# Prosjektbeskrivelse og -plan INF219 - DietPlan

Prosjektet tar sikte på å utvikle en app, for registrering av kaloriinnhold i måltider, mulighet for å sette opp en ukeplan med måltider, kunne se hvor stor andel dette utgjør av daglig anbefalt inntak i forhold til brukerens vekt, kjønn og alder, og lagre dette som en PDF. Appen skal fungere i Android, med konvertering til iOS om tiden i prosjektet tillater det.

For å utvikle appen, så kodes den i HTML/JavaScript, med data lagret i JSON-format i egen fil, som er raskt i bruk, og enkelt å oppdatere.

**BESKRIVELSE AV REACT NATIVE OG PROSESSEN SKRIVES INN HER.**

[www.matvaretabellen.no](http://www.matvaretabellen.no/) har en relativt omfattende liste over matvarer med næringsinnhold, og denne kan benyttes fritt med kildehenvisning. Tabellen er i excel-format (XLSX), og vi skriver et eget Python - Script som konverterer til JSON-format for enklere bruk direkte i JavaScript.

Matvaretabellen synes ved gjennomsyn å inneholde litt for mange delvis gjentakende matvarer, og vi må få ned mengden registreringer, spesielt med hensyn til brukervennlighet av appen. Dette er en manuell jobb, som vil ta noe tid, men mengden er overkommelig, og er pr. nå på ca 1000 registrerte matvarer (fra ca 1800 initielt).

Matvarene er tenkt kategorisert i en grovkategori og en underkategori, slik at brukeren gjør (maksimalt) tre trykk for å velge en matvare. Dette ivaretar en grei brukervennlighet, samtidig som matvarene deles opp på en måte som gjør det enkelt for brukeren å finne korrekt matvare kjapt.

Hovedkategorier kan være; kjøtt, fisk, brødmat, frukt, grønnsaker, søtsaker, drikke, meieriprodukter. Dette gir totalt 8 hovedkategorier, og med tilsvarende underkategorier, og med f.eks. 12 matvarer i hver underkatagori, så gir det plass til totalt 768 matvarer, som er overkommelig i forhold til listen som nå er på overkant av 1000. Kan også bruke denne kategorimodellen for å gjøre et tilpasset utvalg i den eksisterende listen. I og med at fordelingen av matvarer på kategorier ikke er helt jevn, så beregningen over er kun er snittestimat, som bare er en indikator mer enn en endelig mengde.

Videre så kan brukeren legge inn kjønn, alder, aktivitetsnivå og vekt, og ut i fra dette så vil også appen gi veiledning om hvordan hvert måltid og hver oppsatt dag, forholder seg til anbefalt kaloriinntak i forhold, i henhold til <http://kurs.helsekompetanse.no/pws/10407> :

For å finne hvilemetabolismen brukes følgende formel:

**Kvinner:** BMR = 655.1 + ( 9.563 x vekt i kg ) + ( 1.850 x høyde i cm ) – ( 4.676 x alder i år )  
**Menn:** BMR = 66.5 + ( 13.75 x vekt i kg ) + ( 5.003 x høyde i cm ) – ( 6.755 x alder i år )

*Så en mann på 34 med vekt 75 og høyde 175, vil ha en hvileforbrenning på;   
66.5 + (13.75 x 75)+(5.003 x 175) - (6.775 x 34) = 2346 kcal*

***Samt også valg for aktivitetsnivå:***

Lite eller ingen trening: BMR x **1.2**Lett trening (1-3 dager i uka): BMR x **1,375**Moderat trening (3-5 dager i uka): BMR x **1,55**Hard trening (6-7 dager i uka): BMR x **1,725**Veldig hard trening (to ganger om dagen, hard styrketrening): BMR x **1,9**

*og fra eksempelet over så vil en mann på 34, med vekt 75 kg og høyde på 175, med aktivitetsnivå “lett trening” ha et energibehov på 2346 x 1.375 = 3226*

Har nå skrevet egne funksjoner for disse beregningene, med input høyde, alder (fødselsår), vekt og aktivitetsnivå, som returnerer beregnet energibehov i kcal.

Disse vil importeres inn i appen ved behov, og trengs mer eller mindre kun når appen når den starter, eller hvis det er endringer i noen dataene brukt i beregningene. Utifra å unngå endringer, så bør input på alder endres til fødselsår, slik at appen beregner alder selv, og trenger ikke endre oppsettet hvis appen brukes over lengre tid.

Oppsett av datafil for matvarer med næringsinnhold er nå klar i første versjon.   
Basis for datafilen er nedlastet fra www...... og lastet inn i excel, og lagret som cvs, og deretter konvertert til JSON for rask og enkel bruk direkte JavaScript, som er grunnlaget for DietPlan-appen. Konverteringsmåten er rask og enkel, og gir god mulighet til endringer og å legge til matvarer som nå ikke er med, men som viser seg etterhvert som appen kommer i bruk.

Utviklerkonto hos Google er nå klargjort og appen kan legges ut for åpen nedlasting så snart den har tilstrekkelig med funksjonalitet og utenb vesentlige feil i koden (betaversjon). Etterhvert så kan den legges ut som ferdig versjon.

Funksjonaliteten er tenkt en tanke utvidet i forhold til basisversjonen. I helsesesammenheng er det gitt visse anbefalinger i forhold til måltider, med tanke på balansen mellom proteiner, fett og karbohydrater. For å kunne lage funksjonalitet som tar sikte på å rådgi når en bruker definerer/legger inn et måltid som avviker fra dette, så er også verdiene for protein, fett og karbohydrat tatt inn i databasen for samtlige matvarer. Bruk av disse verdiene vil ikke komme i tidligere versjoner av appen, men er tiltenkt funksjoner utover basisfunksjonalitet og vil legges til når basisfunksjoner (innhenting av databasedata og enkel måltidsberegning og lagring av disse) er på plass. Anbefalt fordeling er: 20-25 % protein, 40-50 % karbohydrat og 25-35 % fett

Databasen er nå etablert med 604 matvarer med kategorier og underkategorier, kcal, fett, karbohydrater og proteiner. Egen testrutine i Javascript/HTML for å hente inn og bruke data i utskrift, slik at vi kan hente kode inn i REACT og bruke i appen sånn som det passer annen funksjonalitet. Neste steg vil være å utforme en enkel versjon av appen, med noen få brukervalg, kanskje med bruk av formelen for energibehov, slik at den kan vise data og beregne andel av dagsbehov ved valg av matvare(r).   
Ytterligere funksjonalitet hvis tiden tillater det.

Har sett på iCAL-formatet, og muligheten for å produsere .ics – filer fra appen, slik at måltider definert i appen, kan enkelt legges inn i brukerens kalender. I android så vil systemet forespørre brukeren om en avtale skal legges inn i kalender, dersom den mottar en .ics-fil.

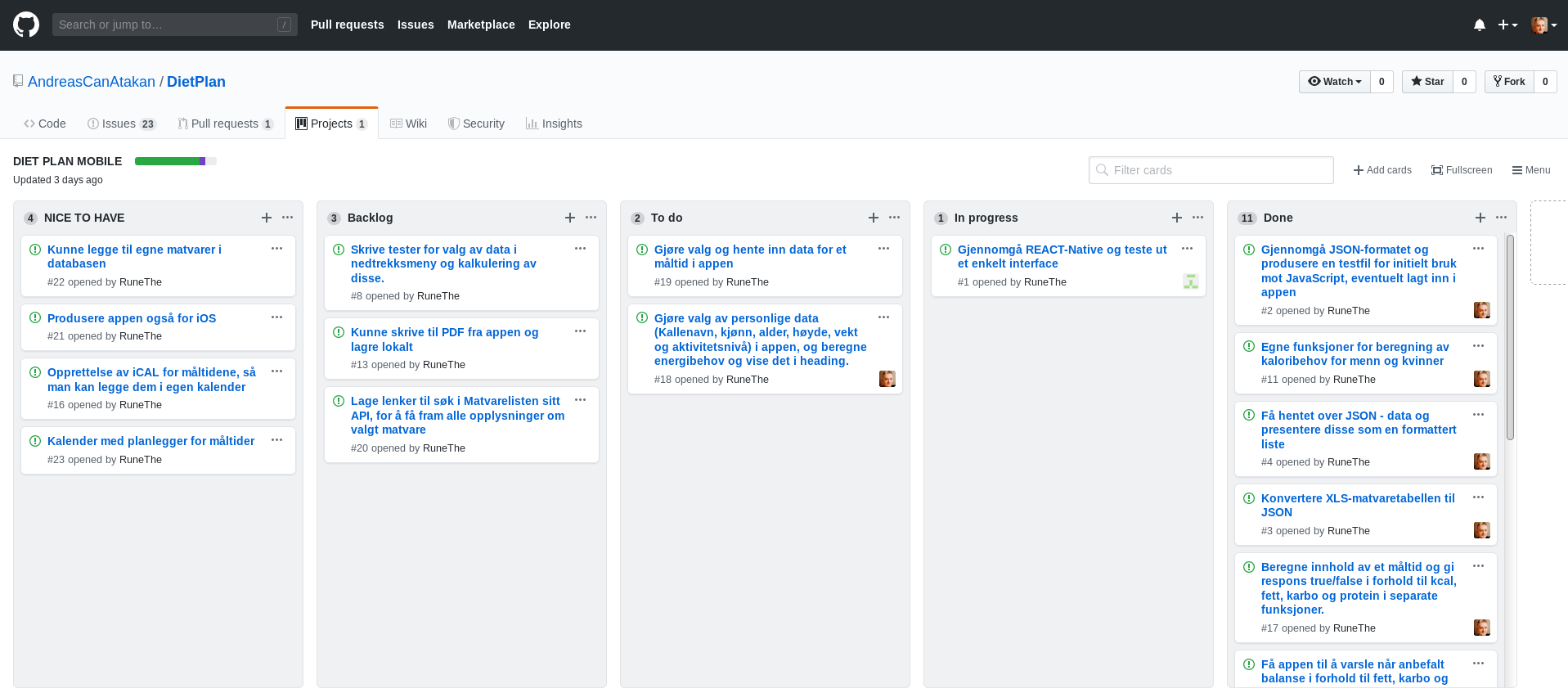
Fant et open source – prosjekt som har en mulig løsning for dette:

<https://github.com/nwcell/ics.js/blob/master/README.md>

Funksjonaliteten er uansett ikke prioritert før appen er langt på vei ferdig hva gjelder hovedfunksonnalitet.

Har laget mindre tester for å sjekke/validere at ingen av verdiene i JSON-databasen ikke har evvikk innefor forventede grenser. Har laget en mindre test som tar en array med tallverdier, og henter matvarene for de respektive tallverdiene fra JSON-databasen og skriver til skjerm, navn, og alle verdiene for matvaren, summerer verdiene og skriver dem til skjerm til slutt. Kun for testing, så ingen spesiell formattering av verdiene.

Har oppdatert project – board med en god del nye oppgaver, flyttet gjennomførte oppgaver, lagt til en del nice to have.



Videreutviklet testrutiner med funksjoner vi kan bruke senere i utvilkingsprosessen, med egne funksjoner som tester balanse i næringsinnhold for de tre viktige fett, karbohydrater og proteiner. Sjekket funksjonene opp mot virkelige data ved utplukk av 7 matvarer fra listen, summering av verdiene for disse, utskrift, og testing av de sammenlagte verdiene opp mot totalen for «måltidet», med respons fra de ulike funksjonene om innholdet er innenfor grensene som er anbefalt. Funksjonene kan enkelt tilpasses å returnere de verdiene man ønsker for videre prosessering eller håndtering i form av visning på brukerens skjerm ved det opprettede måltidet.

Kan vurdere funksjoner som kan lage en begrenset liste over hva man bør legge til for å få måltidet i balanse. Går også an å vurdere om vi kan gi bruker mulighet til å endre på grenseverdiene, f.eks. dersom man ønsker kost med mindre karbohydrater, kanskje med noen enkle skyvebrytere, eller knapper med «mye protein» «mest mulig protein» «minst mulig karbohydrater» «minst mulig fett» osv., for å gjøre det enkelt for brukeren. Kan også vurdere å utelate matvarer som f.eks. Har over en viss prosentandel karbohydrater eller fett. Mange muligheter og mange ideer ut av en sånn tilnærming, men skal kun brukes hvis tiden tillater det, og hvis vi anser det for en viktig nok del av bruken av appen.

Lagt til funksjoner for å lage objekt for et måltid, for å legge til en matvare i et måltids-objekt, fjerne en marvare fra et måltidsobjekt, lage et matvareobjekt som kan legges til et måltid, og laget randomiserte verdier fra matvarelisten, for å kunne teste at funksjonene er konsistente innenfor hele dataområdet.

Modifisert en del på strukturen i testfilen. Lagt til diverse funksjoner. En egen funksjon (med underfunksjoner), kan nå teste et helt måltid, og gi respons på dette i form at et objekt med beregnede verdier, og balansen mellom fett, karbohydrater og proteiner er innenfor anbefalte grenser. Generalisert ytterligere en del av funksjonene, og grenser for næringsinnhold har disse grensene som input til underfunksjonene for å sjekke innhold av fett, karbohydrater og proteiner.

Dette gir mulighet for å legge opp til ulike brukerprofiler, hvor noen f.eks. ønsker å ha mindre karbohydrater, mer proteiner osv. Vi må vurdere om det er mest hensiktsmessig å sette opp noen standard profiler, som brukerne kan velge, for å forenkle dette for brukerne, eller om vi skal gi full tilgang for brukerne til å endre dette selv, f.eks ved slidere. Grensene for de ulike næringsstoffene må på et vis matche, slik at man får en økning av de andre hvis en av dem minskes, og motsatt hvis en av dem økes.

Med hensyn til brukervennlighet, og ryddigere oppsett i koden, så heller jeg mot å lage noen standard profiler, som f.eks. «mindre karbo» «lite karbo» « mer protein» «mye protein»

Har også lagt klar funksjoner for å legge et måltid inn på en dag på spesifikk plass (opptil 7), legge inn en dag i et ukesobjekt (opptil 7), og tenker funksjon for å legge inn en dag på en spesifikk dato, også er nærliggende.

En del av funksjonaliteten er kun for testformål (som å få ut en randomisert liste av ID’er for matvarer), men i hovedsak alle andre funksjoner er tiltenkt å kunne benyttes i ferdig kode, og tar dem i bruk i testdok, for enklere verifisere funksjonalitet og korrekthet i forhold til datasettet som også vil være det samme i produsjons-kode.

***19.sept:*** Nye funksjoner for å produsere et user-objekt, med alle data; kjønn, alder, vekt, høyde, beregnet forbrenning pr. dag, og testet setting av to ulike objekt, gjenhenting av data til utskrift.

En funksjon beregner for både menn og kvinner, og endrer formel utifra parameter kjønn i input.

Har sett på filskriving via javascript, som er noe begrenset å gjøre direkte, da det ligger begrensninger med hensyn til sikkerhet. Må se på løsninger om REACT NATIVE har egne funksjoner som er enklere å legge inn. Har også sett på iCAL – formatet, og eventuelle libraries som kan benyttes til å gjøre dette til en god funksjon i appen. Finnes flere løsninger, men ingen peker seg ut umiddelbart. Må undersøkes nærmere.

***23.sept:***

Nye funksjoner for å evaluere en hel dag hmt energi, fett, karbohydrater og protein laget. Må testes inngående, da de gjør bruk av underfunksjoner, og må sikre at de produserer korrekte utdata, før de settes i produksjon. Egne funksjoner som evaluerer måltid, og egne funksjoner som evaluerer dag med måltider, som da gjør bruk av den første for å beregne summer av energu, fett, karbohydrat og protein, og av disse data beregne balansen i en hel dag. Ser også behov for å kunne evaluere et måltid separat mht balanse, og kunne angi dette for hvert av brukerens måltider også, i tillegg til en hel dag samlet. Har derfor egne funksjoner for måltid og for dag, hvor altså funksjonene for dag, gjør bruk av funksjonenen for måltid.

CheckDay – bruker – checkMeal – bruker checkFat/checkCarbo/checkProtein/checkKcal

CheckMeal – summerer alle verdier for alle parametre for et måltid

CheckDay – summerer alle verdier for alle summerte parametre fra checkMeal

CheckMeal tar et måltid som input, og gir et objekt med 4 parametre:

**sumFat, sumCarb, sumProt, sumKcal.**

CheckDay tar et objekt med liste over måltiders summerte verdier (sumFat, sumCarb, sumProt, sumKcal), og summerer disse til dayFat, dayCarb, dayProt, dayKcal, og kan av disse beregne andel av Fett, Karbo og Protein, og hvor stor andel av daglig anbefalt inntak måltidet utgjør for brukeren, i forhold til de innlagte data om kjønn, alder, vekt, høyde og aktivitetsnivå.

Tar sikte på å lage noen standardmåltider for å bruke til testing, slik at verdiene er enkle å følge, og som da gjør det enkelt å teste funksjonaliteten, og at alle utregninger er korrekte, siden det vil kunne være relativt enkelt å gjøre utregningene manuelt.

Etter dette så kan funksjonaliteten for testing utvides til tilfeldig genererte måltider, for å teste at funksjonaliteten ikke får uante konsekvenser når input varierer (innenfor normale verdier).

**Annen metodikk og planlegging:**

GIThub for deling av kode, branching, og testing.  
GIThub for projectboard - planlegging av enkeltoppgaver, framdrift i koding og REACT NATIVE - konvertering.

**Prosjektplan og framdrift:**

Innleveringsfrist: 1.desember 2019 - ca 13 uker fra 1.9.2019

**Målsettinger og gjennomført framdrift:**

|  |  |  |  |
| --- | --- | --- | --- |
| **UKE** | **Målsettinger** | **Gjennomført** | **Opphenting** |
| 0 (25.8.19) | Matvareliste - tynne ut  Python - konvertere fra XLS til JSON  Projectboard  Starte på beskrivelse  Starte på prosjektplan | Tynnet liste fra 1800 til ca 1000  Python - lest inn XLS  Projectboard satt opp  Enkel beskrivelse ferdig  Prosjektplan påbegynt | Tynne ned til 750  Python - lagre i array, og skrive til JSON i fil  Endret fokus til å lage JSON-fil direkte, for bedre kontroll og interaksjon med app, f.eks. lengde på navn og kategorier |
| 1 (1.9.19) | JSON - datafil (enkel for testing)  GIT repo fungerer med push/pull  Oppdateringer på arbeidsdokument  Designideer og diskusjoner rundt brukervennlighet.  [A] Oppdaterer “Nice to Have” og legger opp større tasks for krav som må gjennomføres. | JSON -datafil med 10 matvarer, 5 kategorier, og diverse underkategorier lagt opp i GIT  GIT funker, testet push, clone  Arbeidsdok i konstant oppdatering, milepæler satt,  forslag til neste ukes målsettinger. |  |
| 2 (8.9.19) | Gjennomgang av REACT NATIVE og kjøring av kode lokalt for testing.  Testing av produksjon av app, og teste enkel app på begge telefoner, Android10 og 9 ?  Teste ut ulike layouts for brukerinteraksjon?  Egne rutiner for beregning av BMR - en for menn og en for kvinner. | Database klar og testet med lokal kode, henter inn alle data til en HTML-side  Rutiner for BMR utarbeidet, slik at de kan hentes inn i appen ved behov.  Testfiler for datavalidering av JSON-data laget. Tester både verdier og kalkulering, i to separate filer.  Ytterligere testfil lagt til, nå med objektifisering av måltider, og funksjoner for å legge til, fjerne, og lage matvareobjekter. Tester også med randomiserte verdier. |  |
| 3 (15.9.19) | Fungerede enkel APP med matvaredata og beregning av BMR med valg for aktivitetsnivå | Testedok utvidet til å inkludere mange funksjoner som også vil benyttes i prod. Datasett utvidet og inntatt i testdok.  Kan bygge opp og teste et helt måltid for sum og balanse, legge måltid-objekt inn i en dag (liste over objekter), erstatte måltid i dag, hente ut måltid fra dag.  Tatt inn formel for BMR i testdok., generere tilfeldige verdier innenfor gitte rammer, for bruk for å teste formel.  Funksjon for å opprette og oppdatere et bruker-objekt med alle data, også BMR lagt inn. | Evaluere måltid i forhold til beregnet BMR. |
| 4 (22.9.19) |  | Startet med komplettering av funksjoner for å evaluere en hel dag med måltider, med underfunksjoner som gjør mellomregning for ett og ett måltid. |  |
| 5 (29.9.19) |  |  |  |
| 6 (6.10.19) | Fungerende enkel app med utvidede testdata |  |  |
| 7 (13.10.19) |  |  |  |
| 8 (20.10.19) |  |  |  |
| 9 (27.10.19) | Fungerende app med design, alle datafelt og ukesmeny |  |  |
| 10 (3.11.19) |  |  |  |
| 11 (10.11.19) |  |  |  |
| 12 (17.11.19) | Fungerende app med lagring og flerukers meny |  |  |
| 13 (20.11.19) | korreksjoner og rapport |  |  |
| 27.11.19 -> | korreksjoner og rapport |  |  |

**Eksempler på MATVARER - kategorier og underkategorier**

BRØDMAT

- brød - grov/fin

- rundstykker - grov/fin

- baguetter - grov/fin

- Søte - boller

- andre - pølsebrød/hamburgerbrød

GRØNNSAKER

* grønne paprika, agurk, salat, kål,
* røde - tomat, paprika, rosenkål, brokkoli,
* rotgrønnsaker - poteter, kålrot, gulrot, sellerirot, pepperrot,
* salat
* kål
* sopp sjampinjong, kantareller, aromasopp, østerssopp,
* bønner/linser
* brune avokado
* gule paprika, mais,
* oransje kålrabi,

FRUKT (farge?)

* sitrus - appelsin, mandarin, klementin, sitron, lime,
* bær - jordbær, bringebær, solbær, blåbær, molter,
* epler o.l. - eple, pære,
* gule - eple, grapefrukt, sitron, banan,
* røde druer, granateple, eple,
* grønne - lime, eple, druer
* oransje - appelsin, mandarin, klementin

Har laget en datafil JSON for testing i JavaScript

Vi vurderer å bygge denne opp etter behov, og for å beholde brukervennlighet, kontra å konvertere en massiv oversikt på 1800 matvarer, hvor mange korreksjoner må gjøres manuelt. Kan være tidsbesparende å tilpasse datafilen etter funksjonalitet og tiltenkt brukervennlighet.

Den originale datafilen er uansett tilgjengelig for senere bruk om vi endrer metodikk igjen underveis.

Anslagsvis 45 sekunder på å legge til et element i listen. For 750 elementer utgjør dette rundt 10 timers jobb, kanskje ned mot 9 timer.

Funksjoner:

|  |  |
| --- | --- |
| **Need to have** | **Nice to have** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |