H4 Applikation udvikling

Af Aputsiaq Sandgreen Lindhardt & Mark Balleby Hansen

Indhold

[Applikationskrav 2](#_Toc79572232)

[Krav-1 2](#_Toc79572233)

[Krav-2 2](#_Toc79572234)

[Applikation Design 3](#_Toc79572235)

[Farvevalg 3](#_Toc79572236)

[Mørk Tilstand 4](#_Toc79572237)

[Skrifttype(r) 4](#_Toc79572238)

[Opbygning 5](#_Toc79572239)

[Funktionalitet 5](#_Toc79572240)

[Applikation Test 6](#_Toc79572241)

# 

# Applikationskrav

## Krav-1

1: Eleven kan udvikle en simpel app til mobile enheder.

2: Eleven kan redegøre for filstrukturen i et native app-projekt i udviklingsværktøjet.

3: Eleven kan redegøre for en app's Life Cycle, dvs. hvilke metoder der udføres når en app startes, skjules, vises og afsluttes.

4: Eleven kan programmere en simpel brugerflade, eksempelvis en knap og et tekstfelt.

5: Eleven kan med et grafisk værktøj designe en brugerflade med de mest almindelige elementer.

6: Eleven kan redegøre for og anvende relevante eventhandlers for brugerfladen.

7: Eleven kan via programkode starte andre programmer/processer på enheden f.eks. en browser.

8: Eleven kan benytte dialogbokse til beskeder og svar fra brugeren.

9: Eleven kan oprette forskellige brugerflader i samme app, hvor der kan overføres data imellem disse.

10: Eleven kan beskrive de væsentligste forskelle på udviklingsmiljøer til forskellige mobil-platforme.

11: Eleven kan dokumentere og kommentere sin kode.

12: Eleven kan gennem en brugertest dokumentere funktionaliteten og brugeroplevelsen i en udviklet app.

## Krav-2

1: Eleven kan udvikle en appløsning, der kan afvikles på forskellige mobile platformsteknologier.

2: Eleven kan installere, konfigurere og anvende et cross-platformsudviklingsmiljø.

3: Eleven kan programmere en appløsning med menuer.

4: Eleven kan programmere en appløsning, der kan vise dynamisk oprettede lister på skærmen og reagere på brugerens valg.

5: Eleven kan programmere en appløsning, der kan gemme persistent data lokalt på enheden.

6: Eleven kan programmere en appløsning, der kan anvende enhedens sensorer f.eks. gyroskop, accelerometer, temperatur osv.

7: Eleven kan tilpasse en responsiv brugerflade, der kan orientere sig efter landskab eller portræt.

8: Eleven kan programmere en appløsning, der kan anvende touch-input fra enhedens skærm, og ved berøring flytte et grafisk objekt rundt på skærmen.

9: Eleven kan programmatisk få et grafisk element til at bevæge sig kontinuerligt på skærmen.

10: Eleven kan udvikle et program, hvor sensorer, touch-skærmen og objekter, der bevæger sig, fungerer sammen på en enkel måde.

11: Eleven kan benytte f.eks. Repository Pattern og Dependency Injection til minimering af afhængigheder mellem moduler.

12: Eleven kan benytte Inversion of Control Pattern (f.eks. Dependency Injection).

13: Eleven kan programmere en brugergrænseflade med asynkrone kald.

14: Eleven kan programmere asynkrone server services i forbindelse med web API.

15: Eleven kan gennem en brugertest dokumentere funktionaliteten og brugeroplevelsen i en udviklet appløsning.

16: Eleven kan redegøre for mulige sikkerhedsproblemer, der skal tages højde for i forbindelse med udvikling af appløsninger.

# Applikation Design

## Farvevalg

Farvevalget til en start er simpelt, brugervenligt, og ensartet. Der blev valgt farven #48C7D9 som er en lys blå nuance. Til Elementer er farven sat til hvid (#FFFFFF), med en gennemsigtighed på 30%.

Farvevalget er lyst, opmuntrende, og kan nemt ses både i lyse solskinstimer, og mørke nætter.

Figur 1 Første App Design

Den hvide gennemsigtighed på elementer står godt til baggrunden, og fremhæver tekst/ikoner på skærmen.

(Figma)

Ved enhedstest så det pænt & simpelt ud ved normal tilstand. Men ved omskiftning til Øjn Venlig tilstand på enhed, skabte farverne en så stor forvrængning i det blå design, at det blev hårdt at se på.

Det kan ikke ses tydeligt på underneden billeder, det er tydeligere ved det blotte øje.

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

Figur 2 Blåt Lys Aktivt

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

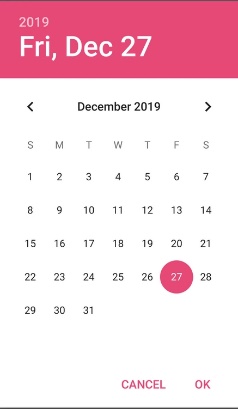
Figur 3 Øjn Venligt Lys Aktivt

Derfor bliver det nuværende farvevalg droppet, på baggrund af at mange har automatisk øjn venlig tilstand slået til på enheder.

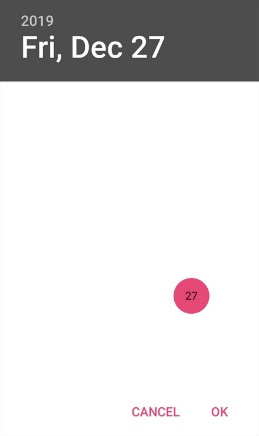
### Mørk Tilstand

I mørk tilstand på enheden, skabte det nuværende design også problemer. I ”Select Day” på figur 1, var hensigten at åbne en datepicker, hvor brugeren kan vælge en bestemt dag.

Farverne i lys tilstand er som de skal være, men i mørk tilstand er der en kendt bug. I mørk tilstand forsvinder datoerne, da at baggrunden bliver hvid, og skriftfarven bliver hvid. <https://github.com/xamarin/Xamarin.Forms/issues/9033>



Figur 4 Datepicker i lys tilstand



Figur 5 Datepicker i mørk tilstand

## Skrifttype(r)

Skrifttype på designet bliver primært valgt til Helvetica, da den er meget læsbar, og til tids dato stadig er den mest brugte skrifttype. Helvetica eksempel.

Den sekundære skrifttype er valgt til Roboto, da det er en skrifttype direkte udviklet fra Google til apps. Roboto eksempel.

De to skrifttyper minder meget om hinanden, dels samme nemme læsbarhed. Ældre og nedsat synede mm. personer vil ikke have et problem med at læse disse skrifttyper.

Skrifttypen på Figur 1 er sat til Roboto, da at design programmet ikke havde mulighed for at vælge Helvetica.

## Opbygning

Applikation er opbygget i sektioner, øverst findes der et søgefelt, hvor der kan søges efter by/sted. Herunder findes der et simpelt overblik over nuværende vejr. Derunder kan man se et overblik over vejr pr. time. I bunden findes der en lille menu med to knapper.

* Søgefelt
* Simpelt overblik
* Liste overblik
* Menu

Til start vil applikationen tage udgangspunkt i din nuværende lokation, dette ville dog kunne ændres i en indstillingsmenu, hvor at der tilføjes en knap i bundmenuen.

På ”Select Day” vil brugeren kunne vælge en bestemt dag, 6 dage frem, for at se vejret for den valgte dag.

## Funktionalitet

Applikationen skal være yderst brugervenlig. Oppetiden skal være 90%, det betyder vejrdataene skal komme fra flere kilder for at sikre oppetid.

Applikationen skal have en kort responstid, og maksimum 3 sekunders ventetid.

Baggrunden på applikationen skal skifte alt efter det nuværende vejr på brugeres lokation, derudover må det gerne indeholde bevægende animationer. Dette skal brugeren kunne slå fra via. indstillinger.

Brugeren skal yderligere kunne slå lokationsdata fra, og give en standard placering for visning af vejr.

Som standard er lokationstjenesten slået til, og tager brugeres lokation som standard.

Brugeren skal have adgang til at ændre baggrund efter brugeres valg, eller bruge standardbaggrunden.

Figur Applikation Livscyklus

Et billede, der indeholder tekst

Automatisk genereret beskrivelse

# Applikation Design 2

Farvevalget på Design 2 er meget lignede Design 1. Den store ændring ligger i baggrunden, hvor der er gået fra en blå skærm til et billede af bestemt vejrtype afhængig af vejret. 

Figur Design 2

Ved test blev designet droppet da at teksten bliver for utydeligt for brugeren at læse.

De skiftende baggrunde ville desuden betyde at skrift farven skulle ændres i takt med baggrunden for at fremstå mest muligt herefter.

# Applikation Design 3

## Farvevalg