|  |  |
| --- | --- |
| Roameo  Your personal travel planner | Procesrapport  Aarhus Tech  Jonas Wittstrøm  Svendeprøve – Datatekniker programmering |

Indhold

[Casebeskrivelse 2](#_Toc193360794)

[Problemformulering 2](#_Toc193360795)

[Metode 2](#_Toc193360796)

[Anvendte fagområder 2](#_Toc193360797)

[Anvendte teknoligier 2](#_Toc193360798)

[Estimerede tidsplaner 2](#_Toc193360799)

[Estimeret flowchart 2](#_Toc193360800)

[Mockups 2](#_Toc193360801)

[Udviklingsprocessen 2](#_Toc193360802)

[.NET MAUI 2](#_Toc193360803)

[AWS 2](#_Toc193360804)

[Google APIs 2](#_Toc193360805)

[Opsætning af sider 2](#_Toc193360806)

[Brugerhåndtering 2](#_Toc193360807)

[Generering af planer 2](#_Toc193360808)

[Realiseret tidsplan 2](#_Toc193360809)

[Logbog (Bilag?) 2](#_Toc193360810)

[Perspektivering 3](#_Toc193360811)

[Konklusion 3](#_Toc193360812)

# Casebeskrivelse

Formålet med denne app, er at hjælpe rejsende med at planlægge deres ture på en struktureret, overskuelig og visuel måde. Den kombinerer en dag-til-dag rejseplanlægger med integration af offentlig transport og brugerens personlige præferencer for mad og aktiviteter

# Problemformulering

Hvordan kan jeg udvikle en brugervenlig rejseplanlægningsapp, der kombinerer dag-til-dag planlægning, kortintegration, offentlig transport og personlige anbefalinger?

# Metode

## Anvendte fagområder

Jeg skriver her i punktform de fagområder jeg berører under mit projekt.

* Client-side programmering
* Server-side programmering
* Objektorienteret programmering (OOP)
* App-programmering
* Versionering
* Cloud-teknologi

Jeg føler at disse spænder bredt, og giver god mulighed for at vise hvad jeg kan inde for mange forskellige områder.

## Anvendte teknologier

Jeg skriver her i punktform de teknologier, som er brugt i mit projekt. Efterfølgende uddyber jeg hvorfor jeg har taget nogle af disse valg frem for andre, samt deres fordele og ulemper.

* Visual Studio Community 2022
* Github
  + Versionering
* ASP .NET
  + MAUI
  + Core Web API
* Amazon Web Services (AWS)
  + Lambda
  + API Gateway
  + DynamoDB (NoSQL)
  + AWS NoSQL Workbench
* Google Maps + APIs
  + Navigation
  + Geocoding
  + Places
* PlantUML
  + Som plugin i Obsidian (Notat-software)
  + Flowchart

### Begrundelse af valg

Valget af Visual Studio (VS22) var nemt, da det for det første er den IDE jeg har arbejdet med mest – både privat og på min læreplads. Udover dette, er det specielt bygget for .NET applikationer og tilbyder skabeloner for de fleste projekttyper, så man hurtigt kan komme i gang.   
Jeg kunne godt have lavet et tilsvarende projekt i Visual Studio Code (VSCode), som er mere letvægt (kræver færre ressourcer af din maskine) og mere fleksibelt på mange punkter.   
Hvis jeg lavede mit projekt i eksempelvis React Native frem for .NET, havde det givet meget bedre mening med VSCode.

Som nævnt ovenfor, valgte jeg .NET MAUI, da jeg i forvejen er mest komfortabel med .NET frameworket, som er det jeg har arbejdet mest med gennem hele uddannelsen – både i skole og på min læreplads. Denne løsning gør det også muligt at skrive apps til flere platforme på samme tid med en fælles kodebase, uden at jeg behøver at lære at skrive til Android eller Apple specifikt.   
Havde dette ikke været tilfældet, kunne en anden god løsning have været React Native, som er baseret på JavaScript, og har et stort fællesskab og et bredt udvalg af plugins og biblioteker. Dog kræver det også at man har noget dybere kendskab til de platforme man udvikler til, end hvad jeg har.

Grunden til at jeg valgte at gøre brug af AWS til mit projekt frem for Azure, som måske havde været det mere naturlige valg, var fordi jeg på hovedforløb 5 havde forsøgt netop dette, men fandt det meget svært at sætte op. Det endte derfor med at jeg ikke havde et produkt at vise efter flere ugers arbejde, fordi jeg sad meget fast.   
Azure ville dog give god mening, da det allerede er en integreret del af Microsoft, og på papiret skulle være meget nemt at sætte op til et projekt som mit.  
Desuden fandt jeg også opsætningsprocessen i AWS meget let, især deres UI var intuitivt at finde rundt i. Jeg følte derfor jeg kunne bruge mere tid og kræfter på udviklingen af mit projekt.

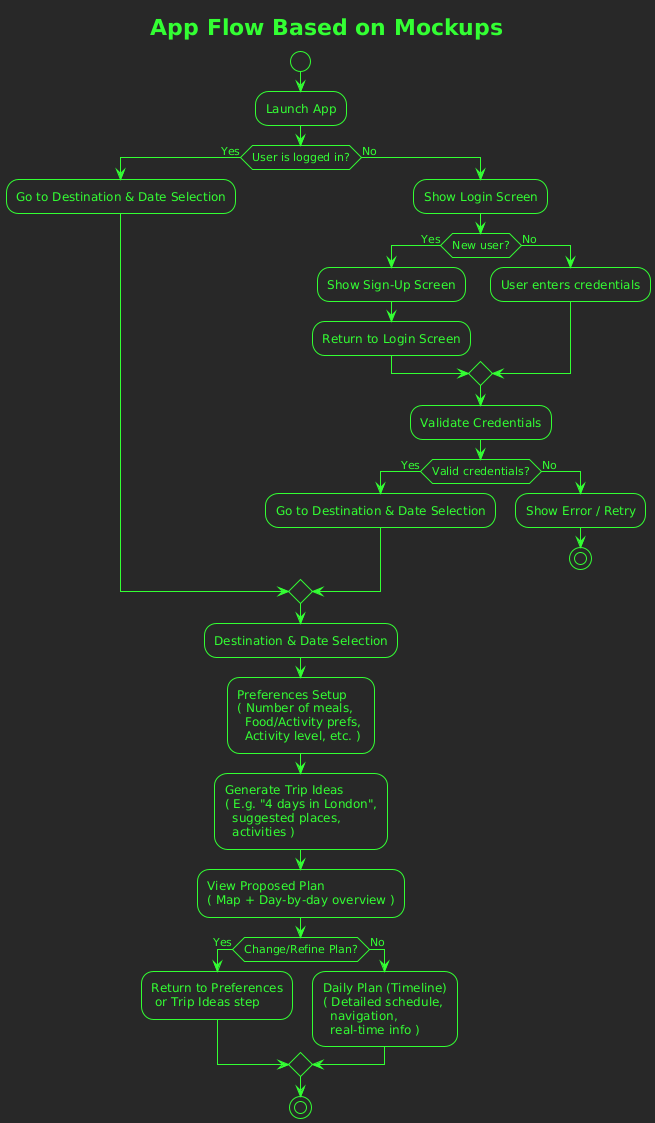
## Estimeret tidsplan



Ovenfor ses min estimerede tidsplan for projektet.

## Estimeret flowchart

Her vises et flowchart over, hvordan jeg havde tænkt mig at flowet skulle være, før jeg startede med at udvikle projektet.



## Mockups

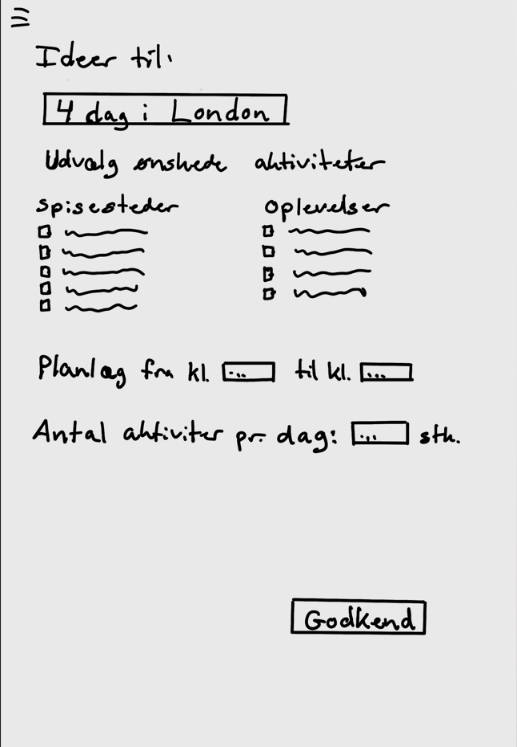
I dette afsnit vil jeg gennemgå de mockups jeg fik lavet før jeg startede udviklingen.

# 

Forside / Destinationsvalgsside. (DestinationPickerPage)



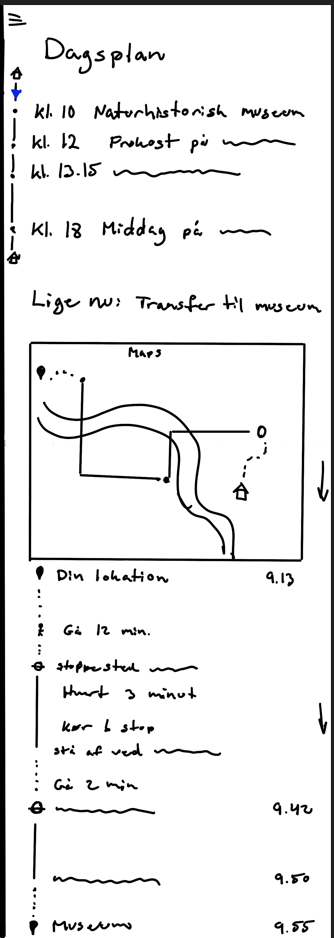
Præferencevalgssiden (UserPreferencesPage)



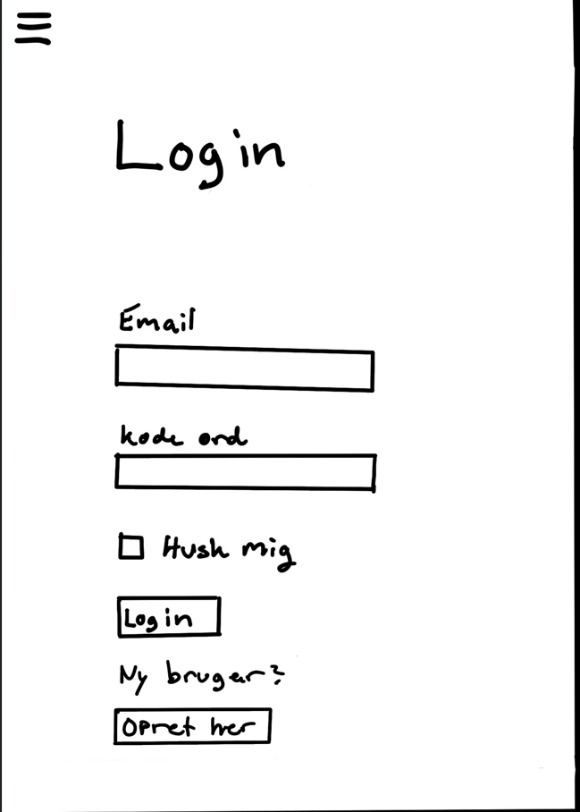
Forslagssiden (SuggestionsPage)



Siden for foreslåede dagssplaner (SuggestedDayPlansPage)



Siden for detaljeret dagsplan (DetailedDayPlansPage)



Login side.

# Udviklingsprocessen

Hvad gik godt/skidt? Hvordan blev problemet løst?

## .NET MAUI

## AWS

## Google APIs

## Opsætning af sider

## Brugerhåndtering

## Generering af planer

## Realiseret tidsplan

Hvordan afviger den fra den estimerede? Hvorfor?

## Logbog (Bilag?)

# Perspektivering

# Konklusion

# Kilder og referencer

## YouTube

[Amazon DynamoDB in .NET - The Getting Started Guide](https://www.youtube.com/watch?v=BJYDWMN8taI) – Mukesh Murugan

[Deploy .Net Core Web API (.Net 8) to AWS Lambda Function from VS2022 and Config with The API Gateway](https://www.youtube.com/watch?v=9a-wpPg0QyI) – Lucky Happy

## AI

[ChatGPT](https://chatgpt.com/)

GitHub Copilot

## Grafik

Michelle Wittstrøm Petersen – Mockups, app ikon og forsidebillede.

# Bilag