Programmerings synopsis

What does it contain?

Heading

What is the idea

It contains the process of what to do

It has code examples and explanations

Sources

Diagrams of the app and documentation

Problemformulering I guess

Talk about character raising programmer socks

test

discussion

* Scale--- sql
* Fysiske dimser

Jeg har sammen med Thomas fået ideen om at lave et program til ens køleskab som kan holde styr på de varer man har i køleskabet. Det program kan man bruge til for eksempel at finde opskrifter på internettet automatisk baseret på hvad man har derhjemme, eller man kan gå ind på sin telefon og se om man havde fået købt mælk eller ej.

Proces

Det første jeg gør når jeg skal lave et nyt projekt er at finde ud af hvad det generelle mål er og hvad strukturen ca kommer til at være. Her fandt jeg ud af at jeg ville have en server som skal virke som en database til de køleskabe og brugere der har ting i deres køleskab, og så skulle der være to typer klienter. En type klient skulle være det der kørte på selve computeren i køleskabet, som kan registrere hver gang man sætter noget i køleskabet, og en klient som en bruger kunne logge ind med til at se hvilke varer der ligger i køleskabet.

Når jeg programmere bruger jeg på en måde ”stepwise improvement” metoden, hvor jeg starter med at lave en lille del af programmet og får det til at virke, før jeg går videre med at tilføje flere funktioner. Det jeg startede med, var at lave en server som man kunne tilgå fra en browser, for at få etableret det helt basale, en server og en klient. Derefter fik jeg klienten og serveren til at kommunikere sammen direkte så hjemmesiden kan være dynamisk og så det er lettere at have flere personer på samme tid. Jeg skal også have en database til serveren, sådan så den kan huske hvilke varer der ligger i forskellige køleskabe. Da jeg kun skulle bruge dette til en lille skala valgte jeg bare at opbevare data i en fil på min computer i stedet for at have en ordentlig database. Nu hvor jeg havde de basale ting til en server kunne jeg beslutte hvordan min datastruktur skulle være. Altså jeg lavede et overblik over hvilke data hver ting man kunne have i sit køleskab skulle have, jeg lavede en liste af endpoints som min server skulle svare på med data til clienten, og jeg lavede en liste af ting som klienten skal kunne modtage når serveren sender dem.

Forklaring af programmering

Jeg har valgt at bruge programmeringssproget JavaScript til at lave dette projekt fordi det fungerer godt til webapplikationer. JavaScript er et Objekt orienteret sprog som de fleste store programmeringssprog er, og det betyder at det er baseret på objekter og klasser som er fleksible og kan indeholde store mængder forskellige data. Objekt orienteret er modsætningen til funktionel programmering som er mere fast og bruger funktioner til at behandle data. Det ses typisk brugt i meget optimerede systemer hvor det er vigtigt at alt går meget hurtigt, eller hvor man ikke har så mange ressourcer tilgængelige, som i mikroprocessors. JavaScript er også et high-level sprog som er forskellig fra for eksempel C som er mere low-level. Det betyder at JS er længere væk fra selve den processor den kører på, fordi alt det bliver styret af ECMA Script engine og man skal så ikke selv håndtere ting som memory management og cpu registers.

I JavaScript er der nogle forskellige data typer. Disse inkluderer

* Boolean
* Number
* String
* Object
* Function

Boolean datatypen er den mest simple datatype. Den repræsenterer en enkelt bit, som er en værdi af enten sandt eller falsk, også repræsenteret som 1 eller 0, og tændt eller slukket. En boolean værdi kan bruges direkte ved bare at skrive enten true eller false.

true //sandt

false //falskt

Man kan også bruge en boolean værdi ved forskellige conditions. Det kan man gøre ved at bruge forskellige operatorer som fx større end, mindre end, og lig. De bruges ved at man har en værdi på hver side af operatoren og baseret på de værdier bliver det enten sandt eller falskt

5 > 2 //true

8 < 6 //false

14069 == 4 //false

Number datatypen er et tal. I andre programmeringssprog har man typisk forskellige slags tal, fordi de er brugt så meget. De typiske datatyper er integer, long integer, float, og double som betyder heltal, decimaltal, og decimaltal med dobbeltpræcision. Men i JS består alle tal af doubles, hvilket betyder at et tal kan have en værdi på op til 9007199254740991 (2^53) hvorefter tallet ikke længere er præcist.

100

75

90911901919

var a = 100;

a //100

hvis man gerne vil have tal der er større end 9007199254740991, kan man bruge klassen BigInt, som ikke har en størrelses grænse, men som kun kan have heltal. Den bruger også mere ram og støtter ikke alle operationer.

BigInt(100); //100n

BigInt(129048319048391048910348109348)+BigInt(1); //129048319048391048910348109349n

Datatypen string er en række af bogstaver, som fx et ord, en sætning eller anden kombination af bogstaver og tal. Næsten hver gang man skal have tekst på en skærm bruger man strings til at opbevare den tekst. Der er ikke nogen grænse for hvor store strings kan være. Men man skal selvfølgelig passe på at man ikke løber tør for ram. Et bogstav i JS fylder som standard 2 bytes, så en string kommer til at fylde længde\*2 bytes.

"bogstaver"

"string" + "string" //"stringstring"

var navn = "Nicklas";

var efternavn = "Østerberg";

navn + " " + efternavn //"Nicklas Østerberg"

hvis man vil være lidt mere fancy, kan man også bruge string substitution, som er hvor man har en string og man sætter en anden string ind i den.

`${navn} ${efternavn}` //"Nicklas Østerberg"

JavaScript objektet er nok en af de bredeste funktioner i sproget. Næsten alt man bruger i JS er på en eller anden måde et objekt. Et objekt i sin simpleste form er bare nogle key-value-pairs, altså man har værdier som kan være en af alle JS datatyper, eller med andre ord, kan være lige hvad man vil, og så har man keys som er navnene på de værdier og det man bruger til at referere til de værdier. Keys kan kun være af datatypen string.

var person = {navn: "Nicklas", efternavn: "Østerberg", alder: 18, sej: true, sjovJoke: () => {return "What do you call a fish with no eyes? Fsh";}};