brugere som databasen indeholder og at kunne tilføje nye brugere til databasen.

Da vi bruger en Express template-engine der hedder pug, har vi selvfølgelig også oprettet flere pug filer, som indeholder vores HTML der vises på vores webside. De to pug-sider vi har oprettet til at fremvise hhv. bøger og personer fra databasen, er bygget op på samme måde. Vi gør her brug af en tabel til at fremvise de data vi gerne vil vise, hvorefter vi looper igennem fx alle bøgerne i databasen for at kunne fremvise alle informationerne.
Informationer vi bruger i loopet kommer fra forskellige JavaScript filer, som vi vil forklare lidt nærmere om senere. For at man har mulighed for også at tilføje bøger eller personer til databasen, har vi oprettet to pug-sider mere til at håndtere dette. Her er opbygningen af de to dokumenter også det samme, da vi i begge tilfælde bruger en ganske almindelig formular med input felter til at registrere de korrekte data vi gerne vil sende videre til vores database.

For at håndtere forbindelsen til vores database og gøre det muligt at læse fra og skrive informationer til den, har vi som sagt lavet forskellige JavaScript dokumenter til at håndtere dette. I tilfældet med vores books har vi gjort brug af tre forskellige JavaScript filer, og med persons har vi gjort brug af to. For at undgå at gentage os selv, fokuserer vi her primært på filerne til at håndtere vores bøger.
For at gøre vores MongoDB databasehåndtering lettere at have med at gøre, bruger vi Mongoose, som er et bibliotek der kan bruges sammen med MongoDB. Mongoose gør det muligt at oprette noget der hedder et “schema”, som bruges til at definere opsætningen af de data man sender af sted til en specifik collection i sin database. Vores første JavaScript dokument, books.js, indeholder derfor et sådant mongoose schema, hvor vi definerer nogle navne til vores data, samt hvilken datatype de kan indeholde. Til sidst laver vi vores schema om til en mongoose model, som vi kan arbejde med i fx vores dokument til at håndtere hhv. vores læsning og vores oprettelse af bøger. Vores andet JavaScript dokument, handleBooks.js, har to funktioner som bruges til både at læse data fra vores books collection i databasen, samt at oprette bøger til den samme collection. I begge tilfælde åbner vi som det første forbindelsen til databasen, da vi ellers ikke har mulighed for at få adgang til det data som er til stede deri.
For at vi kan oprette en ny bog i databasen, har vi i dokumentet importeret vores schema fra books.js dokumentet, så vi kan sætte værdien af vores inputfelter sammen med de korrekte

dele af vores schema når vi sender det over i databasen. Når vi opretter en ny bog, laver vi en variabel med et “new Book” objekt, som fortæller hvordan data fra input felterne hænger sammen med vores mongoose schema’s opsætning. Her er det altså vigtigt at navnene på den ene sider matcher det der står i vores schema, hvor navnene på den anden side skal matche “name” attributen i de forskellige input felter. Til sidst gør vi brug af Mongoose’s “create” funktion, til at sende dataen afsted til vores database og collection og så er der oprettet en ny bog i biblioteket. Samme princip bruges til at registrere nye lånere til persons kollektionen, den væsentligste forskel her er dog at vi samtidig i processen gør brug af et modul som kan hashe/kryptere det kodeord som låneren vælger.

Analyse

I vores problemformulering har vi spurgt os selv hvordan vi kan sikre at vores løsning indeholder funktionalitet som både kan bruges af en bibliotekar og en bruger. Kigger man på vores løsning i konstruktions afsnittet, har vi ikke et fuldt funktionsdygtigt bibliotek hvor man kan låne bøger, reservere osv. Vi har dog implementeret muligheden for at læse hvilke bøger og brugere/lånere, som eksisterer i databasen ved hjælp af html tabeller som gerne skulle være med til at gøre det overskueligt for en bruger og bibliotekar. Derudover har vi lavet simple html formularer til at oprette bøger og brugere til databasen, med tekst inde i input felterne for at fortælle hvilke informationer der skal skrives i hvert felt. For at sikre at de personer der skal oprette nye informationer til databasen, heller ikke glemmer noget, har vi gjort vores input felter required de steder hvor der skal være data. Hvis man ikke indtaster noget information i de felter, kan man ikke få lov til at sende formularen afsted mod databasen.

Konstruktion

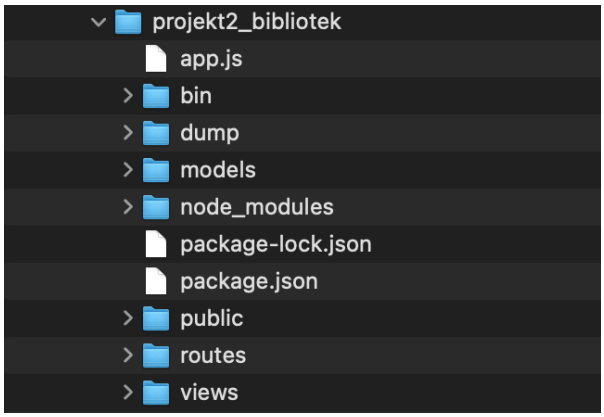
På de fire nedenstående billeder ses mappestrukturen for vores projekt; projekt2_bibliotek. Vi har delt de mappe filer op efter deres sammenhæng og/eller selve filtypen. Konstruktionen er med til at skabe overskuelighed iblandt de mange filer, der kommer med i denne type projekt. Yderst i den overordnede mappe har vi tre filer; app.js, package.json & package-lock.json. JSON-filerne bruges til til at liste projektets scripts, dependencies og fastgøre disse til bestemte versionsnumre. Der bliver i app.js defineret errors/errorhandlers og alle de teknologier, der bruges i projektet og forbundet til vores routes mm.

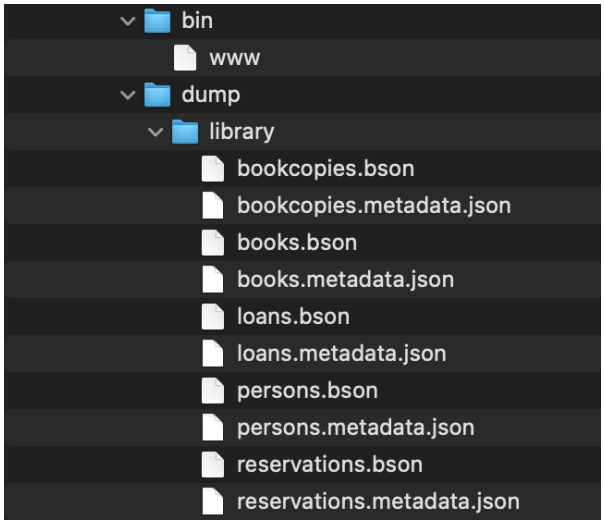
I den øverste mappe, bin, har vi vores www-dokument. Dets primære formål er at lave vores http-server, få porten der skal lyttes til og lave eventlisteners på eventuelle errors der sker med at lytte til porten.

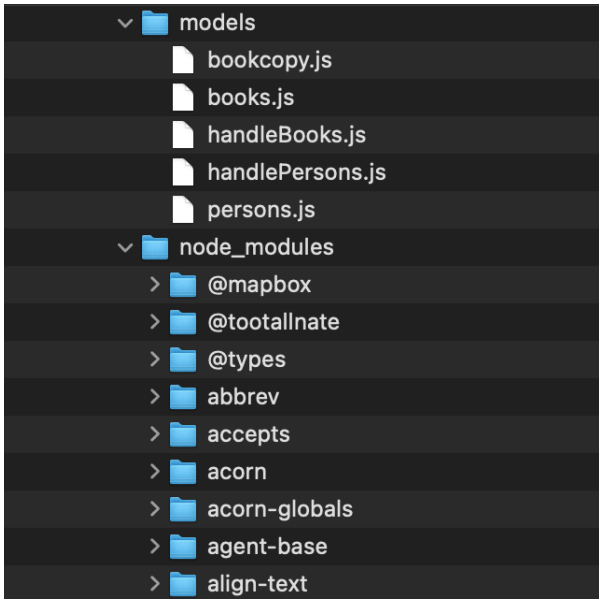
Næst har vi dump-mappen, der indeholder vores library, som er en backup af databasen. Det var en del af det repository vi arbejdede ud fra, så vi har ikke ændret i indholdet. Models-mappen indeholder fem regulære js-dokumenter. De bruges til at oprette nye bøger og personer til databasen, samt at implementere Mongoose i vores projekt. Det er et object data modeling library indenfor Mongodb, som vi blandt andet bruger til at holde styr på relationen mellem datasæt, altså dem vi får gennem vores input felter, som har sammenhæng med dataen der opbevares.

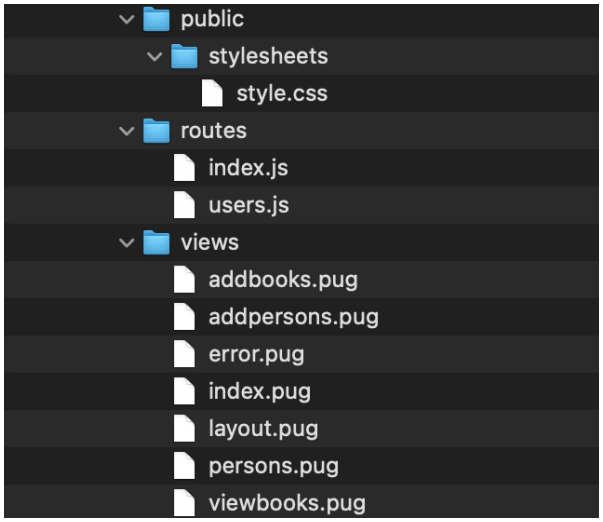
Node_modules-mappen indeholder en lang række node-moduler, som vi bruger til at implementere netop det ind i vores projekt. Mappen indeholder virkelig mange, og det er ikke alle vi bruger, dem vi bruger nævnes i vores dependencies, som tidligere nævnt, i vores JSON-filer, der ligger løst i selve projektmappen, projekt2_bibliotek.

Public-mappen er lavet til stylesheets, som styler indholdet i vores bibliotek. Derunder er routes-mappen, som vi bruger til at håndtere vores GET- og POST-requests. Altså henholdsvis router.get og router.post. De definerer callback-funktioner, som aktiveres når kaldt på via routing. Den nederste mappe i hele vores konstruktion er views. Den indeholder pug-filer, som er en template engine for Node.js. De opbevarer indholdet på bibliotekets sider, altså formularerne på add persons/books, tabellerne på persons/books med mere. Mappen indeholder også forsiden, hvor vi linker til de forskellige undersider.




bin
www
dump
library
bookcopies.bson
bookcopies.metadata.json
books.bson
books.metadata.json
loans.bson
loans.metadata.json
persons.bson
persons.metadata.json
reservations.bson
reservations.metadata.json


models
bookcopy.js
books.js
handleBooks.js
handlePersons.js
persons.js
node_modules
@mapbox
@tootallnate
@types
abbrev
accepts
acorn
acorn-globals
agent-base
align-text


public
stylesheets
style.css
routes
index.js
users.js
views
addbooks.pug
addpersons.pug
error.pug
index.pug
layout.pug
persons.pug
viewbooks.pug

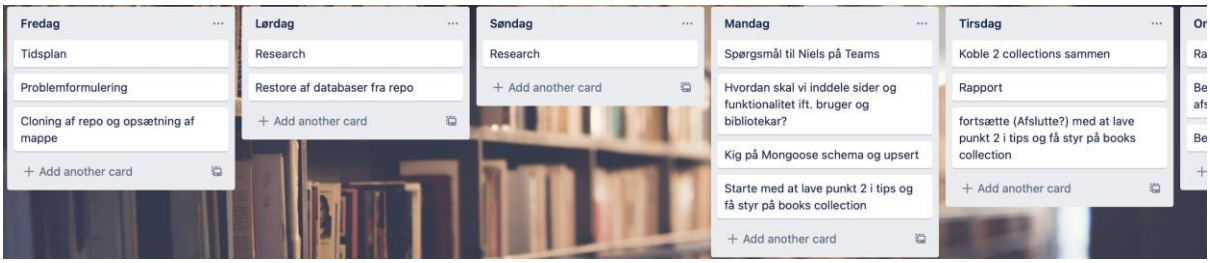
Evaluering af proces

Udviklingsprocessen var udfordrende og det resulterede i at vi mere eller mindre dagligt stødte på problemer, som vi ikke selv kunne løse og derfor fik hjælp for at kunne komme videre. Vi tilsidesatte en del tid i starten og løbende i projektet til at researche løsningsmuligheder, men følte alligevel et stort tidspres. Det afspejler sig selvfølgelig i vores slutprodukt, da vores løsning ikke er fuldstændig funktionsdygtig ifølge projektbeskrivelsens krav og også kunne bruge en mere grundig styling for at forbedre brugervenlighed. Vores proces har derfor ikke været optimal.

Konklusion

Går vi tilbage til vores problemformulering, så har vi ikke fuldført opgaven til punkt og prikke. Vi mangler blandt andet reservationer og en forskel på login mellem bibliotekar og kunde/bruger med mere. Selvom vi prøvede, så var vi sandsynligvis ikke godt nok beredt til at kunne løse opgaven fuldt ud indenfor den givne tidsramme på en uge. Det var både mangel på tid og evner der skabte problemer for os. Vi er klar over at vi skal fokusere på en grundigere forberedelse inden projekt 3, men vi har også et tredje gruppemedlem til den tid, så forhåbentligt vil det lette på nogle af de restriktioner som en tomandsgruppe har givet os indtil nu.

Referencer

1. http://dkexit.eu/webdev/site/index.html (n.d.) *Web Development* [online] available from <http://dkexit.eu/webdev/site/index.html> [14 March 2021]
2. Wexler, Y. (2019) *Get Programming With Node.Js*. Shelter Island, NY: Manning Publications Co
3. 

4. 
Mandag
Spørgsmål til Niels på Teams
Hvordan skal vi inddele sider og funktionalitet ift. bruger og bibliotekar?
Kig på Mongoose schema og upsert
Starte med at lave punkt 2 i tips og få styr på books collection
Add another card
Tirsdag
Koble 2 collections sammen
Rapport
fortsætte (Afslutte?) med at lave punkt 2 i tips og få styr på books collection
Add another card
Onsdag
Rapport
Begynde med punkt 3 (hvis muligt afslutte det samme dag)
Begynde med punkt 4
Add another card
Torsdag
Afslutte punkt 4
Rapport
Præsentation
Add another card