
Projekt 1 - Krisen kradser

Efterårssemester 2024

Jeppe Bøgeskov Bech
jepp9920@zbc.dk

Alexander Schade Knudsen
alex245h@zbc.dk

Andreas Jensen
andr328q@zbc.dk



**Vær med.
Verden er til at forandre.**

2. x

ZBC Handels- og Teknisk gymnasium Slagelse
Akademisk år 2024-2025
8. december 2024

1 Abstract

This report encompasses the project of developing an application that can be used to manage crises. The application will be developed by a new company called "KEP". The application will be developed using a modern application development solution, ensuring optimal performance and compatibility.

2 Forord

Vi vil gerne rette en tak til Morten Bach Jensen, cand. hort. Morten hjalp os med vores aogrødeberegner, han hjalp os også med information om sæddosering samt arealallokering, anvendt til beregning.

Vi vil endvidere rette en tak til Jacob Søgaard, Ph.d Video Quality Assessment, som har hjulpet os med viden om hvordan vi møder vores krav om at kunne gemme videoerne på en optimal måde, så videoerne fylder mindst muligt og kvaliteten er højest muligt.

Sidst, men ikke mindst vil vi takke vores lærer, Henrik Neumann Poulsen, for sin støtte og sine gode råd igennem hele projektet.

Indhold

1 Abstract	i
2 Forord	ii
3 Projektstyring (meta)	1
3.1 Rollefordeling	1
3.2 Tidsplan	1
3.3 Projektmappe	1
4 Problemidentifikation	2
4.1 Idegenerering	2
4.1.1 Mindmap	2
4.1.2 Lyskurven	2
4.1.3 Identificering af nøgleproblem	3
5 Problemanalyse	4
5.1 Problemtræ	4
5.2 Kvalitativ metode	5
5.3 HV-modellen	5
6 Produktprincip	6
6.1 Målgruppe	6
6.2 Kravspecifikation	6
6.3 Konkurrenter	6
7 Produktudformning	7
7.1 Overordnet	7
7.2 Kodestack	7
7.2.1 Framework: React Native	7
7.2.2 Sub-Framework: Expo	7
7.2.3 Runtime: Node.js	7
7.2.4 Database: MongoDB*	7
7.2.5 Programmeringsprog: Typescript	7
7.3 Brugergrænseflade	8
7.3.1 Konceptdesign	8
7.3.2 Design	8
7.4 Produktgennemgang	8
7.4.1 Filstruktur	8
7.4.2 Bibliotekskoden	9
7.4.3 E-beregner	10
7.4.4 Afgrødeberegner	12
7.4.5 Indstillinger	13
8 Produktionsforberedelse	16
8.1 Serveropsætning	16
8.2 Betatesting åben Quality Assurance	16
9 Produktrealisering	17

9.1	KEP-branding	17
9.2	Profiteringsmodel	18
9.2.1	Etik og ansvar	18
9.3	Appmarkedsdistribution	19
9.3.1	Apple App Store	19
9.3.2	Google Play Store	19
9.3.3	Egendistribution - apk	19
9.4	Viderdownload-prompt	19
10	Selvevaluering (meta)	20
11	Post scriptum om visionen for produktet	21
Litteraturliste		22
Appendiks		22
A	Projektbeskrivelse	23
B	Konceptart	25

3 Projektstyring (meta)

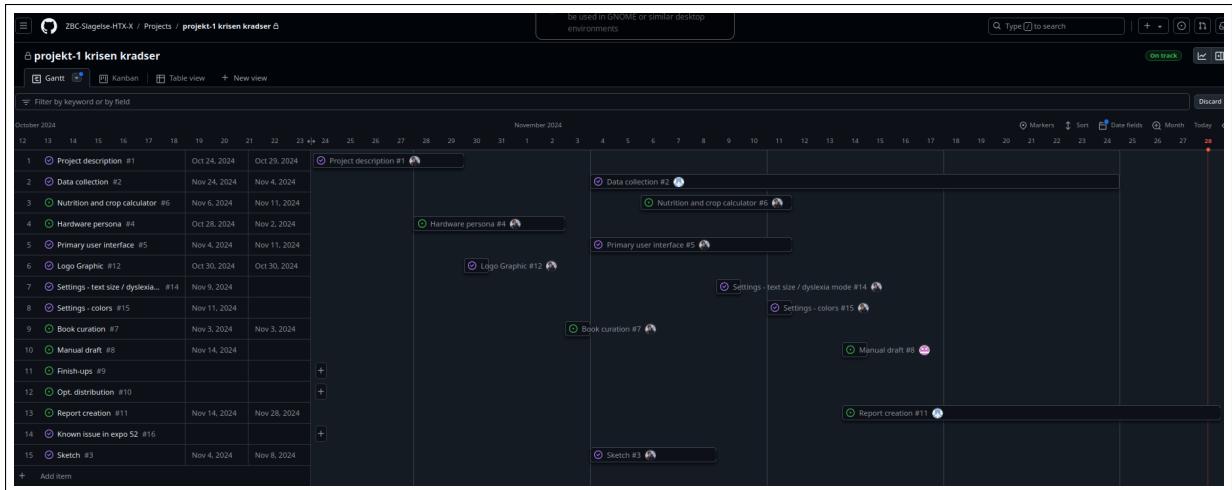
3.1 Rollefordeling

	Roller:
Alexander:	Visionsansvarlig, dokumentarist, strukturering
Andreas:	Konceptartist, informationssøgning
Jeppe:	Produktansvarlig, programør, artistsupervisor, vision

Tabel 1: Viser rollefordelingen for vores projekt.

3.2 Tidsplan

Vores tidsplan er håndteret via GitHub's indbygget funktion, se forneden, dog er den opdateret version her:



Figur 1: Viser tidsplanen for projektet.

Link til aktuel tidsstyring:

3.3 Projektmappe

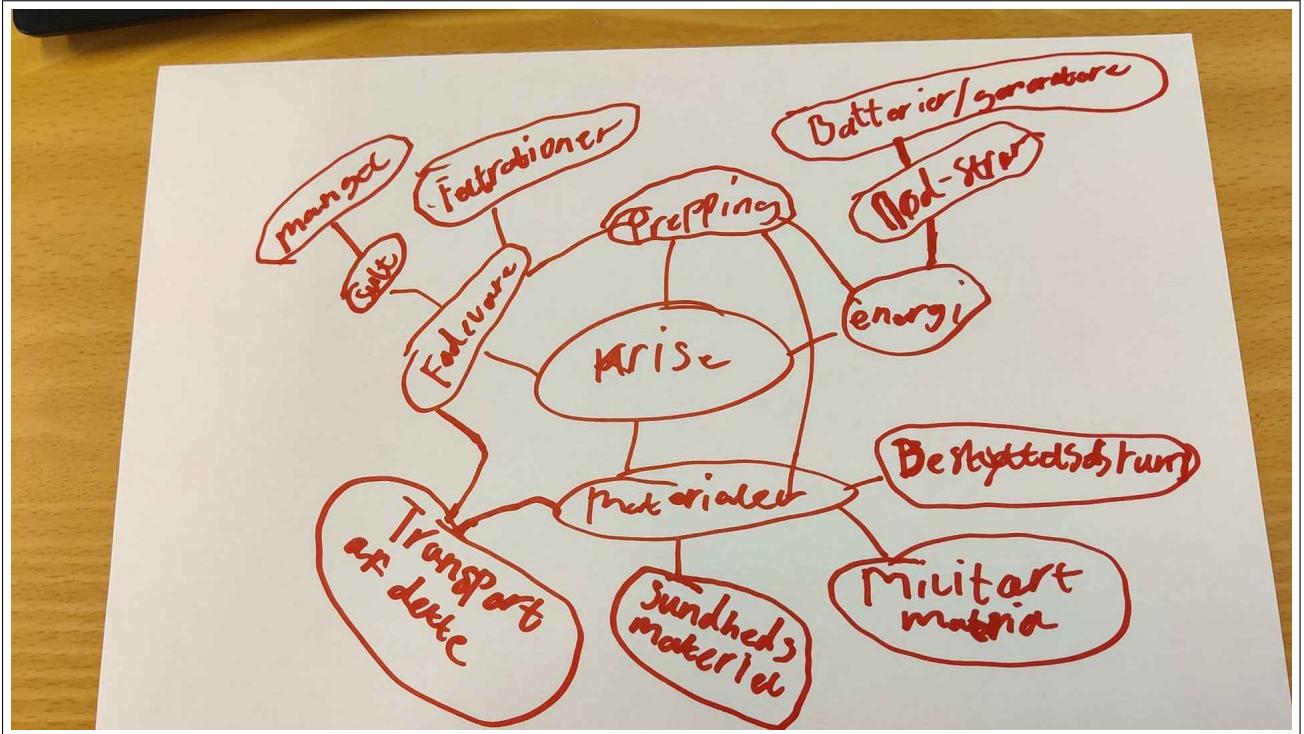
Link: [Vores GitHub fil](#)

4 Problemidentifikation

4.1 Idegenerering

4.1.1 Mindmap

Vi har valgt at lave et mindmap, da dette er en effektiv måde at generere ideer på, og få et overblik over hvilke emner der er relevante at beskæftige sig med.



Figur 2: Viser vores mindmap

4.1.2 Lyskurven

Lyskurven er en metode, man kan bruge til at sortere ideerne i forhold til hvilke der er mest relevante. — Bla bla waffel mere on lyskurvediagrammet —

Fødevarer	Grøn
Beskyttelsesrum	Gul
Nødstrøm	Rød

Tabel 2: Viser et meget abstrakt lyskurvediagram i form af en tabel; det vi anvendte

4.1.3 Identificering af nøgleproblem

Her anvendes spørgsmål til at sikre, at det umiddelbart interessant emne jf. lyskurven også rent faktisk er interessant:

...

følgende spørgsmål:

1. Hvorfor er det her interessant?
2. Hvem er det interessant for?
3. Er det noget, vi laver for vores egen fornøjelses skyld?
4. Er det noget, som en bestemt gruppe i samfundet kan have gavn af, eller er det noget, der er til gavn for alle?

(Spørsmålene, som her gøres brug af, er hentet fra systimebogen¹)

Besvarelsen på disse spørgsmål ser således ud:

1. Krisehåndtering er et interessant emne, da det har en stor betydning for alle i samfundet.
2. Det er relavant for alle.
3. Nej, ideen med produktet er at kunne hjælpe almindelige mennesker med at håndtere kriser, mindre som store.
4. Produktet skulle kunne gavne alle, som kunne stå i en krisesituation.

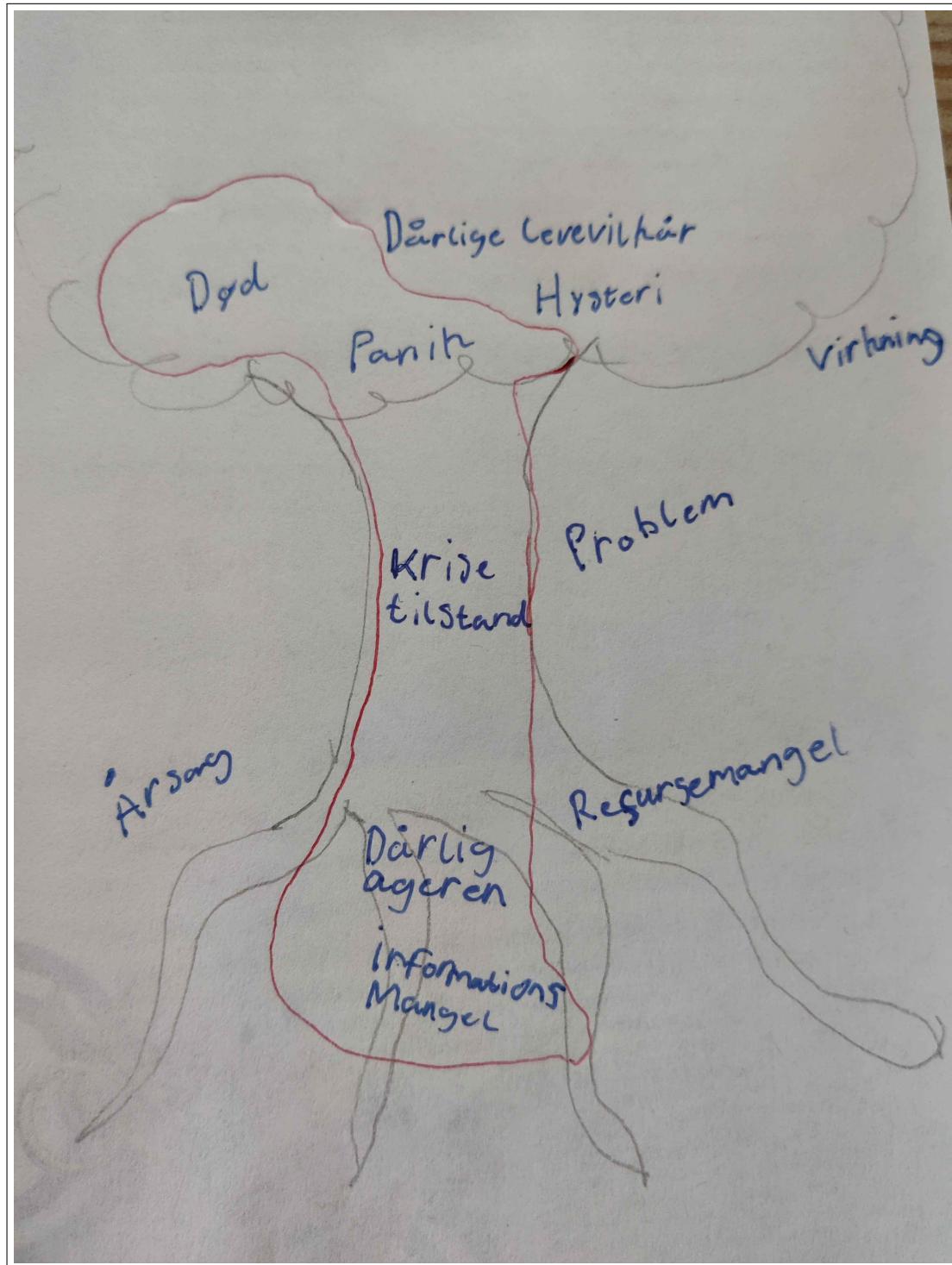
Her ses, at det mest interessante emne, jf. overstående, er krisehåndtering, da det både har en så stor relevans, herunder for mennesker i krisesituationer, og samfundet, og har en reel efterspørgsel udover den intrinsiske.

¹Systimebogen: Projektarbejdet > 3. Problemidentifikation > 3,3 identificering af nøgleproblem | ID 271 [1]

5 Problemanalyse

Problemanalysen tager udgangspunkt i nøgleproblemet krisehåndtering jf projektbeskrivelsen, der kan findes som appendiks (A) samt kan man læse kapitlet om problemidentifikation (4).

5.1 Problemtræ



Figur 3: Viser vores problemtræ

5.2 Kvalitativ metode

Beredskabstyrelsen har udarbejdet og sendt en opfordring til alle danskere om kriseparathed, hvor de tydeligøre vigtigheden i at være beredt på at kunne håndtere en eventuel krisesituation. De beskriver i denne, hvordan danskerne skal kunne klare sig selv i 3 døgn, og hvad der skal til for at dette er muligt. I vores husstande, i det private, er disse opfordringer blevet taget meget seriøst. Men hvad gør man, hvis man løber tør for drikkevand, mad eller lignende? Man kan nødvendvis ikke tilgå hjælp via internettet. Vi så her en god mulighed, med vores appudviklingsenskaber, at kunne løse dette problem med at lave en app, der kan hjælpe med at håndtere kriser, både forberedene, på kort sigt og på lang sigt.

5.3 HV-modellen

Ligedan er HV-modellen blevet brugt for at konkretisere, hvilke trin som tages for at udføre vores projekt.

Hvad? - Det skal gøres overskueligt og konkret, hvordan man skal handle i en krisesituation.

Hvorfor? - Det skal gøres, så man kan være beredt i en eventuel krisesituation. Fra et firmasynspunkt er der et stort udbyttepotentiale, da det må forventes, at folk værner om sit og sine næstes liv. Desuden er det sandsynligt, at eftersom beredskabstyrelsen har varslet information om kriseparathed [2], at man via en aftale med staten eventuelt kunne få et subsidium til udarbejdelsen af en sådan applikation, som heri benævnes.

Hvem? - Projektet skal udarbejdes af en nyopstartet virksomhed, her fiktivt, "KEP". Se mere om "KEP-størrelsen under kapitlet om appmarkedsdistribution [9.1](#).

Hvordan? - Det skal gøres ved at lave et program, der gør overstående jf. "hvad?", via en moderne applikationsbyggeløsning, således at denne finde den gyldne vej imellem optimisering og kompatabilitet.

6 Produktprincip

6.1 Målgruppe

Det tilsigtes, at produktet skal være tilgængeligt for alle danskere, da kriser ikke diskriminerer. Det betyder i praksis, at personer, der har diverse handicap, skal være i stand til at kunne produktet, herunder folk, som er ord- eller farveblinde.

6.2 Kravspecifikation

Da det må forventes, at produktet skal kunne anvendes i tilfælde af et nedbrud af internettet, herfor skal produktet kunne fungere uden internetforbindelse dvs. applikationen skal være offline-og da det er planen, at appen skal indeholde videoer, skal disse ligedan nedhentes. Desuden skal produktet være simpelt og intuitivt at forstå, herunder have ordforklaringer og selve brugergrænsefladen skal være let at navigere rund i og følge nuværende UI-standarder. Produktet skal ligedan være et kompromis mellem design og letvægtighed, herved forstås der, at appen ikke er resursekraævende, således at selv ældre hardware kan tilgå appen.

Hermed på listeform:

- Offlinefunktionalitet
- Letvægtighed
- Simpel brugergrænseflade samt ordforklaringer

6.3 Konkurrenter

Der er ikke umiddelbart nogen decideret konkurrent til produktet, især ikke på det danske marked, som henvendes til, men der et utal af videoer og guides om overlevelse i det vilde og diverse beregnere på internettet, men her adskiller det forslået produkt sig ved at være tilgængeligt uagtet internetforbindelse, mere omfattende og tilgængeligt på dansk.

7 Produktudformning

7.1 Overordnet

Koden er opbygget, således at den kan skrives som en pseudo-webapplikation, sidenhen anvendes en "bro", der gør det til en native applikation og kompatibel med de mest almindelige styresystemer, herunder IOS (Apples mobile styresystem) og Android².

7.2 Kodestack

7.2.1 Framework: React Native

React Native er et framework³, der muliggør udvikling af native⁴ applikationer til IOS og Android.

7.2.2 Sub-Framework: Expo

Expo er et framework, der er bygget omkring React Native, der muliggør at køre en pågældende React Native-applikation igennem deres egen platform, Expo Go, hvorfor er smart, fordi man ikke behøver at kompile koden, dvs. oversætte programmeringskoden til eksekverbar maskinekode, efter hver ændring, såfremt man vil teste den på en fysisk enhed.

7.2.3 Runtime: Node.js

Node.js er den runtime, der muliggør, at applikationen kan køre på en enhed fremfor i en browser, da programmeringssproget JavaScript oprindeligt var designet til at køre udelukkende i et browser-miljø. Dette er altså en nødvendighed for, at applikationen skal kunne køre på en fysisk enhed.

7.2.4 Database: MongoDB*

Ideen var oprindeligt at have en meget let applikation, der kunne nedhentes fra et styresystems nativ app-bibliotek, hvorefter denne ville prompte brugeren til at nedhente ekstrapakker, fx videoer og manualer, fra vores egen server, som skulle være administreret via MongoDB. MongoDB er et program, som tillader en at lave og administrere en NoSQL-database, i stedet for en SQL-database, da det er mere skalerbart og fleksibelt i den forstand, at man hurtigt kan tilføje nye data, fx brødtekst, og hente disse. Se gerne kapitlet om serveropsætning [8.1](#) for at lære mere om denne database.

7.2.5 Programmeringsprog: Typescript

Typescript er et programmeringssprog udviklet af Microsoft, som bygger på JavaScript. Typescript bruger stærke datatyper, hvilket vil sige, at datatypen angives per data. Typescript anvendes i denne applikation, fordi det er et kompilersprog, hvilket vil sige, at fejl kan blive fanget, når koden kompileres, fremfor i runtime, mens programmet kører, hvilket

²Selvsamme teknik anvendes af højprofilerede virksomheder, såsom Facebook, Discord og Tesla. Læs mere her: reactnative.dev

³Et »framework« er en samling af biblioteker, der gør det lettere at skrive kode på en standardiseret måde, hvori en masse valg er taget på forhånd

⁴»Native« betyder at applikationen kører direkte på den pågældende enhed

er en stor fordel i programmeringsprocessen, da det hindrer, at der kommer oversete fejl i koden.

7.3 Brugergrænseflade

Brugergrænsefladen er det, brugeren oplever, når han interagerer med applikationen. Brugergrænsefladen er blevet tegnet i Figma, et program, som er speciallavet til at konstruere brugergrænseflader, bl.a. har den prælavede elementer, fx knapper og tekstinputfelter, desuden kan dette gøres interaktivt.

7.3.1 Konceptdesign

Jf. appendiks B er følgende skitser blevet udarbejdet. Herefter er disse blevet implementeret i selve applikationen. Brugergrænseflade består primært af en bjælke, der er placeret i bunden af skærmen, hvorfra brugeren kan navigere mellem forskellige sider.

7.3.2 Design

De stilistiske valg der blev taget om appens udseende fokuserede på at den skulle fremstå neutral og rolig, et minimalistisk udtryk der benyttede sig af naturlige farver var den endelige konklusion da det bedst udtrykker appens stil. Til at komme frem til denne konklusion blev Design Thinking-modellen benyttet [3]

7.4 Produktgennemgang

Koden samt ideerne som applikationen bygger på, vil blive gennemgået nedenfor.

7.4.1 Filstruktur

Måden hvorpå filerne er struktureret, er altafgørende for kodens funktionalitet og læsbarhed. Da frameworkt React Native, kigger efter spicifikke filstrukturer, til at køre og vise den dertilhørende kode det korrekte sted.

```

/
├── app
├── assets
├── components
├── constants
├── data
├── hooks
├── node_modules
├── scripts
├── .gitignore
├── app.json
├── babel.config.js
├── eas.json
├── package-lock.json
├── package.json
└── tsconfig.json

```

Figur 4: Top-level filstrukturen for applikationen. Mapperne er farvet blåt

Måden, hvorpå brugeren navigerer applikationen, er ved at bruge en navigationsbar, som er placeret i bunden af skærmen. Filstrukturen som viser navigationsbaren, ser således ud i kodebasen:

```

/app
├── (tabs)
│   ├── afgrøde-beregner
│   ├── bibliotek
│   ├── e-beregner
│   ├── indstillinger
│   ├── layout.tsx
│   ├── index.tsx
│   ├── layout.tsx
│   └── +not-found.tsx

```

Figur 5: Top-level filstrukturen for routes-systemet. Mapperne er farvet blåt

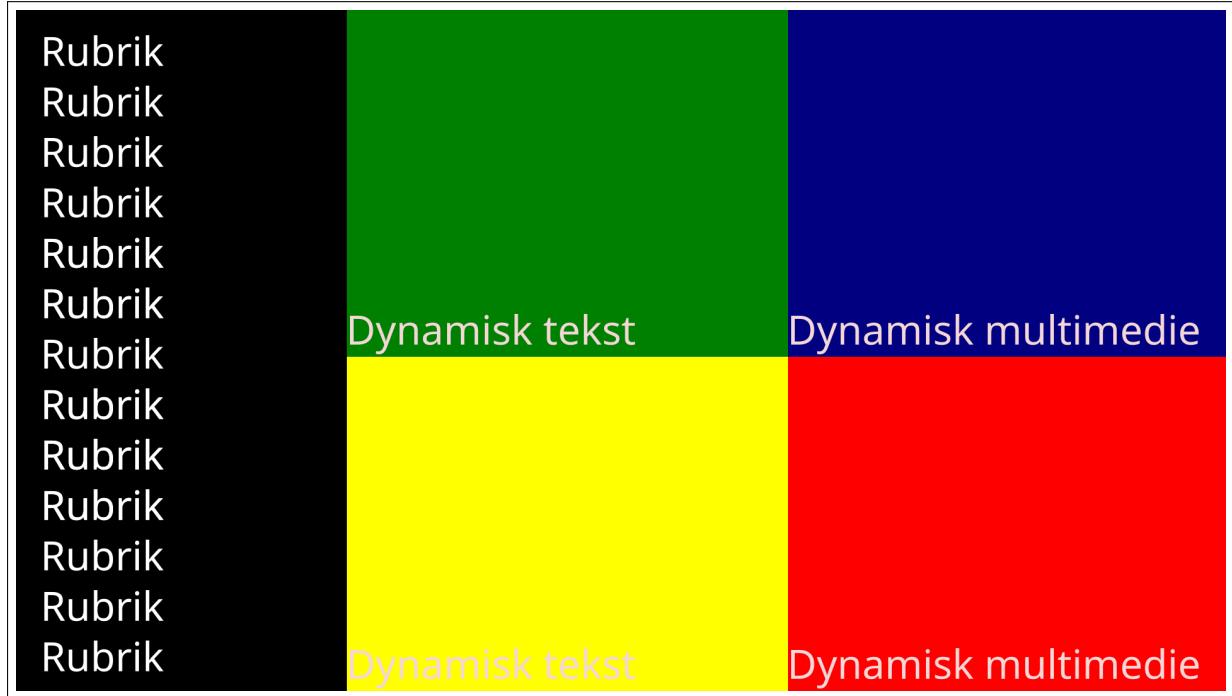
Det smarte ved netop denne filstruktur, er at det er komponentbaseret, hvilket vil sige, at man kan hente og genbruge komponenter fra mappen med disse, hvorfor koden bliver mere effektiv, da man ikke behøver genkode tidligere kodede elementer.

7.4.2 Bibliotekskoden

Her benyttes dynamic-routes⁵ til at hente brødtekst og rubrikker fra en database. Dette muliggør, at navigationsbjælken til højre kan være renderet konstant, mens den til højre (se nedenfor illustration) kan være renderet dynamisk, alt efter hvilken route brugeren

⁵ »Dynamic routes« er en funktion i React-frameworket [7.2.1](#), som gør man kan lave routes ud fra dynamisk data. Man kan forestille sig fx en generisk side, der populeres med data hentet eksternt, fx dens rubrik, underrubrik, url, etc.

befinder sig på. Læg mærke til, at det er opdelt via en flexboks⁶, hvilket gør, at der er fire kasser, de to yderste til venstre indeholder kun tekst, hvor de to til højre kan indeholde tekst, billeder og videoer, som hver især optager 50% af den resterende (navigationsbøjlen optager også plads) skærmplass.



Figur 6: Viser illustration af routes-systemet

I det her tilfælde, laves datakald til en JSON-fil⁷. Denne svarer i samme format, som en database ville, hvorfor dette er anvendt til at demonstrere datahåndteringsfunktionalitet uden at en reel database er nødvendig. I kapitlet 8 kan der læses mere om en eventuel implementering af en database.

7.4.3 E-beregner

E-beregneren er en simpel beregner, der har til formål at informere brugeren om, hvor stort deres kalorieindtag skal være ud fra nogle data. Denne tager højde for BMR, basalmetabolisme, som udgør ca 70% af totalenergiforbruget og PA, fysisk aktivitet, som udgør ca. 20% men ikke DIT, diætinduceret varmedannelse, som udgør 10%[4].

Matematikken Til formålet anvendes basalmetabolisme, som beskriver et varmblodet dyrs stofskifte.[5] Her tages udgangspunkt i menneskets basalmetabolisme. Til dette formål anvendes Harris-Benedict-formlen, som er blevet revideret af Mifflin og St. Jeor i 1990[6], der ser således ud:

For mænd:

$$\text{BMR} \left[\frac{\text{kcal}}{\text{dag}} \right] = (10 \cdot \text{vægt} [\text{kg}]) + (6,25 \cdot \text{højde} [\text{cm}]) - (5 \cdot \text{alder} [\text{år}]) + 5$$

⁶»Flexbox« er en form for gitter, der skaleres dynamisk ift. de omkringliggende elementer.

⁷JSON (JavaScript Object Notation) filer er et letvægts- læsbart filformat, som bruges til at organisere strukturere information.

For kvinder:

$$\text{BMR} \left[\frac{\text{kcal}}{\text{dag}} \right] = (10 \cdot \text{vægt [kg]}) + (6,25 \cdot \text{højde [cm]}) - (5 \cdot \text{alder [år]}) - 161$$

Forskellen i formlen, skyldes mænd har en lidt højere basalmetabolisme end kvinder.

For at estimere det resterende kalorieforbrug, multipliceres BMR med en konstant, som afhænger af en persons fysiske aktivitet. Tallene er fundet fra en beregner på nettet med lignende funktionalitet [7]:

Aktivitet	Konstant
Sedentær	1,2
Lidt eller ingen aktivitet	1,4
Let aktivitet 1-2 gange om ugen	1,6
Moderat Aktivitet 2-3 gange om ugen	1,75
Hård aktivitet 3-5 gange om ugen	2
Fysisk job eller hård daglig aktivitet	2,4

ergo:

$$\text{Beregnet kalorieindtag} = \text{BMR} \cdot \text{aktivitetskonstant}$$

Implementering Forneden er kodeimplementeringen af formlen:

```
let bmr = 0;
if (currentEntry.koen.toLowerCase() === "mand") {
    // Harris-Benedict equation Men
    bmr = (10 * parseFloat(currentEntry.vaegt))
        + (6.25 * parseFloat(currentEntry.hoejde))
        - (5 * parseFloat(currentEntry.alder)) + 5;
} else if (currentEntry.koen.toLowerCase() === "kvinde") {
    // Harris-Benedict equation women
    bmr = (10 * parseFloat(currentEntry.vaegt))
        + (6.25 * parseFloat(currentEntry.hoejde))
        - (5 * parseFloat(currentEntry.alder)) - 161;
}
const maintenanceCalories = Math.ceil((bmr *
    parseFloat(currentEntry.aktivitetsniveau)));
const g_oksekoed = maintenanceCalories / 2.5;
```

Kilde: app/(tabs)/e-beregner/e-beregner.tsx, linje 42-51

Brugeren indtaster diverse værdier, herunder brugerens alder, vægt, højde, køn, aktivitesniveau og navn, i inputfelter, hvorefter disse kan tilgås i backenden⁸. Først tjekker koden, om hvorvidt brugeren er en mand eller kvinde, hvorefter den anvender enten den mandlige eller kvindelige Harris-Benedict-formel. Alle ledene i formlen er ens, på nær det sidste led, som enten tillægger 5 kalorier for mænd og trækker 161 kalorier for kvinder. Selve køn-variablet omdannes til småt, for at sikre, at der ikke kommer nogle fejl i koden, også selvom brugeren kun kan vælge mellem to prædefinerede værdier, altså en form for intern fejlsikring.

⁸»the part of a computer system or application that is not directly accessed by the user, typically responsible for storing and manipulating data.« - Oxford Languages

Ellers tages vægten, som multipliceres med 10, højden med 6.25 og 5 for alderen. Disse værdier er kun de nuværende indtastede værdier, idet e-beregneren har multipersonfunktionalitet, hvilket vil sige, at værdierne låses, når selve beregningen er gennemført, hvorefter nye værdier kan indtastes, og derved oprettes en ny profil (beregning). Desuden eksemplificerer programmet det daglige kaloriebehovet i diverse hverdagsgoder, i det her tilfælde, gram oksekød. Dette gøres ved at dividere det daglige kaloriebehov med 2,5 kalorier pr. gram oksekød, som er det gennemsnitlige energiindhold i oksekød.

Dette er en simplifikation af den oprindelige vision for programmet, som skulle have kommet med forslag til retter og måltider, alt efter brugerens behov. Læs mere om det i kapitlet [11](#)

7.4.4 Afgrødeberegner

Afgrødeberegneren er en simpel beregner, der beregner, hvor meget såsæd og areal, der skal til for at opnå en angiven mængde meludbytte. Beregningerne og slutresultatet er blevet gennemgået af Hortonom Morten Jensen.

Matematikken Arealfunktion:

$$\text{meludbytte [input]} = \text{Såareal [output]} \cdot \frac{\text{kornudbytte}}{\text{Såareal}} \cdot \frac{\text{meludbytte}}{\text{kornudbytte}}$$

$$\Leftrightarrow \text{Såareal [output]} = \frac{\text{meludbytte [input]}}{\left(\frac{\text{kornudbytte} \cdot \text{meludbytte}}{\text{Såareal} \cdot \text{kornudbytte}} \right)}$$

Sædfunktion:

$$\text{meludbytte [input]} = \frac{\text{meludbytte}}{\text{kornudbytte}} \cdot \frac{\text{kornudbytte}}{\text{sæd [output]}}$$

$$\Leftrightarrow \text{meludbytte [input]} \cdot \text{sæd [output]} = \frac{\text{meludbytte}}{\text{kornudbytte}} \cdot \text{kornudbytte}$$

$$\Leftrightarrow \text{sæd [output]} = \frac{\text{meludbytte}}{\text{kornudbytte}} \cdot \frac{\text{kornudbytte}}{\text{meludbytte [input]}}$$

Implementering heraf Forneden er kodeimplementeringen af formlerne:

```
const froe_pr_m2 = 300;
const korn_pr_m2 = 0.5;
const mel_udbytte = 0.75;

const onsket_mel = parseFloat(currentEntry.mel);
const nodvendigt_korn = onsket_mel / mel_udbytte;
const nodvendigt_areal = nodvendigt_korn / korn_pr_m2;

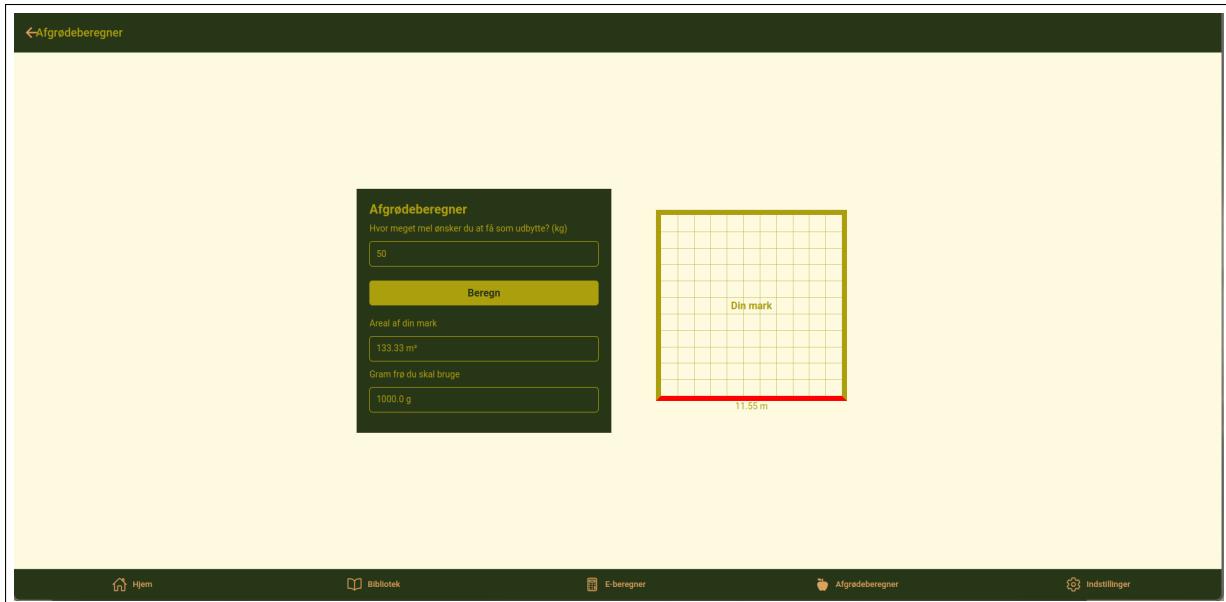
[...]
const antal_froe = (nodvendigt_areal * froe_pr_m2) / 40;
```

Kilde: [app/\(tabs\)/afgrøde-beregner/afgrøde-beregner.tsx](app/(tabs)/afgrøde-beregner/afgrøde-beregner.tsx), linje 36-47

Brugeren indtaster meludbyttet i et inputfelt og trykker beregn, hvorefter en funktion af meludbytteinputet (onsket_mel-konstanten) køres, som beregner den nødvendige mængde hvedesæd og arealallokeringen hertil for at opnå den ønskede mængde malet korn.

Udover at rendere resultatet heraf, så illustreres arealet ved hjælp af en kvadrat, hvis siderlængder er tilsvarende kvadratrodens af det givne areal. Gitterlinjernes kvadrater svarer cirka til en kvadratmeter, hvorfor brugeren kan forstå, hvor stort arealet er. Dette er en vigtighed, da markers størrelse hurtigt kan bliver uoverskelige, hvis de ikke holdes i perspektiv.

Fornede ses den visualisering af arealet, som koden laver:



Figur 7: Viser visualiseringen af arealallokeringen, programmet laver.

7.4.5 Indstillinger

Forklaring Indstillinger-siden er lavet, således at produkteretkriteriet omkring tilgængelighed opfyldes. Det gøres ved, at der er en to switches⁹:

1. En dysleksi-switch, der styrer om brugeren vil have en dysleksi-venlig skrifttype. Dysleksi-fonten, »Open Dyslexic«, er udviklet med videnskabelige undersøgelser om læsbarheden af tekst med henblik på at forbedre læsbarheden for dyslektikere, i minde[8]. Herfor er den selvsagt valgt.
2. en farveblindheds-switch, der styrer om alle elementer skal renderes farverigt eller sort-hvidt. Sort-hvid-paletten er valgt, da alle farveblinde vil kunne afkode elementer med disse nuancer, selv folk, der lider af monokromatisk farveblindhed[9].

Implementering heraf

Dysleksi-switch Forneden ses koden, der modtager input fra dysleksi-switchen og anvender dette til at indstille skrifttypen. Denne import sker i præamblen til en hvilkensomhelst fil, der indeholder tekst, som skal vises til brugeren:

```
import { useFont } from "@/components/fontContext";
const { dyslexiaMode } = useFont();
```

Kilde: Præamblen¹⁰ til en hvilkensomhelst fil, der indeholder tekst.

Her ses teksten, hvori de importede »themes« (temaer) er anvendt i dennes style-værdi. Dette går igen igenem hele koden:

⁹»Switch« er en visualisering af en sandhedsværdi, som enten er sand eller falsk, altså et binært 1 eller 0, en fysisk manifestation heraf kunne være en vippekontakt.

¹⁰»Præamblen« er det første kode, der køres. Heri kan metainformation angives om koden, fx titel, forfatter, dato, version, etc., men også kodningsrelevante elementer såsom importeringen af andre filer, konstanter, elementer, biblioteker, etc, som anvendes generelt eller senere i koden. Således indstilles "kodemiljøet"heri.

```
{ fontFamily: dyslexiaMode ? 'open-dyslexic' : 'System' }
```

Kilde: hvilkensomhelst teksts »style«-værdi.

Farveblindheds-switch Forneden ses koden, der modtager input fra farveblindheds-switchen og anvender dette til at ændre farvepaletten:

```
import { useTheme } from "@/components/themeContext";
import { normalTheme, colorBlindTheme } from "@/constants/themes";

const currentTheme = theme === 'normal' ? colorBlindTheme: normalTheme;
const { theme } = useTheme();
{ backgroundColor: currentTheme.calculatorBackground }
```

Kilde: en hvilkensomhelst fil, der anvender en form for farvepallete, altså vises til brugeren.

Forneden ses koden, hvor farvepalleteerne, som eksporteres er, afhængigt af, hvad værdi, switchen har. Læg mærke til, at de forskellige farvepaletter, normalTheme og colorBlindTheme som er defineret i to rækker¹¹:

```
export const normalTheme = {
  background: "#222b00",
  headerColor: "#283618",
  arrowIconColor: "#ddaa15e",
  fontColor: "#ac9f0d",
  tabBarActiveTintColor: "#ddaa15e",
  tabBarInactiveTintColor: "#ddaa15e",
  bibLinkStyle: "#DDA15E",
  bibBackground: "#283618",
  grejLineStyle: "#000",
  grejIconColor: "#ddaa15e",
  calculatorBackground: "#fefae0",
  redBlack: "black",
  blackRed: "red",
  calculatorButton: "#000",
  newBibBackground: "#fefae0",
  grejViAnbefaler: "#ddaa15e",
  afgrødeBeregnerButton: "#0000FF",
}

export const colorBlindTheme = {
  background: "#fff",
  headerColor: "#fff",
  arrowIconColor: "#000",
  fontColor: "#000",
  tabBarActiveTintColor: "#000",
  tabBarInactiveTintColor: "#000",
  bibLinkStyle: "#000",
  bibBackground: "#fff",
```

¹¹»Række« er en række af elementer, som har et navn og en værdi, her et navn og en farve.

```
grejLinkStyle: "#000",
grejIconColor: "#fff",
calculatorBackground: "#fff",
redBlack: "red",
blackRed: "black",
calculatorButton: "#0000FF",
newBibBackground: "#fff",
grejViAnbefaler: "#000",
afgrødeBeregnerButton: "#000",
}
```

Kilde: `constants/themes.ts`, linje 1-39

Andre forslag til "indstillinger-siden" findes i post scriptummet omkring visionen for produktet [11](#).

8 Produktionsforberedelse

8.1 Serveropsætning

Serverhosting Da der heri projektet er tale om en applikation, der skal kunne hente data, fx tekst, videoer og billeder, fra en ekstern server, skal sådan en selvsagt bruges. Hertil er det meningen, at KEP skal hoste sin egen server, som så ville kunne håndtere disse data og udsende dem til forbruger.

MongoDB Hertil ville MongoDB være anvendt til at administrere dataene. Læs mere om dette i kapitlet om Database [7.2.4](#).

8.2 Betatesting || åben Quality Assurance

For at sikre kvaliteten af produktet, vil programmet bliver betatestet¹². Perioden, hvori programmet betatestes, vil det være frit tilgængeligt for alle med det forbehold, at de ikke er kunder, hvorfor KEP heller ikke kan holdes ansvarlig for eventuelle mangler eller fejl. Læs mere om overvejelser af ansvar og etik i kapitlet om [9.2.1](#).

¹²»Betatesting« er en form for test, hvor brugereude i den virkelige verden får lov at afprøve produktet, så de kan give feedback, hvor eventuelle mangler kan rettes op på.

9 Produktrealisering

9.1 KEP-branding

KEP er en imaginær privat virksomhed, der står for at udvikle og distribuere applikationen samt hoste dataene, denne nedhenter. »KEP« er et akronym, der står for »Krisemanual- og ernærningsberegningsprogram«. Akronymet eksistens begrundes i, at det fulde navn er for langt til at være memorabelt og genkendeligt.

KEP har ligedan et logo og en farvepalette, (læs mere om farvepaletten i sektionen om design 7.3.2). Logoet er simpelt og består af akronymet KEP. Logoet er lavet med skriftypen Bebas Neue, hvorefter denne er blevet modifieret, så den minder mere om DRs gamle logo:



Figur 8: Viser logoet for DR (1964-1996), som er blevet brugt som inspiration til KEPs logo

Der tages udgangspunkt i DRs logo, da det er æstetisk samt signalerer stabilitet og troværdighed.

Selv logoet er blevet lavet i Inkscape¹³, via vektorgrafik, så det kan genanvendes på forskellige måder, uden at miste kvaliteten samt skaleres.

Hertil er trinvisforbedring anvendt til at skabe logoet. Forneden er nogle af iterationerne, som logoet har gennempet:



Figur 9: Viser første udkast til logoet.



Figur 10: Viser andet udkast til logoet.

Herefter det endelige logo:

¹³»Inkscape« er et gratis vektorgrafikprogram.



Figur 11: Viser det endelige logo.

9.2 Profiteringsmodel

For at gøre appen profitabel, vil denne koste et gebyr for at anvende. Grundet kriteriet om at appen skal være tilgængelig uden internetforbindelse (jf. kapitlet om produktprincip [6](#)), vil det være umuligt at gøre således, at appejerskab kontrolleres, såfremt der betales regulært, når denne startes. Herfor er det ikke muligt at gøre tjenesten abonnement-baseret.

Derfor er det valgt, at appen koster et engangsgebyr, hvorefter yderligere DLC-pakker^{[14](#)} vil kunne tilkøbes, hvilket vil give mulighed for løbende udvikling mhp. at krisesikre brugeren mod diverse hypotetiske situationer. I takt med, at det vurderes, at efterspørglen efter appen stiger, vil priserne for disse pakker og grundkøbet hæves løbende og ligedan sænkes, hvis efterspørgslen vurderes lavere.

For at sikre KEPs interesser jf. overstående, da er det essentielt, at koden ikke viderdeles af ikke-autoriserede personer eller organisationer. Således vil der i appens TOU^{[15](#)} blive beskrevet, hvordan brugeren ikke må videreføre appens kode, ellers vil de kunne retforfølges for erstatning som måtte følge heraf. Ligedan vil appens ideer og struktur blive patenteret, hvorfor andre virksomheder ikke vil kunne udvikle en applikation magen til. Desuden vil diverse brand- og logokomponenter blive varemærket, således at de ikke bliver associeret med andre virksomheder uden KEPs tilladelse, altså er brandopfattelsen indenfor KEPs kontroldsfærite, så det kan sikres, at KEP opfattes troværdigt.

9.2.1 Etik og ansvar

Da appen vil være dyr, må det forventes, at KEP er ansvarlig for at appen har korrekt og brugbart indhold, altså betaler man også for dette som bruger. Ligedan i perioden, hvor appen er gratis, vil KEP ikke være ansvarlig for eventuelle fejl eller mangler i appen. For at sikre indhold af højest mulig kvalitet, vil KEP samarbejde med eksperter og organisationer om at udarbejde dette.

¹⁴ »DLC« står for »Downloadable Content«.

¹⁵ »TOU« står for »Terms of Use« og beskriver hvilke vilkår og regler, en bruger skal overholde for at kunne benytte sig af appen

9.3 Appmarkedsdistribution

De to primært anvendt styrestystemer på tablets er iOS¹⁶ og Android¹⁷. I Danmark, er har IOS en markedsandel på 65,95%, hvor Android har en markedsandel på 33,62%^[11]. Herfor giver det også god mening, at appen er udviklet til at køre på begge dele.

9.3.1 Apple App Store

På IOS, er den mest almindelige package manager¹⁸ App Store. Dette har længe været den eneste måde at nedhente applikationer på, da det er Apples egen platform, dog på grund af et EU's Digital Markets Act, skal Apple gøre således, at brugere i EU kan tilgå andre pakke-forvaltere.^[12] Således vil KEP også gøre, således at brugere i EU, kan downloade appen direkte fra KEP, uden mellemmand, dog vil den stadig være tilgængelig på App Store og andre pakke-forvaltere på iOS, hvis disse eller bliver udbredt.

9.3.2 Google Play Store

På Android, er den mest almindelige package manager Google Play Store. Dog findes der også andre pakke-forvaltere, som f.eks. F-Droid¹⁹. F-droid har kun applikationer, som er open source²⁰ og gratis, hvilket ikke harmonerer med KEPs profiteringsmodel og kodenstruktur ??, som kommer til at være proprietær, hvorfor denne vil ikke være tilgængelig her.

9.3.3 Egendistribution - apk

9.4 Viderdownload-prompt

¹⁶iOS er Apples styresystem til deres mobile enheder.

¹⁷Android er et linuxbaseret styresystem, som udvikles af Open Handset Alliance, der består af 84 forskellige virksomheder, hvor Google har hovedtetten.^[10]

¹⁸»package manager« er en software, der adminisrerer softwarepakker, således at eventuelle softwareafhængigheder installeres. Desuden opdaterer den pakker og kan slette nuværende.

¹⁹»F-Droid« er ligesom Google Play Store, en pakke-forvalter til Android, som er open source og gratis.^[13]

²⁰Indenfor programvare, differentierer man imellem proprietær kode, som kun kan tilgås af ejeren || udvikleren, og open source kode, som kan tilgås og bidrages til af alle.

10 Selvevaluering (meta)

11 Post scriptum om visionen for produktet

Litteraturliste

1. Jeppesen, M. M., Henriksen, L. B. & Routhe, H. W. *Projektarbejdet - Teknologi og teknikfag* Systime. <https://projektarbejdet.systime.dk/> (2024).
2. *Forberedt på krise* Beredskabstyrelsen. https://www.brs.dk/globalassets/brs---beredskabsstyrelsen/dokumenter/forberedt/forberedt_paa_kriser_lowres-.pdf (2024).
3. Dahl, L. E. S., Andersen, L. D. & Pedersen, L. A. *Systime Design Grundbog* Systime. <https://designgrundbogen.systime.dk/?id=404> (2024).
4. *Energy expenditure* Wikipedia. https://en.wikipedia.org/wiki/Energy_expenditure (2024).
5. *Basal metabolic rate* Wikipedia. https://en.wikipedia.org/wiki/Basal_metabolic_rate (2024).
6. *Harris-Benedict equation* Wikipedia. https://en.wikipedia.org/wiki/Harris-Benedict_equation (2024).
7. MD, A. Z. & PhD, J. M. *Maintenance Calorie Calculator* <https://www.omnicalculator.com/> (2024).
8. Gonzalez, A. *Open Dyslexic, Related Research* OpenDyslexic. <https://opendyslexic.org/related-research> (2024).
9. *Types of Colour Blindness* Colour Blind Awareness. <https://www.colourblindawareness.org/colour-blindness/types-of-colour-blindness/> (2024).
10. *Android* Wikipedia. https://en.wikipedia.org/wiki/Open_Handset_Alliance (2024).
11. *Mobile Operation System Market Share Denmark* StatCounter. <https://gs.statcounter.com/os-market-share/mobile/denmark> (2024).
12. *Apple Digital Markets Act* Apple. <https://developer.apple.com/support/dma-and-apps-in-the-eu/> (2024).
13. *F-Droid* Wikipedia. <https://en.wikipedia.org/wiki/F-Droid> (2024).

Appendiks

A Projektbeskrivelse

Projektbeskrivelse af projekt 1 ”Krisen kradser” i faget Teknologi B

Alexander Knudsen, Andreas Jensen og Jeppe Bech

29. oktober 2024

1 Formalia

1.1 Gruppесammensætning

Projektet, som heri beskrives, er udarbejdet af følgende personer: Alexander Knudsen, Andreas Jensen og Jeppe Bech (2. X).

1.2 Tema

Overordnet: Krisen kradser

A
Underemne: Fødevarer
4

2 Produktbeskrivelse

Produktet, kulminationen på projektet, er en offline softwareprogram med diverse delprogrammer til krisehåndtering, herunder en ernæringsberegner, der også indeholder modstykke, der foretager anbefalinger om afgrødevalg og allokering af areal til disse.

Endvidere indeholder programmet også manualer med information til afljælpning af diverse nødsituationer samt praktisk information vedrørende daglig ageren, fx istandsættelse og anvendelse af radio, reparation af bil, isolering af hus, et cetera.

3 Tidsplan

Tidsplanen og tidsstyringen foregår via GitHubs indbyggede funktionlaitet, dog tilstræbes det som udgangspunkt, at projektet foregår over fem faser, dog er dette fleksibel og kan korrigeres undervejs:

1. Dataindsamling og skitseudarbejdelse (Projektesgrundlag) - varighed (1 uge)
2. Brugeroverflade samt udarbejdelse af hardwarepersona (Produktudvikling) - varighed (3 dage)
3. Ernærings- og afgrødeberegner (Produktudvikling) - varighed (1 uge og 4 dage)

4. Manualuddrag samt bogtilvalg til henvisning, finpudsning og evt. *visualiser* - varighed (1 uge)
5. Evt. distribution og afrappotering - varighed (1 uge)

B Konzeptart

Samling af konceptskitser

Alexander Knudsen, Andreas Jensen og Jeppe Bech

5. december 2024

1 Navigationsbjælkekoncept



Figur 1: Viser Navigationsbjælkekonceptet

2 Hovedmenuen



Figur 2: Viser frontpagekonceptet

3 Manualbibliotekskoncept



Figur 3: Viser manualbibliotekkonceptet

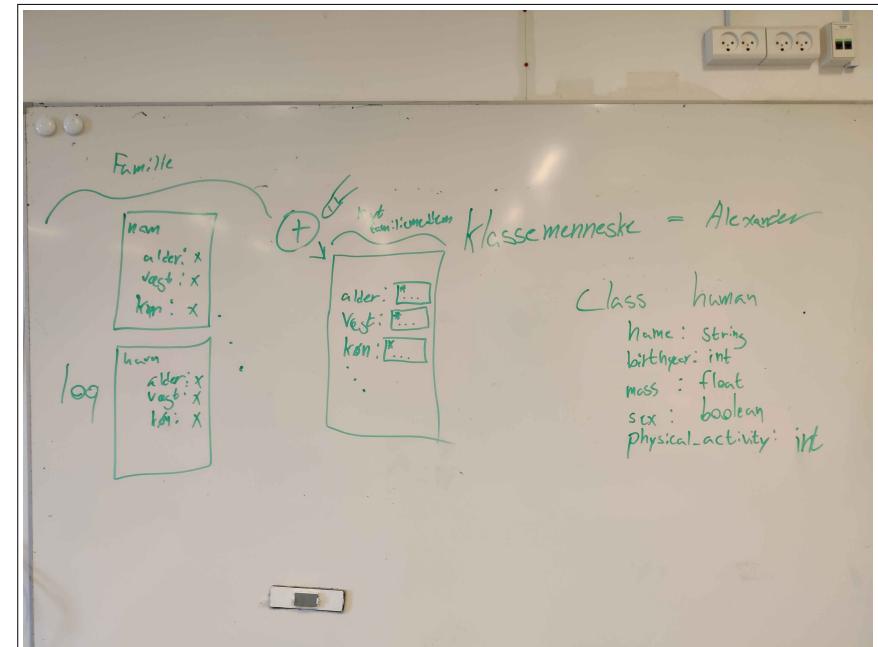
4 Grejlistekoncept



Figur 4: Viser grejlistekonceptet

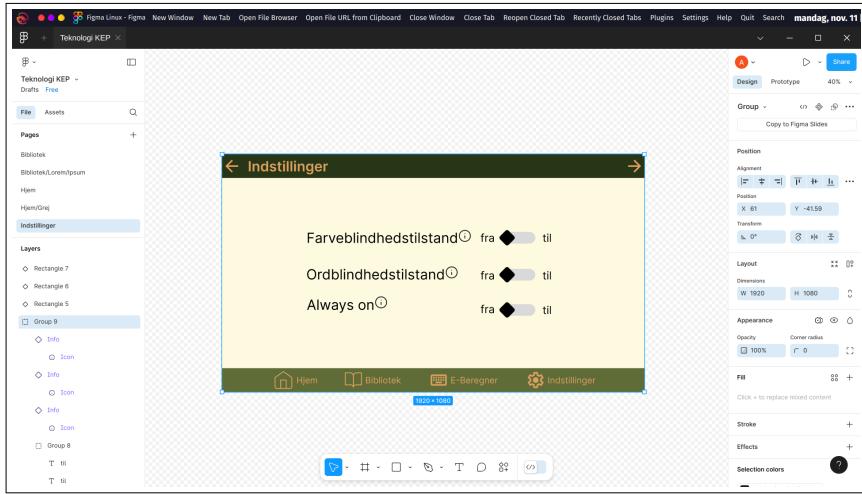
B27

5 Ernæringsberegnerkoncept



Figur 5: Viser ernæringsberegnerkonceptet

6 Tilgængelighedsmenukoncept



Figur 6: Viser tilgængelighedsmenukonceptet