Hovedopgave Datamatiker

Simon Gredal

2024-03-01

Kapitel 1
Indhold

1. Ordoptælling

Tallene her er et estimat fra værktøjet `texcount`, da dokumentet er produceret med XeLaTeX ville det ellers kræve en manuel optælling efter PDF-filen er produceret, fordi i kildeformatet er det svært automatisk at tælle hvor mange tegn en makro producerer, og om de tegn hører til brødteksten eller ej.

```
Total
Number of normal pages (2400 char.)   3.80
Words in text                         1694
Words in headers                      44
Words outside text (captions, etc.)   0
Number of headers                     34
Number of floats/tables/figures       0
Number of math inlines                0
Number of math displayed              0
Files                                 7
```

2. TODOs

3. Indholdsfortegnelse

Kapitel 2
Indledning

1. Introduktion

I noget af min tid på Datamatiker uddannelsen hos Københavns Erhvervsakademi har jeg benyttet Bogstøtten hos Fountain House København.

> Fontænehusmodellen er en international fællesskabsorienteret psykosocial rehabiliteringsmodel, der styrker mennesker i at bryde isolation og til at klare sig selv socialt, økonomisk, uddannelses- og beskæftigelsesmæssigt. Medlemskab er frivilligt og tidsubegrænset, og medlemmer har en høj grad af medindflydelse på alt lige fra dagligdags beslutninger om, hvad menuen i kantinen skal være, til mere strategiske beslutninger i bestyrelsen. Alle er tilknyttet en enhed—i Bogstøtten arbejder medlemmer med egne studier og bidrager også til fællesskab og fælles opgaver, i Ungehuset Fontana arbejder medlemmer kreativt med billedkunst, keramik eller musik og bidrager også til fællesskab og fælles opgaver, i Kernehuset arbejder medlemmer frivilligt i vores arbejdsfællesskaber med at drive café, køkken, kontor og værksted. Nogle medlemmer er aktive i huset hver dag, andre kommer en gang om ugen. Udover dagligdagsprogrammet, har vi et socialt program med fællesmiddag, fyraftensmøder, styrketræning, filmklub, koncerter mm. Vi har omkring 40 medlemmer i huset hver dag og omkring 150 forskellige medlemmer i løbet af en måned. — Fountain House København, *»Om Fountain House København«*

I den tid har jeg opdaget et lile irritationsmoment i forhold til de sociale arrangementer der bliver afholdt. Jeg bragte emnet op, og det endte med

at blive til noget hvor de gerne ville være med til at afprøve idéer til forbedringer og jeg kunne skrive en opgave om det.

2. Problemstilling

Foreningen Fountain House afholder mange arrangementer, både af større og mindre slags. Eksempelvis er der regelmæssige »klubber« (gåklub, filmaften, osv.), diverse møder (interne sofamøder, fællesmøder, åbent hus, osv.), såvel som mere frit planlagte festlige arrangementer (jubilæum, foredrag, julefrokost, osv.).

Til nogle af disse arrangementer er det vigtigt at vide hvor mange der deltager, men andre gange er det vigtigst bare at kommunikere hvor og hvornår det foregår. I dag er løsningen således:

1. Lav et ny Word-dokument.
2. Skriv en overskrift, lidt tekst, og en s.u.
3. Tegn en tabel med plads til navne.
4. Print en håndfuld kopier.
5. Hæng tilmeldingssedlerne fast på opslagstavlerne i bygningen.

Denne løsning er simpel og nemt tilgængelig for husets medlemmer og medarbejdere, men det har også et par svagheder som jeg vil forsøge at opremse her:

- Det kan være svært at vide hvor man skal kigge efter nye arrangementer da der er flere opslagstavler.
- Opslagstavlerne bliver nemt rodede med gamle sedler og plakater som ikke længere er relevante.
- Derfor er det ret nemt at overse nye opslag på opslagstavlerne
- Det ville være godt at bruge mindre papir som bare bliver smidt ud efter 7–14 dage.
- Det kan være besværligt at danne overblik når tilmeldinger er spredt over flere lister.

Fountain House har også en gruppe medlemmer som bruger deres tid på at lave diverse kontor- og IT-opgaver, herunder grafik til plakater, opdatere indhold på webside, samt holde diverse »regnskaber« og Excel-ark ajour. Eksempelvis er man for nyligt begyndt at bruge en tablet i receptionen, hvor medlemmer kan registrere når de ankommer—og evt. når de forlader igen—dette gør det muligt at lave en smule statistik. Statistikken har både til formål at hjælpe med indberetning af antal medlemmer til kommunen i forbindelse med diverse tilskud, men også til at opdage hvis medlemmer begynder at komme sjældnere. Med den oversigt kan medarbejde nemmere gøre noget for at gribe medlemmer der måske er på vej ind i en svær

periode inden de falder helt ud af deres gode vaner.

Så på den baggrund er Fountain House friske på, at køre en test hvor der udvikles et system som kan afløse eller supplere de fysiske tilmeldingsedler. Målet er at få noget erfaring med de fordele som et digitalt system kunne bringe, samt have et øje på de ting som bliver mere omstændige.

3. Problemformulering

Kan jeg med udgangspunkt i en iterativ og evolutionær arbejdsmetode som Scrum, bruge feedback fra medlemmer og medarbejdere i Fountain House, til at udvikle en funktionel prototype på et system hvor medlemmer nemt kan tilmelde sig diverse arrangementer samt opdage nye arrangementer, og kan jeg give medarbejderne mulighed for nemmere at oprette og annoncere nye arrangementer, samt hjælpe dem med at danne sig overblik med tilmeldingerne.

4. Afgrænsning og Metode

Efter som dette er en solo-opgave har jeg besluttet at fokusere på visse dele af projektet end andre; Jeg ville gerne have at processen med at finde frem til en velfungerende brugerflade gennem bruger-testing blev prioriteret, da jeg tænkte at en god måde at få integreret et sådan system er ved at både medarbejdere og medlemmer har lyst til at bruge det. Derfor har jeg også valgt at tage udgangspunkt i Scrum-metoden da den bekymrer sig mindre om fuldkomment definerede artefakter end f.eks. Unified Process, og tilgengæld har nogle regelmæssige ritualer som f.eks. standup og review.

Find kilder på scrum og unified process?

En risiko ved artefakter er også at de risikerer at blive misligeholdt og uddaterede. Dette kan være til stor frustration og forvirring hvis man tror man kan bruge det som dokumentation. En anden tilgang er Literate Programming hvor dokumentation og systemet bygges fra samme kildekode, et eksempel på dette er f.eks. hvis man bruger et rammeværktøj som integrerer med Swagger sådan at når man definerer sine HTTP ruter definerer man samtidigt dokumentationen, eller hvis man bruger et ORM system hvor man både kan producere et ER diagram og SQL DDL.

Find quotes of links til ORM, Swagger, og DDL?

Kapitel 3
Systemudvikling

1. Virksomhedsbeskrivelse

1.1. Interessent analyse

2. Projektplan

2.1. Risikoanalyse

3. Kravspecifikation

3.1. Kravindsamling

Kravene er blevet fundet gennem samtaler med medlemmer og medarbejdere i Fountain House, samt mine erfaringer som medlem i Bogstøtten. Her er de kort opridset som funktionelle og ikke-funktionelle krav.

3.2. Funktionelle krav

Med systemet skal en bruger kunne…

- oprette en konto.
- knytte et navn og billede til deres konto.
- oprette en begivenhed.
- knytte en beskrivelse til en begivenhed.
- tilføje opdateringer til en begivenhed.
- tilmelde sig en begivenhed.
- vise brugere som har tilmeldt sig en begivenhed.
- se de nyeste begivenheder.
- se begivenheder som de har tilmeldt sig.
- se opdateringer på de begivenheder som de har tilmeldt sig.

3.3. Ikke-funktionelle krav

Systemet skal bedst muligt…

- være robust over for brugerinput.
- ligne eksisterende webside.
- have et simpelt og hurtigt flow, der er allerede en »god nok« løsning.
- følge god praksis i forhold til sikkerhed

3.4. User Stories

Her er nogle af de funktionelle krav blevet omskrevet til User Stories, der er blevet udeladt de mest basale krav relateret til konto-håndtering.

> **Som** medarbejder
> **vil jeg** oprette en begivenhed med beskrivelse
> **fordi** andre skal vide hvad der foregår i huset

> **Som** medlem
> **vil jeg** se de nyeste begivenheder
> **fordi** så ved jeg hvad der foregår i huset

> **Som** medarbejder
> **vil jeg** se tilmeldte medlemmer
> **fordi** så ved hvor mange der deltager

> **Som** medarbejder
> **vil jeg** skrive en opdatering på en begivenhed
> **fordi** så jeg kan kommunikere ændringer

> **Som** medlem
> **vil jeg** se opdateringer fra tilmeldte begivenheder
> **fordi** så jeg ved hvis der er ændringer

> **Som** medlem
> **vil jeg** kommentere på begivenheder
> **fordi** så andre kan høre mine input

3.5. ER-diagram

I ånd med at skære ned på artefakter springer jeg domænemodellen over og producerer et ER-diagram. Efter kravindsamlingen og analyses har jeg dannet et godt indtryk af de dele som systemet består af.

ret prisma fil og lav et ER-diagram

Kapitel 4
Implementering

1. Design

1.1. Brugerflade

2. Kode

I dette afsnit vil jeg kommentere på udvalgte dele af kildekoden og retfærdiggøre valget af teknologier og rammeværktøjer og diskutere nogle af de overvejelser der er blevet gjort i den forbindelse.

2.1. Programmeringssprog

Jeg har valgt TypeScript. Jeg har valgt Bun runtime.

2.2. Database

Jeg har valgt at bruge en SQLite database. En af hovedårsagerne er fordi SQLite ikke kræver at der kører en databaseserver ved siden af appserveren, da SQLite »bare« benytter en en enkelt fil på harddisken—en egenskab der også er med til at gøre backup meget simplere.

Find citat fra sqlite.org

2.3. Object Relational Mapping—ORM

Jeg har valgt at bruge en ORM fordi det giver rigtig mange fordele ved at man ikke manuelt behøver skrive DDL eller DML. Man får også, som navnet indikerer, automatisk håndteret vekslingen mellem relationer i databasen og objekter i programmeringsproget, desuden tilbyder en ORM også validering af skema og hjælper med skemamigration. Specifikt har jeg valgt Prisma, fordi de inkluderer et simpelt konfigurationssprog til at definere databaseskemaet og et værktøj som foretager skemamigrationen.

Lav et link til prisma

2.4. GraphQL

GraphQL er et koncept udviklet af Facebook som prøve at gøre op med traditionelle RESTful APIer, og give større fleksibilitet til både frontend og backend teams. Efterhånden som det er blevet mere og mere almindeligt at gå væk fra en »monolit«-struktur og i stedet have separat frontend og backend fandt man ud af at enten kunne man have en API der var meget modulær og løst koblet fra brugerfladen, eller man kunne have en API der var meget skræddersyet og tæt koblet til brugerfladen.

Begge tilgang har sine svagheder, med en løst koblet API ender det med at være meget ineffektivt fordi frontend er nødt til at lave mange forespørgsler til serveren hvor gang en side skal indlæses. Det er både langsomt for slutbrugeren og kan sætte et stort pres på serveren.

Dette er tæt relateret til N+1 problemet i databaser, men det er forværret af at frontend kommunikerer gennem HTTP som har en betydeligt større forsinkelse end en databaseforbindelse.

Omvendt vil en tæt koblet API kræve koordinerering på tværs af teams hver gang der skal ændres i brugerfladen. Koordineringen og overleveringen mellem teams kan virkelig sænke leveringstempoet på ny funktionalitet og fejlrettelser.

Og her kommer GraphQL ind i billedet. GraphQL gør at frontend kan definere præcist hvilke ressourcer de vil se, og hvilke attributer de vil have med. GraphQL-serveren har en såkaldt »resolver« som finder ud af hvordan den bedst muligt kan hente den information fra databasen, om nødvendigt laver den flere forespørgsler til databasen og svarer tilbage med præcist den information som frontend bad om. Nu kan frontend teamet og backend teamet arbejde mere uafhængigt fordi APIen er løst koblet. Men man får stadig fordelen fra en tæt koblet API, nemlig lav forsinkelse fordi der kun sendes én HTTP forespørgsel.

find citat og kode eksempel på graphql, og link til graphql

3. Sikkerhed

Kapitel 5

Afrunding

1. Teori

2. Reflektion

3. Konklusion

Kapitel 6
Bilag

1. Litteratur

Fountain House København: »Om Fountain House København« **fountainhouse**

Fountain House København. »*Om Fountain House København*«. URL: https://fountainhousecph.dk/om-fountain-house/ (hentet 29.02.2024).

Hawking: A Brief History of Time **hawking1988**

Stephen Hawking. *A Brief History of Time. From the Big Bang to Black Holes*. Bantam, maj 1988.

Parenti: Against Empire **parenti2021**

Michael Parenti. *Against Empire*. City Lights, maj 1995.