Et billede, der indeholder person, stående

Automatisk genereret beskrivelse

EKSAMENSPROJEKT - PIEFIT

PBA i Webudvikling 1. semester 2021

gruppe 2: Dennis, Freja og Morten

Indholdsfortegnelse

[Indledning med indhold og kort resume 3](#_Toc90536209)

[Problembaggrund 3](#_Toc90536210)

[Problemformulering 3](#_Toc90536211)

[Afgrænsning 3](#_Toc90536212)

[Metodeovervejelser 4](#_Toc90536213)

[Research 5](#_Toc90536214)

[Design 5](#_Toc90536215)

[Programmering 7](#_Toc90536216)

[Analyse 8](#_Toc90536217)

[Design 8](#_Toc90536218)

[Sitemap 8](#_Toc90536219)

[Skitser 9](#_Toc90536220)

[Wireframes 10](#_Toc90536221)

[Prototype/mock up 11](#_Toc90536222)

[Brugertest 12](#_Toc90536223)

[Annoncer 14](#_Toc90536224)

[Programmering 15](#_Toc90536225)

[Databasen 15](#_Toc90536226)

[Forbedringer til Database 22](#_Toc90536227)

[Views fra databasen til frontend. 22](#_Toc90536228)

[Konstruktion 27](#_Toc90536229)

[Ideer til fremtidige funktioner 39](#_Toc90536230)

[Piefit Community 39](#_Toc90536231)

[Indløs kode 39](#_Toc90536232)

[Trofæer 39](#_Toc90536233)

[Statistik 40](#_Toc90536234)

[Evaluering af proces og produkt 41](#_Toc90536235)

[Konklusion 42](#_Toc90536236)

[Referencer 43](#_Toc90536237)

[Creditliste 44](#_Toc90536238)

[Bilag 45](#_Toc90536239)

[Bilag 1 – Tidsplan 45](#_Toc90536240)

[Bilag 2 – Resultater fra brugertest 46](#_Toc90536241)

# Indledning med indhold og kort resume

Piefit er en lille online virksomhed som hovedsageligt henvender sig til fitnessbranchen. På nuværende tidspunkt består Piefits produkt af PiefitCards, som er små fysiske kort på 120 x 120 x 2 mm med diverse træningsaktiviteter. Kortene findes i tre forskellige varianter, Home-edition, Hiit-edition og Power-edition. Hver edition er delt op i fire træningskategorier: core, upperbody, lowerbody og fullbody, som hver har sin egen farve.

# Problembaggrund

PiefitCards er kun tilgængelige som fysiske kort, hvilket i sig selv kan skabe en del ulemper, for eksempel kan man miste sine kort, de kan blive glemt, med tiden bliver de slidte osv. Derfor er det oplagt at søge andre måder til formidling af de fysiske kort. Et alternativ er den digitaliserede verden. Her vil brugeren nemt og hurtigt kunne få adgang til produktet, i form af en webapp til telefonen. Det i sig selv skaber en bedre brugeroplevelse, og man fjerner mange af ulemperne ved fysiske eksemplarer. Desuden har man ofte sin telefon i nærheden, så PiefitCards vil være tilgængelig på farten.

# Problemformulering

Hvordan kan vi digitalisere PiefitCards, så kunden kan tilgå produktet på deres telefon, i stedet for at have fysiske kort?

# Afgrænsning

Vi vil med webappen afspejle at kunden har købt en boks med Piefitkort. Dette vil sige, at vi ikke vil lave selve købsprocessen. I forhold til produktet, PiefitCards, begrænser vi os til varianten HOME edition.

Vi har valgt at inddrage ekstra funktioner, men her har vi dog valgt at begrænse os til at lave disse til vores prototype, hvilket vil sige, at vi ikke kommer til at programmere dem.

# Metodeovervejelser

**Tidsplan:**

Vi har valgt at lave et Gantt-kort som tidsplan (*GANTT-diagram - Hvad er et GANTT-kort?* 2021), fordi det giver os et godt overblik over, hvilke opgaver vi skal have lavet til hvornår.

**ER-diagram:**

Vi udvider ER diagrammet skitseret i projektbeskrivelsen, for at imødekomme nogle af de forskellige features vi vil implementere i opgaven.

**Wireframe/prototype:**

Som en del af for analysen og research arbejder vi med wireframes og prototyper af de forskellige ideer.

**Brugertest:**

Vi vil anvende en brugertest til vores prototype, for at få feedback til vores design og brugervenligheden af denne.

**Figma:**

Bruges som værktøj til skitsering af det visuelle udtryk i form af wireframes og prototype (*Figma: The Collaborative Interface Design Tool.* n.d.). Programmet bruges også til skitsering af ER diagrammet. Udover programmets nemme tilgængelighed, så er en anden fordel at vi alle har arbejdet med programmet Adobe XD, hvilket deler mange ligheder med Figma.

**Node.js:**

Vi bruger node.js til at modificere ”skelettet” af web-serveren (<https://bitbucket.org/phidip/nativeskeleton/src/master/>), samt yderligere implementeringer af diverse funktioner og udviklingen af vores web app.

**Discord/Microsoft Teams:**

Bruges til kommunikation og projektarbejde udover de vedlagte vejledningstimer. Det er også her vi deler og skriver diverse dokumenter, så vi hele tiden er up to date.

# Research

Som en naturlig del af et projekt vil der altid være en del research, for at få indsamlet en masse relevant data, som skal hjælpe med at understøtte arbejdet og diverse valg baseret på dataene, for at opnå det bedst mulige resultat. Dette gøres typisk ved brug af spørgeskemaer eller lignende, men da vi allerede fra start har fået udleveret alt Piefits analysemateriale, består vores research i at gennemlæse de vedlagte filer, og inddrage materialer samt analyser som vi mener er mest nyttige til at løse projektets opgaver.

Nedenfor er et udpluk af Piefits egne materialer, som vi gennem vores research har fundet relevante at benytte til dette projekt.

## Design

**Designmanual**:

Da der allerede er vedlagt en designmanual, som klart beskriver retningslinjerne for valg af farve og skrifttype vælger vi at forholde os til dem, så formidlingen af det færdige projekt ender ud med et produkt, der nemt kan identificeres med det nuværende udtryk af Piefit.

**Logo:**

Vi genbruger Piefits logo i vores projekt, men er opmærksomme på at det har klare regler for hvordan det skal bruges og hvor det skal placeres.

**Billeder/Videoer:**

Vi har været inde og undersøge, hvilke billeder og videoer, som vi har fået tilsendt af Piefits egne materialer. Her har vi drøftet, hvilke billeder man kunne bruge hvor i forhold til vores prototype.

Derudover har vi snakket om at bruge unsplash.com, hvis vi skulle få brug for flere billeder, end dem der er i Piefits materialer. Unsplash.com er nemlig en hjemmeside, hvor man kan hente gratis billeder, som man må anvende (Unsplash 2021).

**Målgruppe:**

Piefits målgruppe er meget bred, i og med at PiefitCards rammer alle personer der ser dem selv som værende aktive i hverdagen. Men samtidig appellerer produktet også til dem som blot har i sinde at være mere aktiv generelt, da det er en meget simpel måde at få pulsen op på, i stedet for at skulle investere i fitnessmedlemskaber eller udstyr. Desuden er PiefitCards også opdelt i niveauer, så der er noget til både begyndere og mere erfarne brugere. Netop det betyder også at PiefitCards har angivet deres målgruppe til både at være kvinder og mænd i alderen 8-65 år, så stort set alle kan være med.

Vores endelige produkt skal derfor kunne afspejle nogle af de valg vi tager, for at ramme så mange som mulige i den brede målgruppe, det kan for eksempel være forskellige funktioner i vores web app, der skal kunne tilgås lige nemt, hvad enten man er 8 eller 65 år.

**PiefitCards:**

Gennem Piefits materialer og egen hjemmeside, har vi fundet frem til, at Piefits produkter hedder PiefitCards, og kan fås i 3 forskellige udgaver alt efter niveau. De 3 udgaver er:

* **HOME-edition:** Den typiske bruger af HOME edition er en kvinde, men også arbejdspladser hvor der indgår meget stillesiddende arbejde.
* **HIIT-edition:** Igen er den typiske bruger her en kvinde, men også mennesker som ikke har adgang eller gider et fitnesscenter eller udstyr.
* **Power-edition:** Denne kategori er typisk til erfarne individer, som allerede er bekendt med fitnesscenteret i dagligdagen, men som søger nye udfordringer eller alternativer til et fitnesscenter.

**Wireframes/prototype**

Til at lave vores wireframes og prototype, vil vi benytte hjemmesiden Figma (*Figma: The Collaborative Interface Design Tool.* n.d.). Dette giver os muligheden for at arbejde i samme dokument samtidigt, hvilket er en stor fordel.

## Programmering

I forhold til programmeringsdelen, har vi snakket om forskellige funktioner, som vi tænkte skulle med i projektet. Hertil har vi være inde og researche lidt, hvordan disse funktioner eventuelt kan løses.

**NPM**

I vores arbejde med node.js har vi gjort nogle overvejelser omkring relevante packages til brug i opgaven, bland andet “Passport” (*Passport* 2021) og “jsonwebtoken (JWT)” (*Jsonwebtoken* 2018) til autentificering af en bruger, men fik en anden løsning ved hjælp af “Cookies” (*Cookie* 2019), som i sin enkelthed gemmer en cookie på serveren når en bruger er logget ind, til bruger identifikation. For brugervenlighed er cookien sat til at udløbe efter 30 dage, derefter stopper sessionen og brugeren skal igen logge ind. 30 dage er sat da webappen sandsynligvis bliver brugt flere gange dagligt af aktive brugere, og her vil det være meget forstyrrende at skulle logge ind hver gang. Cookien slettes ligeledes hvis brugeren logger ud.

# Analyse

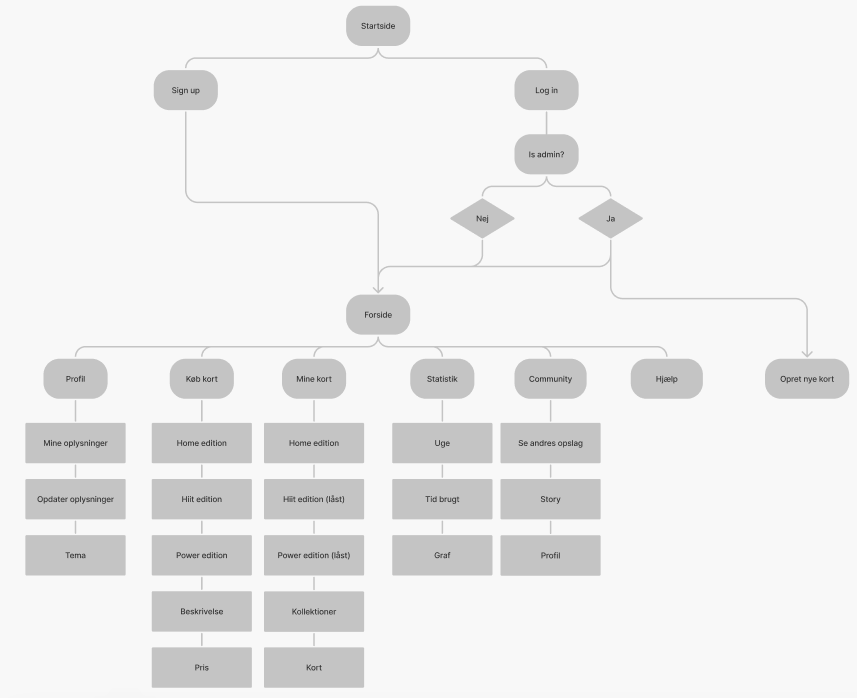
Efter vores research af mulighederne i projektet begyndte vi at analysere hvad der var bedst og mest realistisk at bruge i projektet, for eksempel havde vi under vores research en klar ide om en masse kodning osv. Men efter nærmere analyse besluttede vi os for at begrænse det, grundet tiden, det har også ført til at vi vælger at fokusere på kategorien HOME edition.

## Design

### Sitemap

Som noget af det første, har vi udarbejdet en sitemap. Dette har vi gjort for at få et overblik over, hvilke sider vi gerne vil have med til vores webapp.

Sitemappen er delt op på den måde, at cirklerne repræsenterer de enkelte sider, på nær is\_admin, som er en boolsk værdi i vores database, der skal identificere om brugeren der logger ind, enten har administratorrettigheder eller ej, derefter redigeres brugeren videre. Firkanterne er elementer der skal være til siderne.



### Skitser

Efter vi fik vores sitemap på plads, begyndte vi at lave skitser af udvalgte sider fra sitemappen ovenfor. Her tog vi udgangspunkt i crazy-8-metoden (Google n.d.), hvor vi opdelte vores papirer i otte felter, herefter skitserede vi hver især forskellige udgaver af den udvalgte side. Vores skitser endte med at se således ud:

Et billede, der indeholder tekst

Automatisk genereret beskrivelseEt billede, der indeholder tekst

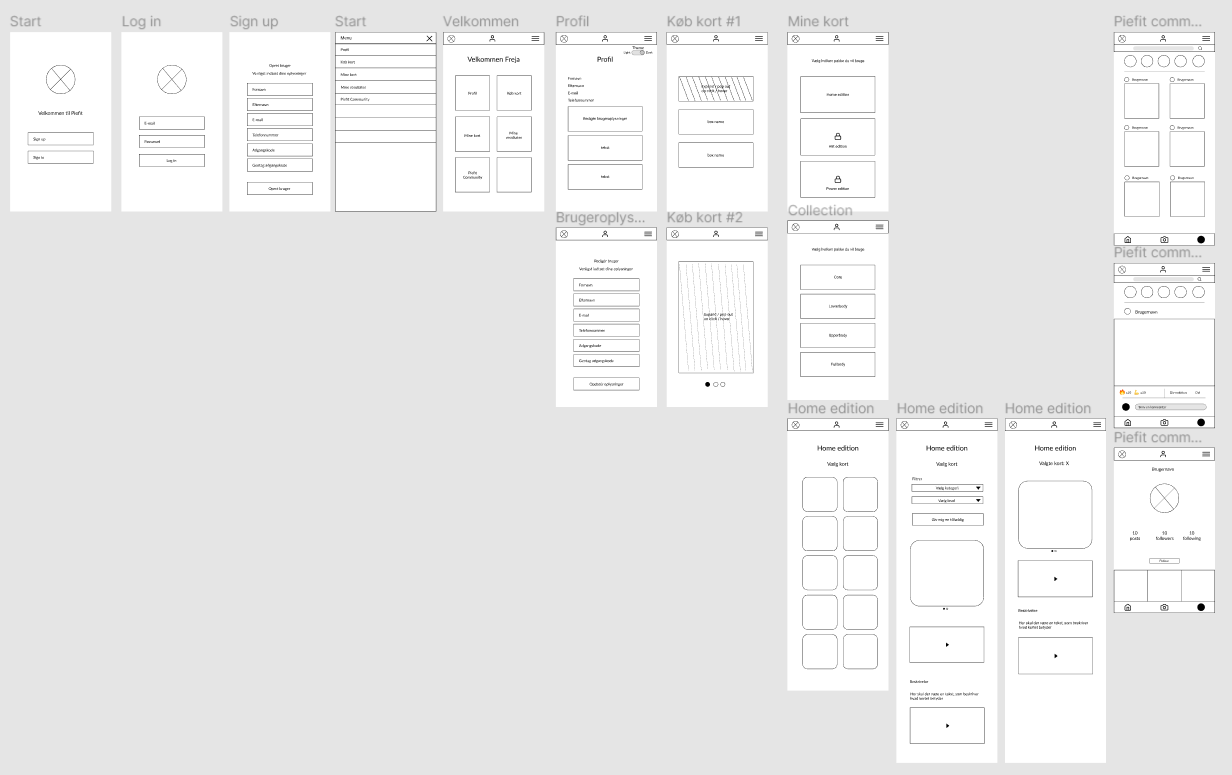
Automatisk genereret beskrivelse

Et billede, der indeholder tekst, whiteboard, sten

Automatisk genereret beskrivelse

### Wireframes

Ud fra vores sitemap og skitser, begyndte vi at lave lo-fi-wireframes for at finde ud af, hvordan vi ville opbygge siderne til webappen. Dette gav os et godt udgangspunkt, til udarbejdelsen af prototypen (Házi 2020).



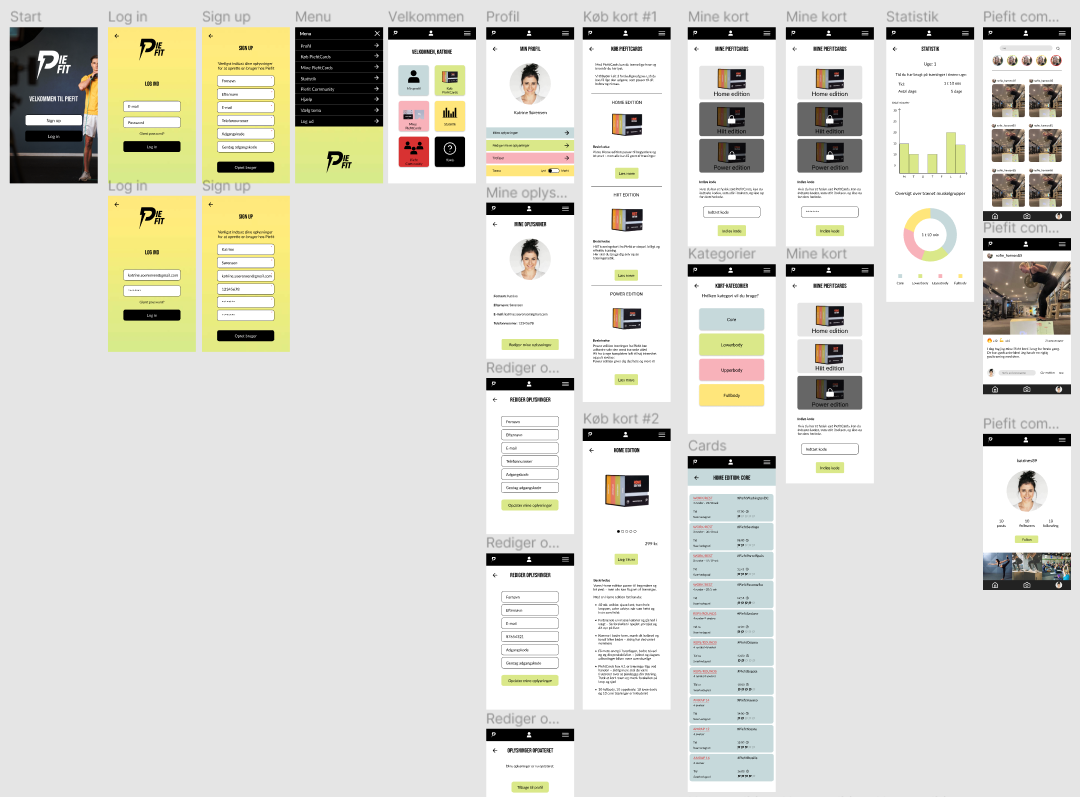
Vores wireframes kan også ses i filen wireframes\_piefit.pdf.

### Prototype/mock up

Efter vi havde lavet vores lo-fi-wireframes, begyndte arbejdet med prototypen. Prototypen bruges som skabelon til det endelige produkt, men også til udarbejdelse af forskellige brugertests senere, for at finde ud af, om vores design er godt og brugervenligt.

Til prototypen har vi haft fokus på at anvende Piefits farver og typografi. Her har vi blandt andet brugt deres designmanual for at sikre, at vi holder os inde for deres retningslinjer. Derudover har vi inddraget de elementer, som vi havde i vores sitemap. Disse informationer har vi implementeret på en måde, som gør det nemt for brugeren at navigere rundt i webappen, da dette medvirker til et brugervenligt design.

I prototypen har vi primært brugt Piefits egne billeder, som vi har fået til rådighed. Dog har vi anvendt et par billeder fra unsplash.com.



Alle vores prototyper kan ses i filerne: prototype\_piefit\_light.pdf, prototype\_piefit\_dark.pdf og admin\_mockup.pdf.

### Brugertest

For at sikre kvaliteten af vores produkt, har vi som nævnt udført nogle brugertests af prototypen for at måle brugervenligheden, og om der eventuelt er fejl og mangler.  
Til vores brugertests har vi valgt at lave en tænke-højt-test, da vi på den måde kan finde frem til, hvad vores brugere tænker og oplever, når de tester prototypen.

Billedet ovenfor er fra Figma, hvor vi har designet prototypen og det kommer med en stor fordel, i og med at hele prototypen kan præsenteres som den eventuelt vil se ud på diverse enheder. Desuden er den fuldt funktionel som var det en rigtig webapp. Der kan klikkes rundt på alle siderne osv.

**Vores brugertest er:**

Tak fordi du har tid har lyst til at hjælpe os med vores brugertest.

Vi har udarbejdet et design til virksomheden Piefit, som sælger fysiske træningskort. Vores design er lavet som en webapp til Piefit, hvor deres brugere kan benytte træningskortene digitalt i stedet for at have dem fysisk.

Vi har i alt 6 spørgsmål, som skal besvares via en tænke-højt-test, hvilket vil sige, at du skal besvare spørgsmålene ved at sige dine tanker højt.

Hvis der undervejs er problemer med designet, så skal du vide, at det ikke er dig der har gjort noget forkert, men at det er vores design, som ikke fungerer optimalt.

**Spørgsmål:**

Hvor mange udgaver findes der af PiefitCards, og hvad hedder disse?

Du vil gerne træne et kort fra Home edition, hvor mange forskellige kategorier kan du træne?

Du synes designet er for lyst, og vil gerne have et mørkt design i stedet for. Hvor kan du gøre det henne?

Du har opdaget, at du har tastet dit telefonnummer forkert. Hvordan vil du ændre det til det rigtige?

Hvor lang tid har du trænet i alt i denne uge?

Hvordan synes du det er at navigere rundt i webappen?

**Resultat af brugertest**

Prøv at registrere dig på Piefit webappen. Hvordan synes du processen var?

* Til dette spørgsmål var alle svar at det var nemt og ligetil at registrere sig på Piefit webappen.

Hvor mange udgaver findes der af PiefitCards, og hvad hedder disse?

* Her svarede alle brugertestere rigtigt med “3 udgaver og Home edition, Hiit edition og Power edition. Nogle fandt det hurtigere end andre, og vi observerede at vores brugertestere fandt svaret 2 forskellige steder.

Du vil gerne træne et kort fra Home edition, hvor mange forskellige kategorier kan du træne?

* Halvdelen af svarene kom hurtigt og sikkert, men den anden halvdel skulle klikke lidt rundt før de fandt svaret.

Du synes designet er for lyst, og vil gerne have et mørkt design i stedet for. Hvor kan du gøre det henne?

* Igen var det splittet mellem vores brugertestere. Halvdelen syntes det var nemt at finde under “profil”, mens den anden halvdel kiggede lidt forvirret rundt i webappen. De fandt dog til sidst ud af at skifte tema under “profil”.

Du har opdaget, at du har tastet dit telefonnummer forkert. Hvordan vil du ændre det, til det rigtige?

* Til dette spørgsmål gik alle brugertestere ind under det rigtige sted med det samme. De syntes de var nemt.

Hvor lang tid har du trænet i alt i denne uge?

* Alle går ind under statistik, som de finder på forsiden. Her kan de hurtigt finde frem til det rigtige resultat som er “5 dage og i alt 1 time og 10 min”

Hvordan synes du det er at navigere rundt i webappen?

* Vores brugertestere syntes det var nemt at navigere rundt. Nogle skulle dog have lidt øvelse først og prøve sig frem, men herefter syntes de ikke det var så svært.

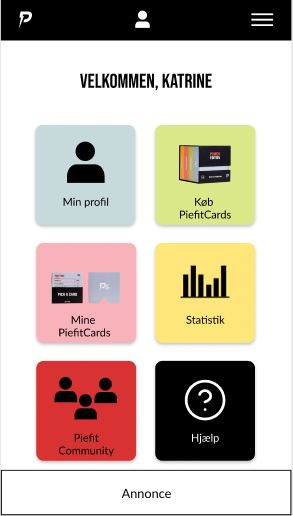
Alle resultater fra brugertesten kan ses i bilag 2.

**Konklusion af brugertest**

Ud fra vores brugertest kan vi konkludere, at vores design og funktioner er til at navigere rundt i. Vi fandt dog ud af, at der var enkelte ting, som skulle gøres nemmere at finde. Dette var for eksempel at skifte tema, hvilket vi så flyttede, så det blev nemmere for vores brugere at finde.

### Annoncer

Som en del af projektet skulle vi i vores design give plads til annoncer. Hertil har vi valgt at sætte fokus på “Mobile Leaderboard” (Bui 2020), som er et bannerformat der er optimeret til mobiltelefoner. Et Mobile Leaderboard-banner har størrelsen: 320 x 50 pixels. Vi har valgt at angive, at en mulig annonce skal placeres nede i bunden af telefonen, som kan ses i eksemplet her:



Grunden til at vi har valgt dette bannerformat og placering er, at det virker mindst forstyrrende. Annoncer har det med at være generende og tage meget opmærksomhed, derfor har vi valgt at gøre annoncen så lidt forstyrrende som muligt.

## Programmering

Som en del af vores research, har vi kigget alle Piefits kort igennem, da vi skal bruge denne information til at oprette et ER-diagram, som senere skal hjælpe os med at oprette databasen.

Som nævnt i vores research har Piefit 3 forskellige udgaver af PiefitCards. Vi har dog valgt at fokusere på én af udgaverne, som er Home edition.

### Databasen

For at skabe vores database, startede vi med at lave et ER diagram (Larsen 2021). Her skabte vi entities, relationships, attributes med cardinalities og participation.

Da vi har lavet vores ER-diagram i Figma, vil for eksempel participation og svage entities/relationships se lidt anderledes ud, end de rigtige. I vores tilfælde er:

*Entitet (stærk) = grøn fyldfarve*

*Entitet (svag) = grøn kant, ingen fyldfarve*

*Relation (stærk) = blå fyldfarver*

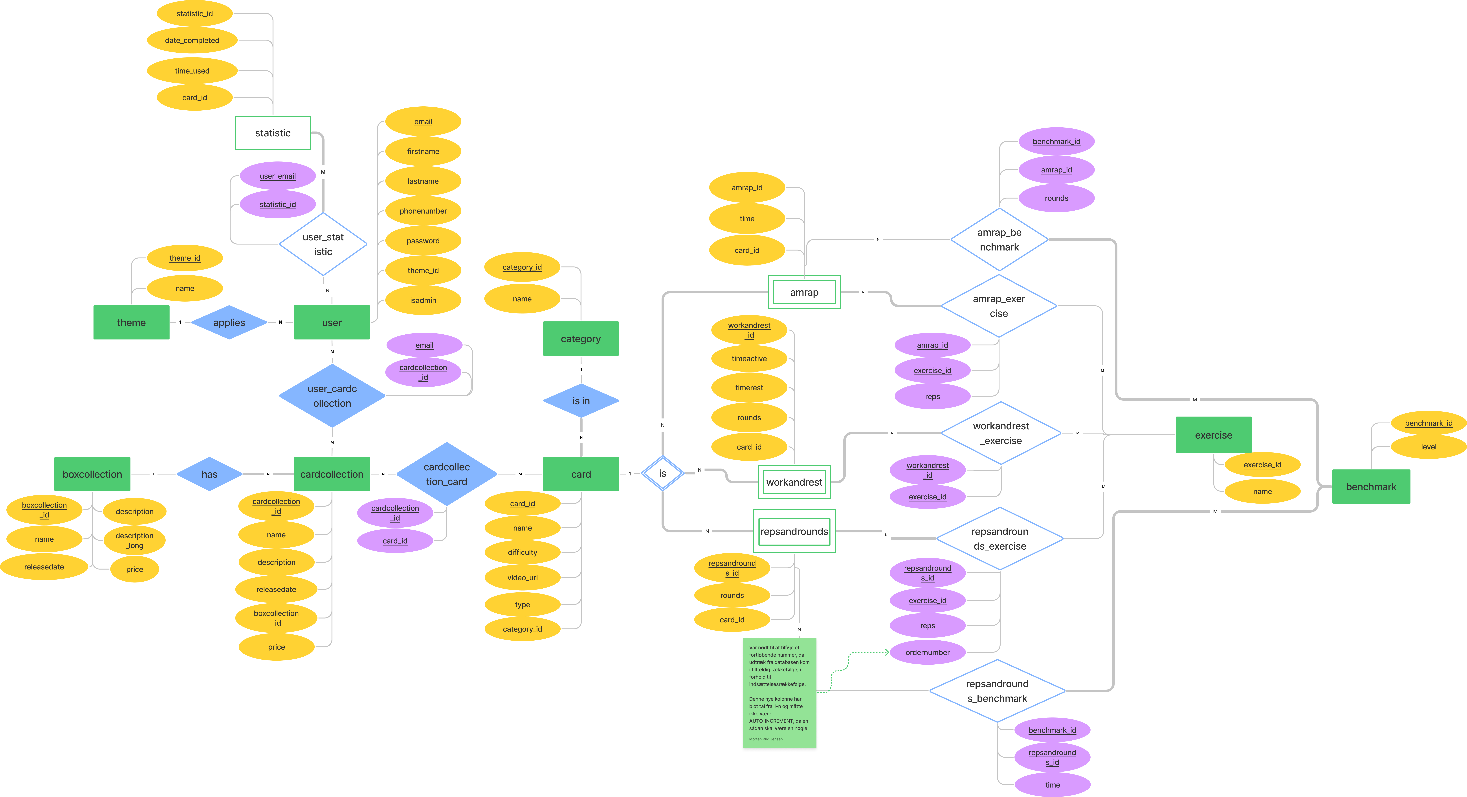
*Relation (svag) = blå kant, ingen fyldfarve*

*Attributter = gul fyldfarve*

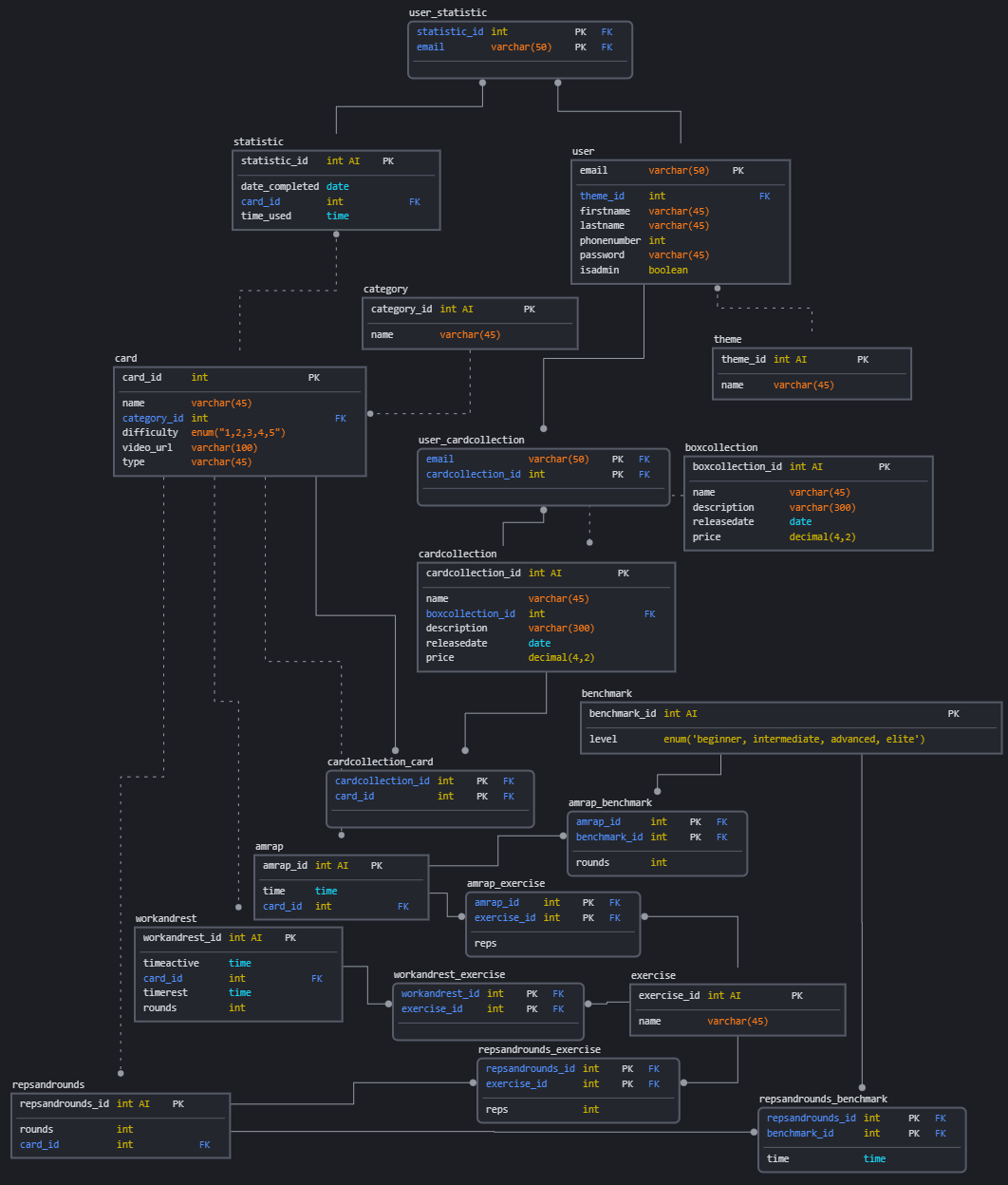
*Attributter på relationstabeller = lilla fyldfarve*

*Tvungen participation = tyk streg.*

Vores ER-diagram tager udgangspunkt i, at en bruger/user skal bruge Piefits webapp, som vi skal udvikle. ER diagrammet ser ud som følgende, og kan ses i fuld størrelse i vedlagte fil: Database\_ER\_diagram.pdf.



Som led i design af databasen benytter vi os af onlineværktøjet SQL DBM (*SqlDBM - Online Database Modeler* n.d.), som kan skabe et overblik over datatyperne, og gøre det lettere at omsætte ER diagram til databasetabeller. Der tages forbehold for enkelte ændringer senere i processen, da databaser altid er dynamiske og sådanne diagrammer kan beskrives som et øjebliksbillede.

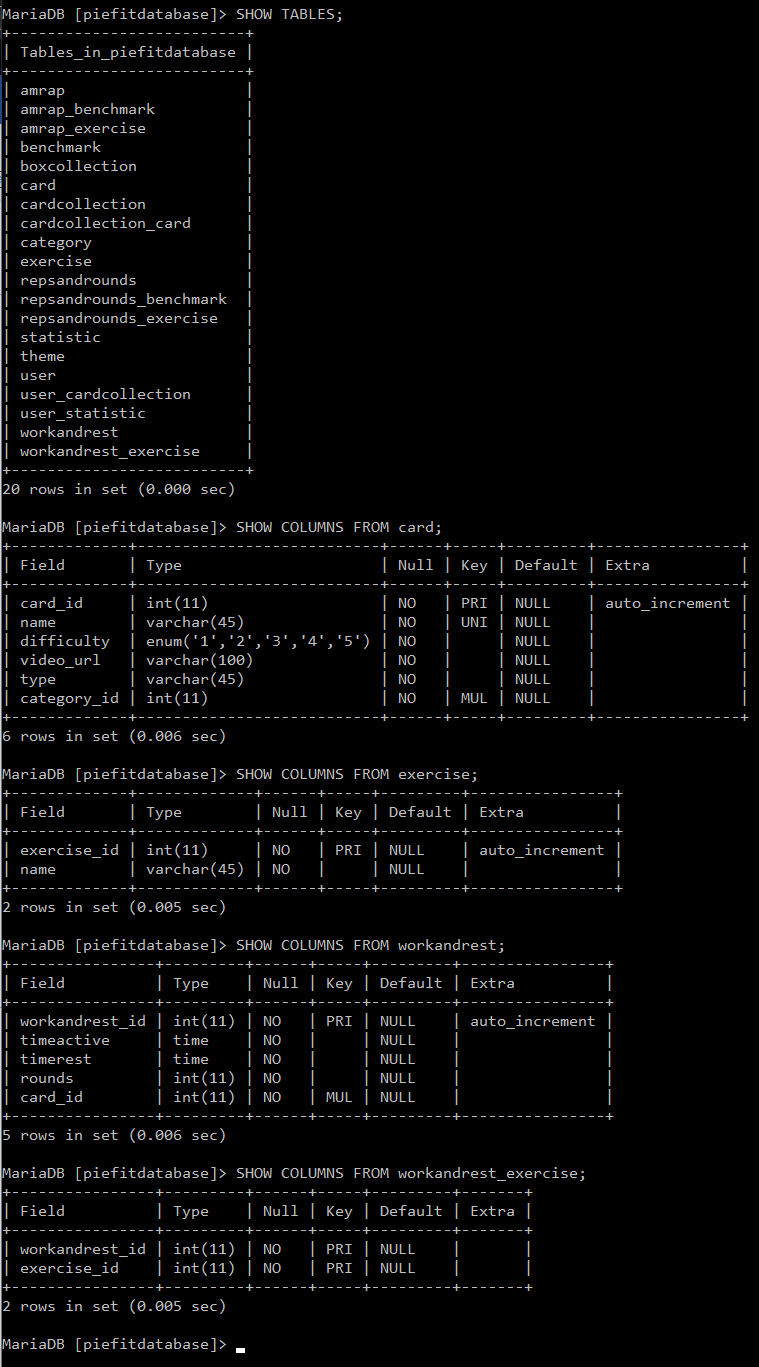


Herefter kan vi begynde at lave databasekoden, som skal oprette databasen, tabellerne og indsætte datatyperne. Her skal nævnes at vi senere i processen har fundet ud af, at der var behov for en user til databasen, som ikke har root adgang til de andre databaser der måtte være i vores eget system. Det ser således ud:

|  |
| --- |
| CREATE USER IF NOT exists piefituser@localhost IDENTIFIED BY 'test';  drop database if exists piefitdatabase;  create database piefitdatabase;  use piefitdatabase;  GRANT ALL privileges ON piefitdatabase.\* TO piefituser@localhost; |

Et udvalgt udsnit af konstruktionen af databasen ses herunder og resten kan ses i piefitdatabase.sql. For at kunne oprette et PiefitCard af træningstypen Work/Rest, er det nødvendigt at have nedenstående sammenkoblinger mellem tabellerne. Da workandrest kun kan eksistere hvis et card er koblet op på det, er det nødvendigt at have en ON DELETE CASCADE, da det ellers ikke er muligt for databasen at slette det tilhørende card. Det kaldes tvungen deltagelse i databasesprog, og workandrest er samtidig en svag entitet, som markeret i ER diagrammet. Ligeledes gælder det for workandrest\_exercise, som er koblingen mellem øvelsestabellen og korttypen Work/Rest tabellen, at det er en svag entitet og kun kan eksistere hvis et træningskort af den type er oprettet og peger på den.

|  |
| --- |
| CREATE TABLE card  (  card\_id int AUTO\_INCREMENT NOT NULL ,  name varchar(45) NOT NULL unique,  difficulty enum('1','2','3','4','5') NOT NULL ,  video\_url varchar(100) NOT NULL ,  type varchar(45) NOT NULL ,  category\_id int NOT NULL ,  PRIMARY KEY (card\_id),  FOREIGN KEY (category\_id) REFERENCES category (category\_id)  ); |
| CREATE TABLE workandrest  (  workandrest\_id int AUTO\_INCREMENT NOT NULL ,  timeactive time NOT NULL ,  timerest time NOT NULL ,  rounds int NOT NULL ,  card\_id int NOT NULL ,  PRIMARY KEY (workandrest\_id),  FOREIGN KEY (card\_id) REFERENCES card (card\_id) ON DELETE CASCADE  ); |
| CREATE TABLE exercise  (  exercise\_id int AUTO\_INCREMENT NOT NULL ,  name varchar(45) NOT NULL ,  PRIMARY KEY (exercise\_id)  ); |
| CREATE TABLE workandrest\_exercise  (  workandrest\_id int NOT NULL ,  exercise\_id int NOT NULL ,  PRIMARY KEY (workandrest\_id, exercise\_id),  FOREIGN KEY (workandrest\_id) REFERENCES workandrest (workandrest\_id) ON DELETE CASCADE,  FOREIGN KEY (exercise\_id) REFERENCES exercise (exercise\_id)  ); |



Efter oprettelse af databasen, kontrollerer vi, at det hele er oprettet korrekt ved at bruge kommandoen SHOW TABLES og SHOW Column from table. For at vide hvordan vi sætter et træningskort af typen Work/Rest ind i databasen fra vores Admin side, er vi nødt til at vide hvilke tabeller der skal i brug, og hvordan INSERT statementet skal formuleres i koden. Derfor laver vi en separat sql-fil med kun koden til at oprette et nyt træningskort med navnet ”#PiefitMyCard”. Når det bliver oprettet i card-tabellen kender vi ikke card\_id, men skal bruge netop dét, for at sætte den rigtige data ind i de næste tabeller, som kobler card sammen med f.eks. exercise. Den viden har vi måtte hente via (*SQL INSERT INTO SELECT Statement* n.d.), og har ved trial and error fundet ud af, at man ikke kan lave multiple value inserts når man gør det på denne måde.

|  |
| --- |
| *-- Create New Card*  INSERT INTO card (name, difficulty, video\_url, type, category\_id)  VALUES  ('#PiefitMyCard', 5, '<https://piefitcards.dk>', 'repsandrounds', 1);    INSERT INTO repsandrounds (rounds, card\_id)  SELECT 8, card\_id from card where card.name ='#PiefitMyCard'; |
| */\**  *insert exercises to the new card, only by knowing the name of the card.*  *We need the repsandrounds\_id, based on the card\_id.*  *Therefore we use a select statement. Other new values is just put after the commas.\*/*  INSERT INTO repsandrounds\_exercise (repsandrounds\_id, exercise\_id, reps, ordernumber)  SELECT rar.repsandrounds\_id, 30, 8, 1  FROM repsandrounds rar  LEFT JOIN card c ON rar.card\_id = c.card\_id  WHERE c.name = '#PiefitMyCard'; |

Ovenstående INSERT må derfor skulle indlejres i et loop, når det skal programmeres i JavaScript, der for hver exercise bliver indsat exercise\_id og reps fra frontend, samt ordernumber som “i” i loop’et. Sidstnævnte kolonne er en tilføjelse, da man fandt ud af at exercises blev printet i tilfældig rækkefølge fra databasen.

Et billede, der indeholder tekst, elektronik, computer, tastatur

Automatisk genereret beskrivelse

### Forbedringer til Database

En forbedringsmulighed på databasen, når man kigger på hvordan user-tabellen er skruet sammen, så burde man have lavet en ID-kolonne, med fortløbende nummer eller anden unik identifikation, da e-mail p.t. er primærnøglen. Hvis e-mail ændres i user-tabellen, skal e-mail også ændres i user\_statistic og user\_cardcollection, da disse refererer til user-tabellen via e-mail, som er nøglen. Spørgsmålet er om det i det hele taget er “lovligt” at ændre på en e-mail når det er primærnøglen. Research af dette har givet svaret nej, og henleder til at man bør have et fortløbende id-nummer på brugere, som er primærnøgle, hvis e-mail skal kunne ændres.

### Views fra databasen til frontend.

Noget af det første vi gjorde, inden vi begyndte at programmere var, at lave en form for pseudoviews fra databasen. Her tog vi fat i vores prototype og noterede ned i klar tekst, hvilke sider der var aktuelle at have med, og hvad der programmeringsmæssigt skulle til, for at få de enkelte sider til at fungere. På den måde analyserede vi og skabte overblik over de elementer som skal være med, for at gøre appen brugbar.

**LOGIN**

Match e-mail og kodeord med user-tabellen. Kodeord skal hashes først og derefter sammenlignes.

Her skal også dannes en cookie, som får user.isadmin. Hvis cookien findes i forvejen, skal den overskrives. Denne cookie skal have et udløb på x-antal dage.

current\_user: Hent alle oplysninger fra user tabel, og læg i et objekt og servér det til siden.

En af de vigtigste funktioner er at få lavet et login, da det er et af kravene i projektet. Til dette har vi dog fået hjælp af Niels, som har programmeret et login for os, som vi har fået til rådighed (Larsen 2021)  
Til login’et havde vi på forhånd tænkt, at der skulle være en form for cookie, som skal sørge for, at brugeren forbliver logget ind. Dette var endnu en funktion, som blev givet til os, som vi efterfølgende har tilføjet ekstra til. Her har vi angivet en timer, hvilket vil sige, at når tiden er udløbet, vil cookien blive slettet, og brugeren skal logge ind igen. Man kan dog diskutere hvorvidt det er nødvendigt med et kort udløb, da denne app vil blive brugt i samme omfang som Facebook osv., som forlader sig på mobiltelefonens egen sikkerhed.

**SIGN UP**

Data fra felterne skal plottes ind i user-tabellen. Password felter skal først matche hinanden, og den regex som bestemmer hvad der må/skal stå. Derefter skal det hashes med bcrypt og lægges ind i user-tabellen.

E-mail skal der også en regex på, så vi er sikre på at den opfylder kravene for en e-mailadresse. Her er det desværre ikke nok at bruge <input type”=email”>, da man kan ændre det i browser inspect tool (*<input Type="email"> - HTML: HyperText Markup Language | MDN* n.d.)

**MENU**

Menuen vil vi hente fra et objekt i sin egen fil, og sætte ind dynamisk på alle sider.

**VELKOMMEN**

Også menu fra samme klasse som ovenfor, men med en ekstra overskrift, som skal hente firstname fra database (current\_user.firstname).

**MIN PROFIL**

Menu for profil med routes.

Tema skift. Ved tryk på slider, opdateres databasen på user.theme\_id.

**MINE OPLYSNINGER**

Vis current\_user details. En brugers oplysninger skal hentes fra databasen og vises på skærmen.

**REDIGER MINE OPLYSNINGER**

I hvert felt skal current\_user egenskaber stå. Når der trykkes OPDATER MINE OPLYSNINGER skal de felter, som er ændret, opdateres i databasen, hvor e-mailen er magen til brugerens. Password opdateres naturligvis med b.crypt, men resten af update-statement ses herunder.

|  |
| --- |
| *--try update on martin*  UPDATE user u SET  u.firstname = "martin spasmager",  u.lastname = "fjollesen",  u.phonenumber = 12345678  WHERE u.email = '[martin@piefitcards.dk](mailto:martin@piefitcards.dk)'; |

**KØB PIEFIT CARDS**

Vis navn og beskrivelse fra de forskellige boxcollections.

**KØB HOME EDITION**

Hent navn, pris, lang beskrivelse fra databasen

**MINE KORT**

Vis alle boxcollections.

Hvis current\_user har en cardcollection med en boxcollection, skal den vises her. Ellers skal der være lås på.

Længere nede er der mulighed for at indløse en kode for at åbne for en anden boxcollection. (Dette er ikke noget vi vil programmere, som beskrevet i afgrænsningen)

**KORT KATEGORIER**

Vis kategorier fra card\_categories.

Hvis man er inde på home, skal der tjekkes om man har adgang til de forskellige collections.

Dette kan evt. checkes op imod en objectvariabel.

**HOME EDITION: CORE**

Her skal alle kort fra Core Home version vises.

Detaljer fra databasen: card.type, card.name, card.difficulty  
Hvis type = workandrest: workandrest.time\_active, workandrest.time\_rest, workandrest.rounds. Time total udregnet fra ((time\_active + time\_rest) \* rounds).

Hvis type = repsandrounds: repsandrounds.rounds, repsandrounds\_exercise(count), repsandrounds\_benchmark.time(benchmark\_id = 1).

Hvis type = amrap: amrap.time, amrap.rounds, amrap\_exercise(count).

**CARD**  
Generelt efter kortvisning: card.video\_url, type video (hardcoded)

WORK/REST:   
**Forside**: type, rounds, “exercises”, time\_active / time\_rest, exercises, name.  
**Bagside:** “References”, “beginner - warm up – finisher", totaltime, video\_url (play video button, rul ned til video), difficulty.

REPS/ROUNDS:

**Forside:** type, rounds, “exercises”, “reps”, exercises + reps, name.  
**Bagside:** “Benchmark”, “beginner - warm up – finisher", “level”, “time”, repsandrounds\_benchmark, difficulty, video\_url (play video button, rul ned til video).

AMRAP:

**Forside:** type, “exercises”, “reps”, exercises + reps, name

**Bagside:** “Benchmark”, “beginner - warm up – finisher", “level”, “rounds”,

amrap\_benchmark, difficulty, video\_url (play video button, rul ned til video).

**START TRÆNING**

Visning af forside og bagside af kortet, som ovenfor.   
Stopur, som gemmer den totale tid af workout – dette skal logges i databasen ved endt træning på statistik når man trykker på “stop træning”. Der er også 2 knapper, som kan starte og stoppe uret undervejs

Hvis træningstypen er workandrest, skal der være et ekstra ur, som tæller ned fra hhv. time\_active og time\_rest. Gerne med en lyd på de sidste 4 sekunder.

**STATISTIK**

Ugenummer (denne uge som standard), “tid du har brugt på træninger i denne uge”. Swipe for at skifte ugenummer, og/eller dropdown på “uge”.

Tid: total for denne uge – udregnes via user\_statstic - statistic.time\_used sat op mod dato.   
Antal dage: ugenummer sat op mod datoer der forefindes i user\_statistic – statistic.date\_completed.

Søjlediagram (svg) som viser antal minutter sat op mod dato mandag - søndag.

Cirkeldiagram (svg) som viser fraktion af tid brugt på hver category i denne uge.

**SERVER, MODELS, CONTROLLERS**

Inden vi begyndte på projektet, fik vi at vide, at Niels ville lave en skabelon til os, som vi skulle bruge. Denne skabelon indeholdt både server, models, controllers (router, cookie, handlers), som vi skal redigere i, så det passer til vores projekt.

# Konstruktion

**MODULER**

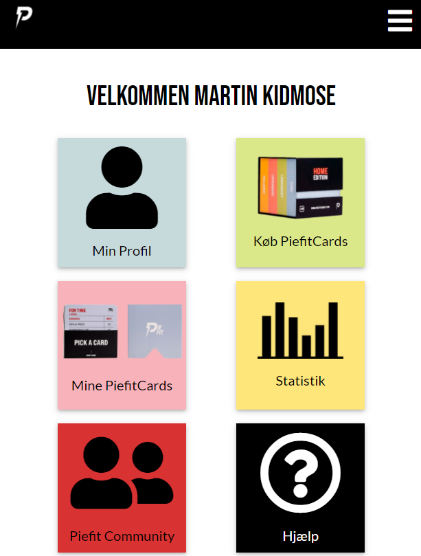
Gennem projektet har vi valgt kun at vil have en JavaScript-fil tilkoblet til hver HTML-side. Dette betyder at vi har anvendt moduler, hvor vi har kunne eksportere og importere diverse funktioner fra forskellige JavaScript-filer.

**DATA FRA DATABASEN**

Vi har to sider, hvor vi skal have informationer fra databasen, til at blive vist på selve siden. Dette gælder vores forside, hvor der skal stå “Velkommen” efterfulgt af navnet på brugeren. Derudover har vi en side under “profil”, hvor brugeren kan se sine egne oplysninger. Disse oplysninger skal også hentes fra databasen.

**VELKOMMEN**

Velkommen siden er den side en bruger der logger ind, bliver omdirigeret til. Siden er opbygget i et grid design med genveje til diverse sider. I toppen af siden er der en dynamisk velkomst, som ændres alt efter brugernavnet.



Navnet er dynamisk fordi det hentes fra cookien efter brugeren har logget ind.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

**MINE OPLYSNINGER**

Under siden “mine oplysninger” kan man se oplysninger som man har indtastet under registreringen. Det er data som vi henter fra databasen via brugerens cookie, som er firstname.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Funktionen userData selecter alt fra den row hvor firstname er det samme som cookien, hvilket vi efter overvejelser og research også har konkluderet er meget dårlig praksis, da det udgør mange sikkerhedsrisikoer, desuden kan der være flere personer med samme navn. Så det mest optimale ville nok være at oprette et unikt userid til hver enkelt bruger.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

For at få dataene ud i HTML’en, bruger vi så “userinfo” fra objectet og idsætter det således <42 userinfo 24>.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Her er så doTheMagic skabelonen der er brugt i dette tilfælde.

**REDIGER OPLYSNINGER**

Når man so bruger er logget ind, skal der kunne redigeres i den oplyste informationer. Det kan gøres via formularen på “rediger oplysninger” siden. Princippet er det samme som når man registrerer sig som bruger på siden, men her vil man bare bruger UPDATE i sql-sætningen, for at opdatere oplysningerne som står i den row som står i brugerens navn.

**TEMASKIFT**

Som et krav til projektet, skulle vi gøre det muligt at skifte mellem to forskellige udseender. Hertil har vi valgt at konstruere det sådan, at man som bruger kan vælge imellem et lyst og et mørkt tema.  
Her har vi først lavet en switch-knap, hvor der er mulighed for at skifte mellem de to temaer. Konstruktionen af selve temaskiftet er:

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Designet af knappen er inspireret af (*How To Create a Toggle Switch* n.d.). Som det kan læses fra ovenstående billede, skiftes der mellem 2 teamer i switchTheme(e), derefter bliver det gemt i localstorage, så det valgte tema også bliver brugt på de andre sider.

Selve temaet farver bliver defineret i CSS således.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

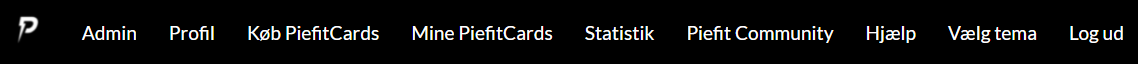
**MENU**

Til at lave vores menu, har vi anvendt JavaScript, hvor vi har flere forskellige filer i gang. Den første fil indeholder et array med objekter med den tekst som skal stå som menupunkt og url’en. Dette array eksporterer vi via moduler, da vi skal bruge det senere i en anden JS-fil.

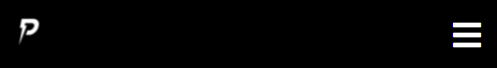
Et billede, der indeholder tekst

Automatisk genereret beskrivelse

I en anden JS-fil har vi funktionerne, hvor vi indsætter selve elementerne i menuen.



**Indsætning af burgermenu**



Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Da vi har lagt fokus på mobilversion og ikke desktop, skal vi have lavet en burgermenu, som vi har konstrueret på følgende måde

Et billede, der indeholder tekst, plade, skærmbillede

Automatisk genereret beskrivelse

**Indsætning af logo i menuen**

Vi opretter først et <li> element, som efterfølgende får tilknyttet et anchor-tag, som skal føre hen til forsiden, når man trykker på logoet.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

**Indsætning af menu-elementer:**

Her bruger vi en “forEach” som looper igennem funktionen og henvender sig til informationerne fra vores tidligere array, hvor den finder oplysningerne “text” og “url”, som den indsætter. På den måde opretter den en menu, med de angivne menupunkter og url’er.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

**Implementering af menu**

For at implementere vores menu har vi i alle vores JS-filer som indsættes i vores HTML’er, importeret menuen, og lavet følgende funktion, som indsætter menuen i en Id på siden:

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

**CARD LIST**

I dette view har det været meningen, at listen over de forskellige kort skulle komme fra databasen, og sættes dynamisk ind fra et loop i JavaScript. Det har vi dog ikke kunne nå, men for at vise, hvordan det skulle se ud, har vi konstrueret det hele direkte i html-viewet.

Et card i listen ser således sådan ud:

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

**ADMIN**

I administrationsdelen er det et krav, at man skal være en administrator, for at kunne få adgang. I databasen har vi på USER en kolonne ISADMIN, som er enten 0 eller 1.

For at give en administrator adgang til administratordelen, og låse denne adgang til andre brugere, som ikke er administratorer, har vi tilføjet et ekstra element til vores menu. Her har vi oprettet et nyt array med et objekt. Denne tilføjelse til menuen kan alle brugere dog se i menuen, men det er kun administratorer som kan komme ind på siden.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Når en ISADMIN logger ind, genereres der sammen med login-cookien en ekstra admin-cookie, som bliver testet på, når routen til /admin kaldes.

|  |  |
| --- | --- |
| async **verifyLogin**(req, res, data) {  let session = cook.**cookieObj**(req, res); *// create session object*  let obj = lib.**makeWebArrays**(req, data); *// home made GET and POST objects*  let r = await models.**verify**(obj);  let isAdminResult = await models.**isAdmin**(obj);  *( . . . . . . . .)*  if (isAdminResult.length == 1 && isAdminResult[0].isadmin == 1) {  session.**set**("admin", true, { signed: true, expires: exp }); *//set isAdmin cookie true*  } else {  session.**set**("admin", false, { signed: true, expires: **Date**.**now**() }); *//set isAdmin cookie false and remove it*  }  (. . . . . . . . .) | |
| async **admin**(req, res) {  if (!(await **isLoggedIn**(req, res))) {  req.url = "/login";  module.exports.**login**(req, res);  } else {  if (!(await **isAdmin**(req, res))) {  req.url = "/velkommen";  module.exports.**velkommen**(req, res);  } else {  let r = await models.**listExercises**(req, res);  let content = "text/html; charset=utf-8";  let path = "views/admin.html";  **getAndServe**(res, path, content, {exercises: r});  }}}, |
| *//returns true or false*  const **isAdmin** = async function (req, res) {  let session = cook.**cookieObj**(req, res);  let admin = session.**get**("admin", { signed: true });  if (admin) return true;  else return false;  }; |

**DYNAMISK INDTASTNING**

Til vores admin-side, hvor man kan oprette et nyt kort, har vi lavet den funktion, at når man indtaster en værdi i et input-felt, vil denne værdi blive vist på kortet med det samme. Dette giver en lækker brugervenlighed, da man kan se resultatet med det samme. For at kunne gøre dette har vi først lavet fire funktioner, som henter diverse HTML-elementer på den tilhørende HTML-side. Dette gøres for at lave nogle shortcuts til senere brug, for at spare kodelinjer.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Derudover laver vi to objekter – en til inputs og en til labels. I objekterne opretter vi keys som svarer til navnene til vores input-felter, så vi får oprettet en til hver.  
Til hver key benytter vi vores shortcut, hvor vi henter elementer fra HTML-siden.

Et billede, der indeholder tekst, plade

Automatisk genereret beskrivelseEt billede, der indeholder tekst, plade, resultattavle

Automatisk genereret beskrivelse

Efter at have oprettet ovenstående objekter, har vi tilføjet en eventlistener på hver af input-felterne. Vi vælger dog kun at vise et eksempel, da de andre konstruktioner er ens. Til eventlisteneren laver vi en funktion som sætter værdien i en data-attribute på den label vi vil ændre til værdien fra inputfeltet.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Et billede, der indeholder tekst

Automatisk genereret beskrivelse



**OPRET NYT KORT I DATABASEN**

For at kunne oprette et nyt træningskort i databasen via frontend, er vi nødt til at vide hvordan det oprettes direkte i SQL. Det har vi undersøgt længere oppe i rapporten. Nu vil vi bruge de informationer til at oprette nogle funktioner, som kan gøre det for os, når vi trykker på “Opret Kort” knappen.

Vi starter med at få et POST kald til routeren.   
Et billede, der indeholder tekst

Automatisk genereret beskrivelse

I “**obj**” findes samtlige indtastninger fra de inputfelter, som er i <form> på admin.html. **Obj** bliver så sendt videre til til HandleSundry.js, som håndterer alle kald til databasen. I funktionen createCard(es **obj’s** egenskabe ud og placeres i sigende variabler. På nuværende tidspunkt har vi kun funktionsdygtig kode til at lægge et card ind, samt lægge dette card ind i den respektive type-tabel.   
Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Læg mærke til, at der i sqlType benyttes en SELECT, for at få card\_id flyttet med over, da vi kun kender til ${hashtag}, som altså er unikt for hvert kort.

Planen er at lave endnu en sql query, som i et loop lægger de 4 forskellige exercises ind i repsandrounds\_exercise tabellen, og koble sammen med repsandrounds tabellen ved at bruge samme select metode som ovenfor.

# Ideer til fremtidige funktioner

Som ekstra har vi fundet på nogle ideer til fremtidige funktioner. Disse funktioner har vi inddraget i vores prototype, så man kan få en bedre forståelse af disse. Vi har dog valgt ikke at programmere disse funktioner.

## Piefit Community

Den første funktion er “Piefit community”, hvor man som bruger inde i Piefit webappen kan tilgå en form for socialt medie for medlemmer. Her er vores tanke, at man som bruger kan poste billeder, følge andre brugere og interagere med andre brugere. På den måde kan Piefits brugere inspirere og motivere hinanden med for eksempel træningen.

## Indløs kode

I dag er PiefitCards et fysisk produkt, og vores projekt går ud på at digitalisere kortene. Men hvad så med dem som gerne vil tilmelde sig webappen, men som har købt de fysiske kort? Bliver de så nødt til at købe dem igen på webappen, for at få kortene digitalt? Til dette har vi fået en ide til en løsning:

Vores ide går ud på, at Piefit kan anvende en unik kode i hver fysisk boks. Denne kode skal kunne indløses på webappen, så man som bruger får låst op for den edition, som man allerede har købt. På den måde kan brugere få sine PiefitCards i en digital udgave, uden at skulle betale dobbelt.

Databasemæssigt ville det betyde en ekstra tabel noget nær sådan: id, code, boxcollection\_id, redeemed\_by\_user.

Denne funktion har vi inddraget i vores prototype, men det er igen en funktion, som vi ikke vil programmere.

## Trofæer

Under “profil” har vi tilføjet en ekstra funktion som hedder “trofæer”. Her er tanken at skabe noget gamification til brugere, hvor man fx kan se, hvor mange gange i træk, man har trænet, hvor mange forskellige træninger man har lavet, osv. Eventuelt at man nu har lavet 1000 burpees i alt.

## Statistik

Tanken bag denne side var at lave noget statistik, fra statistiktabellen, og vise hvor meget man har trænet i den indeværende uge, og med et dynamisk svg genereret søjlediagram vise hvor mange minutter man har trænet hver dag.

Lige nedenunder søjlediagrammet ville vi have et cirkeldiagram, som skulle vise fordelingen af muskelgrupper/kategorier man har trænet over hele perioden. Det har til formål at anspore brugeren til at prøve at fordele sin træning mellem alle fire kategorier.

# Evaluering af proces og produkt

Generelt synes vi at processen gennem vores projekt har gået udmærket. Dog synes vi lidt, at projektet har haft en høj sværhedsgrad, også selvom vi har fået en skabelon at arbejde ud fra. I mange tilfælde har vi ikke kunnet nøjes med at bruge det vi har lært, men vi har måttet lave en masse research til diverse funktioner.

Projektet har været stort nok til, at vi undervejs måtte prioritere vores opgaver, idet vi ikke kunne nå at gennemarbejde alle de ideer vi havde. Men vi synes at vi er kommet godt i mål i forhold til vores egne læringsmål og projektets minimumskrav.

Det har været spændende at have en rigtig virksomhed koblet på projektet. Det har især været fedt, at vi har haft en masse materialer og forarbejde at arbejde ud fra.

# Konklusion

Igennem vores projekt har vi fået lavet en webapp til Piefit, hvor deres brugere kan oprette et login. Ved at logge ind, vil brugerne få adgang til webappen, som kun kan tilgås via login. Gennem webappen har vi fået digitaliseret Piefits produkt, da man som bruger kan gå ind og få vist PiefitsCards gennem webappen. Derudover er det muligt for en administrator at gå ind og oprette et card. Dog er det ikke fuldt implementeret med databasen.

Ud fra vores brugertest fandt vi frem til, at vores design til webappen var nem at navigere rundt i – nogle skulle dog bruge mere tid end andre. Vi fandt ud af, at nogle få ting skulle gøres mere tydelige eller der skulle være flere veje, for at komme derhen, hvilket vi efterfølgende har rettet til.   
På baggrund af vores afgrænsning kan vi konkludere at vi har opfyldt de vigtigste projektkrav.

# Referencer

PieFits filer

*<input Type="email"> - HTML: HyperText Markup Language | MDN* (n.d.) available from <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/email> [16 December 2021]

Bui (2020) *8 Best Mobile Ad Sizes for 2021* [online] available from <https://directiveconsulting.com/blog/mobile-ad-sizes/> [13 December 2021]

*Cookie* (2019) available from <https://www.npmjs.com/package/cookie> [16 December 2021]

*Figma: The Collaborative Interface Design Tool.* (n.d.) available from <https://www.figma.com/> [16 December 2021]

*GANTT-diagram - Hvad er et GANTT-kort?* (2021) available from <https://dinero.dk/ordbog/gantt-diagram/> [16 December 2021]

Google (n.d.) *Crazy 8’s* [online] available from <https://designsprintkit.withgoogle.com/methodology/phase3-sketch/crazy-8s> [7 December 2021]

Házi, C. (2020) *7STEPUX: The Complete UX Process from Strategy to Design*. 3. udgave.

*How To Create a Toggle Switch* (n.d.) available from <https://www.w3schools.com/howto/howto\_css\_switch.asp> [16 December 2021]

*Jsonwebtoken* (2018) available from <https://www.npmjs.com/package/jsonwebtoken> [16 December 2021]

Larsen, N.M. (2021) *11.4. Entity Relationship Modelling, ER* [online] available from <http://dkexit.eu/webdev/site/ch11s04.html> [9 December 2021]

Larsen, N.M. (2021) *Phidip / Nativeskeleton — Bitbucket* [online] available from <https://bitbucket.org/phidip/nativeskeleton/src/master/> [16 December 2021]

*Passport* (2021) available from <https://www.npmjs.com/package/passport> [16 December 2021]

*SQL INSERT INTO SELECT Statement* (n.d.) available from <https://www.w3schools.com/sql/sql\_insert\_into\_select.asp> [16 December 2021]

*SqlDBM - Online Database Modeler* (n.d.) available from <https://app.sqldbm.com/> [16 December 2021]

Unsplash (2021) *Beautiful Free Images & Pictures | Unsplash* [online] available from <https://unsplash.com/> [15 December 2021]

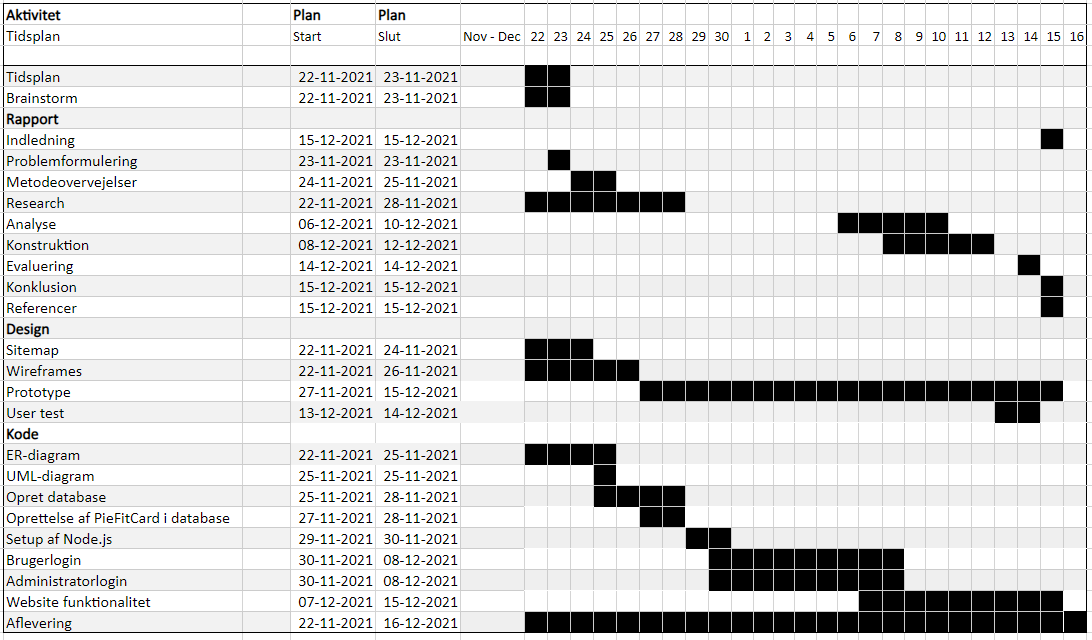
# Creditliste

Vi har lavet en creditliste over dem, som har taget de billeder vi har anvendt til vores prototype.

|  |  |
| --- | --- |
| **Bruger** | **Link** |
| Jake Nackos | https://unsplash.com/photos/IF9TK5Uy-K |
| Edward Cisneros | https://unsplash.com/photos/\_H6wpor9mjs |
| Emily Sea | https://unsplash.com/photos/coiWR0gT8Cw |
| Meghan Holmes | https://unsplash.com/photos/wy\_L8W0zcpI |
| Inspired Horizons Digital Marketing | https://unsplash.com/photos/kufLAiPoIlI |

# Bilag

## Bilag 1 – Tidsplan



## Bilag 2 – Resultater fra brugertest

**Resultat af brugertest**

Prøv at signe up til Piefit webappen. Hvordan synes du processen var?

* Det var nemt
* Ligetil og nemt
* Nemt nok
* Det var nemt

Hvor mange udgaver findes der af PiefitCards, og hvad hedder disse?

* Jeg kan hurtig se at der er 3 udgaver
* Home, Hiit og Power
* Jeg går med det sammen under “køb piefitcards” - der er 3 udgaver (home, hiit og power)
* Der er 3 udgaver (home edition, hiit edition og poweredition) (fandt svaret under “mine piefitcards)

Du vil gerne træne et kort fra Home edition, hvor mange forskellige kategorier kan du træne?

* 4 (hurtigt og sikkert svar)
* 4 (hurtigt og sikkert svar)
* Jeg kigger videre under “Køb piefitcards” og klikker ind på “home edition”, men kan ikke se informationen der. Klikker tilbage på forsiden og går ind under “mine PiefitCards” og finder at der er 4 kategorier.
* Kan ikke rigtig finde det. Prøver at klikke sig frem. Går ind under “køb piefitcards”, hvorefter derstadig klikkes rundt. Finder til sidst ud af, at der er 4 kategorier. Det var ikke så nemt, fordi jeg allerede stod det rigtige sted, så jeg var på vildrede.

Du synes designet er for lyst, og vil gerne have et mørk design i stedet for. Hvor kan du gøre det henne?

* Under profil – nemt at finde
* Det var nemt at finde frem til
* Kigger forvirret rundt, og ved ikke helt hvor der skal klikkes. Først klikkes ind på Piefit community. Kan ikke finde ud derfra. Med hjælp finder man ind på profil, og får ændre temaet.
* Klikker på burgermenu – klikker ind på piefit community. Finder ind på min profil, og klikker ind på “rediger mine oplysninger” og finder den herefter inde på min profil.  Jeg synes ikke det giver mening at have tema under profil. Måske det var bedre at sætte den i bunden på forsiden eller måske i burgermenuen.

Du har opdaget, at du har tastet dit telefonnummer forkert. Hvordan vil du ændre det, til det rigtige?

* Jeg vil gå ind på min profil, og kan se at der er en “rediger mine oplysninger”, som jeg vil trykke på. Her vil jeg ændre det
* Jeg vil gå ind under min profil. Her kan jeg se at jeg kan redigere mine oplysninger og klikker derind.
* Jeg går ind under min profil, og går ind under rediger mine oplysninger – det var nemt

Hvor lang tid har du i alt trænet i denne uge?

* 5 dage og har trænet 1 time og 10 min
* Jeg tænker at jeg skal ind under statistik og kan se at jeg har trænet i 1 time og 10 min.
* Går først ind under “mine oplysninger”, da jeg står der. Går herefter ud på forsiden og ind i statistik. Her kan jeg se at jeg har trænet i 5 dage og i 1 time og 10 min.

Hvordan synes du det er at navigere rundt i webappen?

* Det er meget nemt at navigere rundt, det er meget minimalistisk
* Meget nemt og overskueligt at navigere rundt
* Når jeg får prøvet mig frem, er det ikke så svært
* Med lidt øvelse skal det nok gå. Når det er nyt, skal man lige ind og kigge hvad der gemmer sig bag.