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

Indhold

[Indledning 2](#_Toc193368514)

[Casebeskrivelse 2](#_Toc193368515)

[Overordnet Funktionalitet 2](#_Toc193368516)

[Kravspecifikation 2](#_Toc193368517)

[Funktionelle krav 2](#_Toc193368518)

[Ikke-funktionelle krav 2](#_Toc193368519)

[Produktbeskrivelse 3](#_Toc193368520)

[Teknisk oversigt 3](#_Toc193368521)

[Frontend 3](#_Toc193368522)

[Backend 3](#_Toc193368523)

[Database og data persistens 3](#_Toc193368524)

[Diagrammer 3](#_Toc193368525)

[Flowchart (bilag 1) 3](#_Toc193368526)

[ER-diagram (bilag 2) 4](#_Toc193368527)

[Vejledning 5](#_Toc193368528)

[Installationsvejledning uden Visual Studio installeret 5](#_Toc193368529)

[Installation med Visual Studio allerede installeret 6](#_Toc193368530)

[Brugervejledning 7](#_Toc193368531)

[Test og validering 22](#_Toc193368532)

[Case 1: Indtastning af destination og navigationsflow gennem appen 22](#_Toc193368533)

[Case 2: Alternering mellem spisesteder og oplevelser på planer 23](#_Toc193368534)

[GenerateDayPlans-metoden. 23](#_Toc193368535)

[Konklusion 24](#_Toc193368536)

[Muligheder for videreudvikling 24](#_Toc193368537)

[Kilder og referencer 24](#_Toc193368538)

[YouTube 24](#_Toc193368539)

[AI 24](#_Toc193368540)

[Grafik 24](#_Toc193368541)

[Bilag 25](#_Toc193368542)

[Bilag 1 – Flowchart 25](#_Toc193368543)

[Bilag 2 – ER Diagram over de mest centrale modeller 26](#_Toc193368544)

[Bilag 3 – GenerateDayPlans-metoden. 27](#_Toc193368545)

# Indledning

I dette afsnit vil jeg kort gennemgå følgende: En kort beskrivelse af min case, produktets formål og en overordnet beskrivelse af produktets funktionalitet.

## 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

## Overordnet Funktionalitet

Appen hjælper brugeren med at planlægge en personlig rejse ud fra egne præferencer. Brugeren vælger først destination samt datoer for rejsen. Derefter indtastes ønsker om mad, aktiviteter og generelle præferencer.

Ud fra disse valg foreslår appen relevante aktiviteter og spisesteder, som brugeren kan tilpasse efter behov. Når brugeren har godkendt den foreslåede rejseplan, genererer appen en overskuelig dag-til-dag oversigt med aktiviteter og måltider.

Under rejsen kan brugeren starte sin valgte dag direkte fra appen, vælge transportform, og få rutevejledning til aktiviteterne via integration med Google Maps.

# Kravspecifikation

I dette afsnit vil jeg gennemgå både de funktionelle og ikke-funktionelle krav til min app.

## Funktionelle krav

Her kommer en liste over appens funktionelle krav i punktform:

* Brugeren skal kunne vælge en ønsket destination for sin rejse.
* Brugeren skal kunne sætte start- og slutdato for sin rejse.
* Brugeren skal kunne sætte sine præferencer for spisning, oplevelser og aktivitetsniveau
* Brugeren skal kunne logge ind.
* Brugeren skal kunne lukke appen ned, og fortsætte med at lave deres planer på et senere tidspunkt.
* Brugeren skal kunne dele sin planlagte rejse med eventuelle rejsemakkere.
* Appen skal kunne præsentere brugeren for forslag til spisesteder og oplevelser, baseret på brugerens valg.
* Brugeren skal kunne sætte et ønsket tidsrum for sine dagsplaner.
* Appen skal kunne præsentere brugeren for foreslåede dagsplaner baseret på brugerens valg.
* Brugeren skal have mulighed for at ændre sine planer inden endelig bekræftelse.
* Appen skal præsentere brugeren for valgte dagsplaner på en let og overskuelig måde.
* Appen skal gøre det nemt for brugeren at starte navigation via Google Maps fra sted til sted.

## Ikke-funktionelle krav

Ligesom ved de funktionelle krav, kommer her de ikke-funktionelle i punktform.

* Appen skal være nem og hurtig at starte.
* Brugerrejsen gennem appen skal være intuitiv.
* Appens brugergrænseflade (UI) skal være overskuelig.
* Appen skal leve op til GDPR-lovgivningen.
* Appen skal kunne køre på alle nyere Android og Apple enheder.
* Der skal være tydelig feedback til brugeren angående fejl i appen, ved manglende internetforbindelse og lign.

# Produktbeskrivelse

Dette afsnit deler jeg op i to underafsnit: En teknisk oversigt, der dækker over arkitektur, teknologier og platforme. Andet afsnit er diagrammer, som i dette tilfælde er en gennemgang af mit flowdiagram og et mindre ER-diagram for de mest centrale modeller i projektet.

## Teknisk oversigt

Dette afsnit dækker over mine valg af frontend, backend, databaser og datapersistens, integration af eksterne API ‘er og hvordan det hele i sidste ende hænger sammen.

### Frontend

Frontenden er lavet i .NET MAUI, som er en platform af Microsoft, der gør det muligt at udvikle apps til Android og Apple på samme tid, med en fælles samlet kodebase. Selve siderne er skrevet i .xaml format og den bagvedlæggende logik er skrevet i C#.

### Backend

Min backend er en ASP .NET Core Web API, som er deployed hos AWS (Amazon Web Services) som en AWS Lambda funktion. AWS Lambda er en serverløs tjeneste, som kan køre kode uden behov for serveradministration. Der vil blive fortalt mere om dette i procesrapporten.

Jeg bruger AWS API Gateway til at læse og hente data fra min database.

Backend kommunikerer med frontend gennem http kald.

### Database og data persistens

Til lokal lagring på brugerens enhed, bruger jeg SQLite, som er en letvægts, filbaseret database, som gemmer data lokalt. Dette gør den uden brug af servere.

Til styring af fjerndata bruger jeg AWS DynamoDB, som er en NoSQL database.

## Diagrammer

### Flowchart (bilag 1)

I mit endelige projekt nåede jeg ikke at få opsat brugerhåndtering (mere om dette i procesrapporten). Derfor endte mit diagram med at blive meget lineært. Det viser brugerens vej fra valg af destination til at man får sin rutevejledning i Google Maps. Jeg vil alligevel i punktform beskrive brugerens rejse gennem appen.

1. Brugeren starter på destinationsvælgersiden (DestinationPickerPage)
2. Her indtastes destination og start- og slutdato for rejsen
3. Brugerens input valideres:
   1. Hvis det ikke er gyldigt, skal der tastes igen
   2. Hvis det er gyldigt, fortsætter man
4. Navigerer til valg af præferencer (PreferencesPage)
5. Her vælger brugeren sine præferencer:
   1. Antal måltider om dagen
   2. Madpræferencer
   3. Oplevelsespræferencer
6. Præferencer gemmes
7. Appen genererer en midlertidig rejseplan
8. Navigerer til forslagssiden (SuggestionsPage)
9. Her vælger brugeren følgende:
   1. Tidsrum for sine aktiviteter
   2. Antal aktiviteter pr. dag
   3. Spisesteder
   4. Oplevelser
10. Disse valg gemmes og der genereres foreslåede rejseplaner.
11. Der navigeres til siden for foreslåede dagsplaner (SuggestedDayPlansPage)
12. Her kan brugeren vælge at bekræfte eller lave ændringer til sine planer.
    1. Ændr plan (Change Plan):
       1. Navigerer til EditPlansPage
       2. Brugeren foretager sine ændringer af tidligere valg.
       3. Navigerer tilbage til SuggestedDayPlansPage
    2. Bekræft plan (Confirm Plan):
       1. Fortsæt til siden for endelige dagsplaner (FinalizedDayPlansPage)
13. Brugeren gennemgår en oversigt over de genererede dagsplaner.
14. Vælger en bestemt dag og trykker på ”Start Day”
15. Navigerer til siden for de detaljerede dagsplaner (DetailedDayPlansPage)
16. Her vises en detaljeret oversigt over dagens planer.
17. Brugeren kan her vælge sin foretrukne transportmetode og trykke på ”Get Directions”
18. Appen tjekker tilladelse til lokation.
19. Brugerens aktuelle placering hentes.
20. Google Maps åbnes og brugeren kan nu tage hen til sin valgte aktivitet.

### ER-diagram (bilag 2)

Mit ER-diagram viser forholdet mellem de mest centrale datamodeller i min app. Siden jeg ikke fik brugerhåndtering til at virke, bliver det meste data håndteret af min Trip-entitet.

Trip repræsenterer selve rejsen og indeholder som destination, start-og slutdato (StartDate, EndDate). Mere komplekse detaljer som brugerpræferencer (UserPreferences) og Google Maps data, holder den som JSON-strenge.

UserPreferences har en 1:1 relation til Trip og beskriver brugerens præferencer som mad, måltider per dag, aktiviteter og energiniveau.

DayPlan (dagsplaner) har en 1-til-mange-relation med Trip og beskriver planlægningen for hver enkelt dag på turen. DayPlan indeholder et dagnummer, dato og en kort opsummering.

ScheduleItem (punkt på tidsplanen) er knyttet til DayPlan i en 1-til-mange relation, og holder detaljer om de enkelte aktiviteter som navn, tidspunkt, type af aktivitet (spisning eller oplevelse), og en lokationsforespørgsel til Google Maps.

# Vejledning

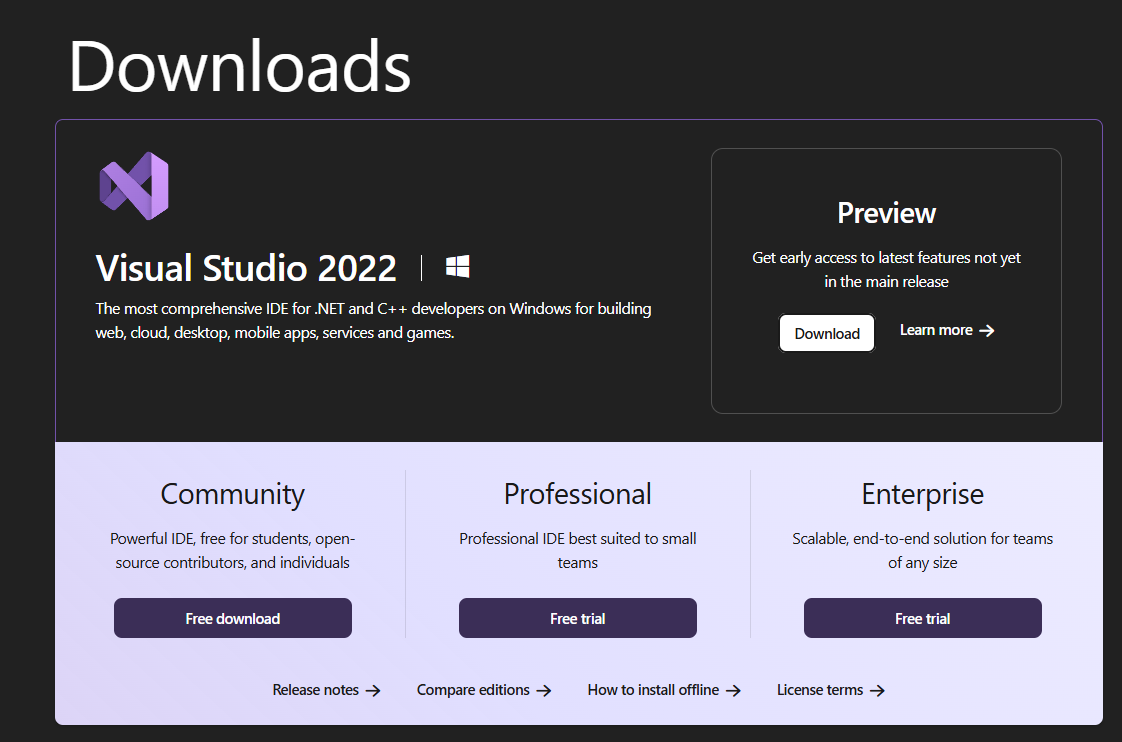
I dette afsnit vil jeg gå igennem en installationsvejledning og en brugervejledning til min app.

## Installationsvejledning uden Visual Studio installeret

Har du allerede Visual Studio, kan du springe dette afsnit over og gå videre til næste.

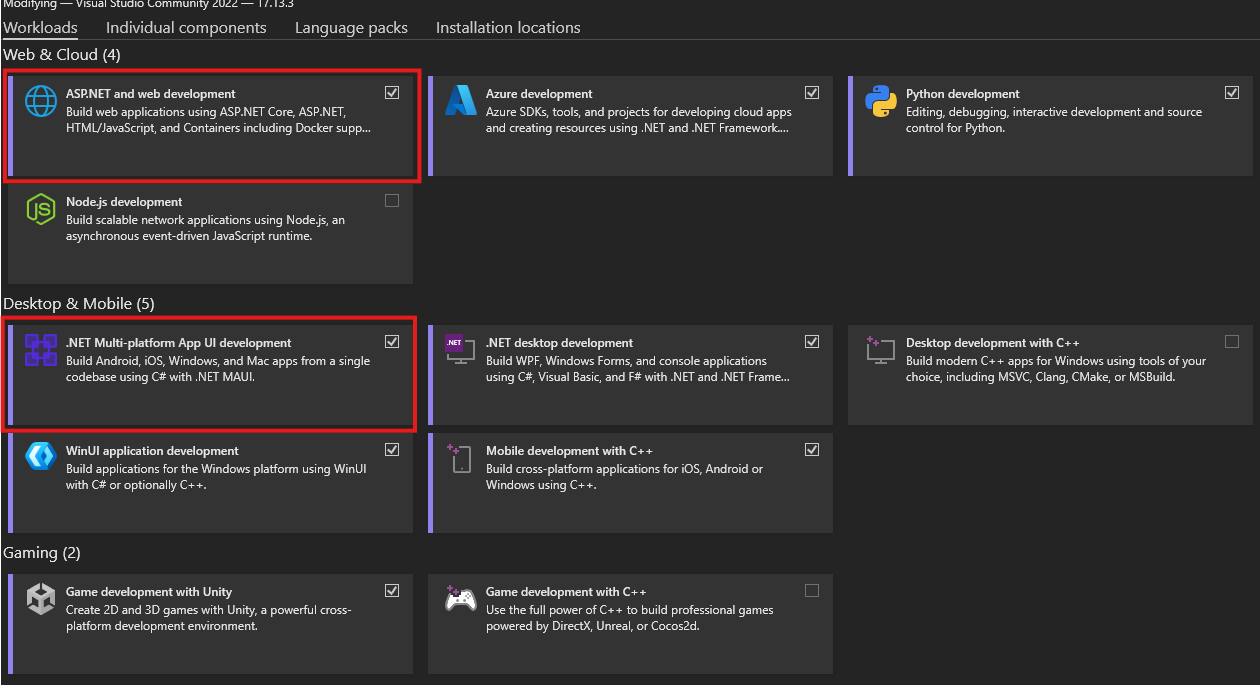
For at installere løsningen og køre appen på dit system, skal du først have Visual Studio 2022 (VS22) installeret på din PC. VS22 kan hentes gratis på <https://visualstudio.microsoft.com/downloads/>.

På ovenstående side bliver du mødt af disse valg:

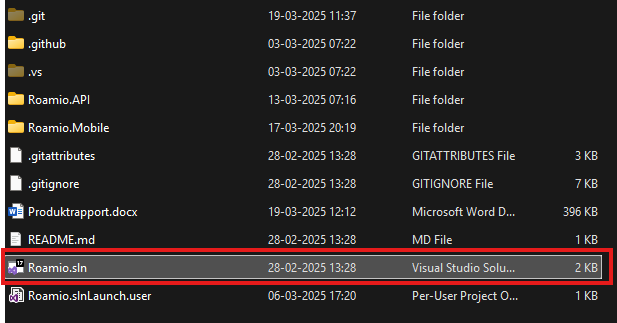


Her vælger du Community, som er gratis.

Under installationen skal der vælges ”Workloads”. Her er det vigtigt at du vælger ”ASP.NET and web development” og ”.NET multi-platform App UI development”.



Disse er nødvendige for at kunne køre min solution (fil for det samlede projekt.)

Når alt dette er gjort, kan du åbne Roamio.sln filen i den afleverede mappe: 

## Installation med Visual Studio allerede installeret

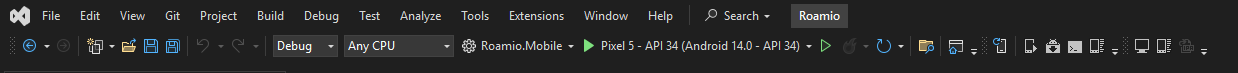
Du starter med at åbne Roamio.sln

For at fortsætte her, er det anbefalet at du tilkobler en mobil enhed til din maskine. Der er mulighed for at køre appen på maskinen med en Android emulator. Men det har jeg ikke testet, jeg kan ikke garantere for hvor godt det virker. Jeg har en gammel mobil liggende, så det er den jeg selv har brugt under udviklingen.

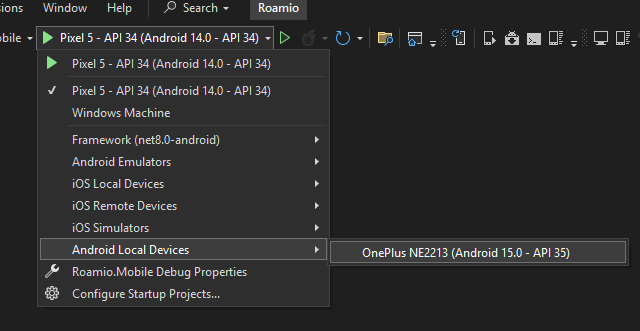
Det er også vigtigt at du har usb-debugging slået til. En guide til at gøre det, kan findes her: <https://www.howtogeek.com/129728/how-to-enable-developer-options-menu-and-enable-and-usb-debugging-on-android/>

Når dette er gjort, er du klar til at køre appen på din enhed.

Inde i VS22, kan du i toppen af skærmen se dette:



Hvor der på billedet står ”Pixel 5 – API 34 (Android 14.0 – API 34 (Dette er en emulator, som jeg har installeret tidligere))”, åbner du dropdown-menuen og vælger din enhed, som kan findes under ”Android Local Devices”. Det skulle gerne se ud noglelunde som dette:

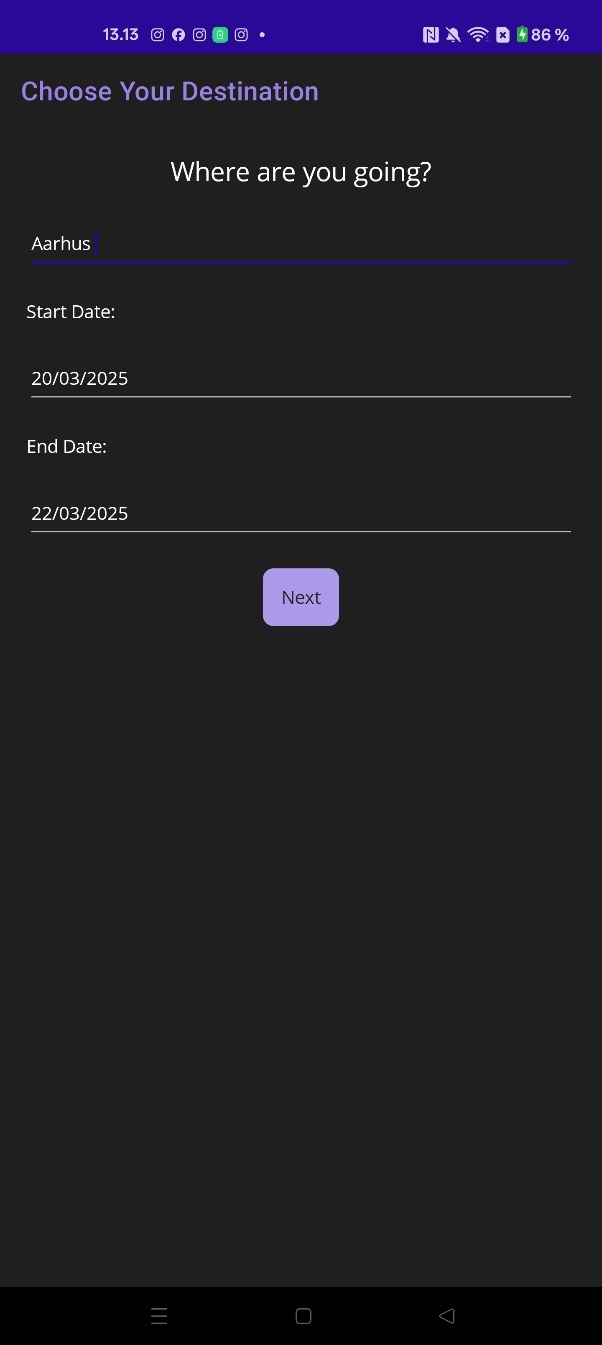


Du kan nu deploy (installere) appen på din enhed ved at trykke på knappen igen, som nu skulle have ændret sig til din valgte enhed.

## Brugervejledning

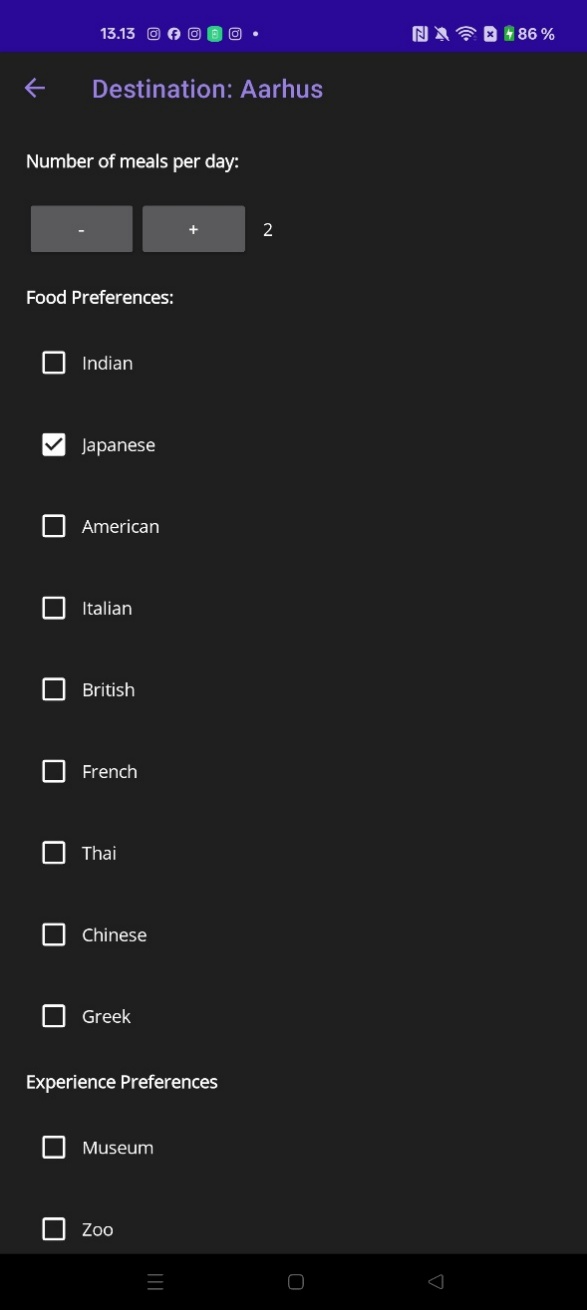
Når du har installeret projektet på din maskine og koblet din enhed (eller emulator) til, er du nu klar til at bruge appen.

Du starter på DestinationPickerPage, som ser sådan ud:

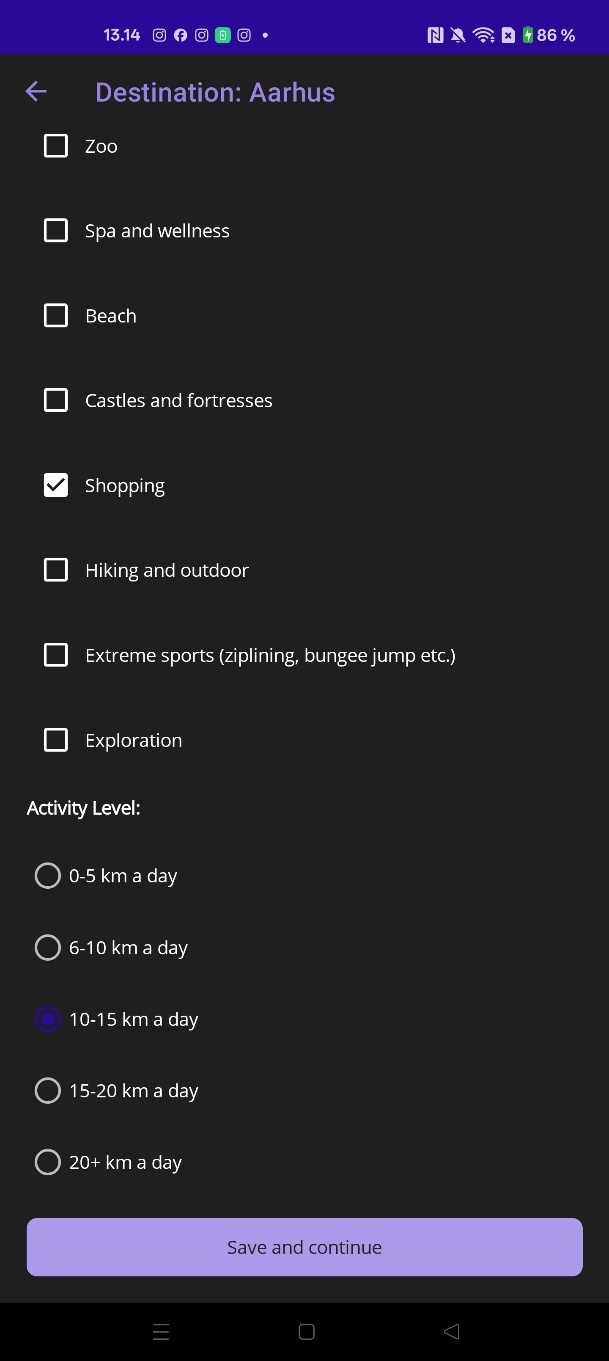
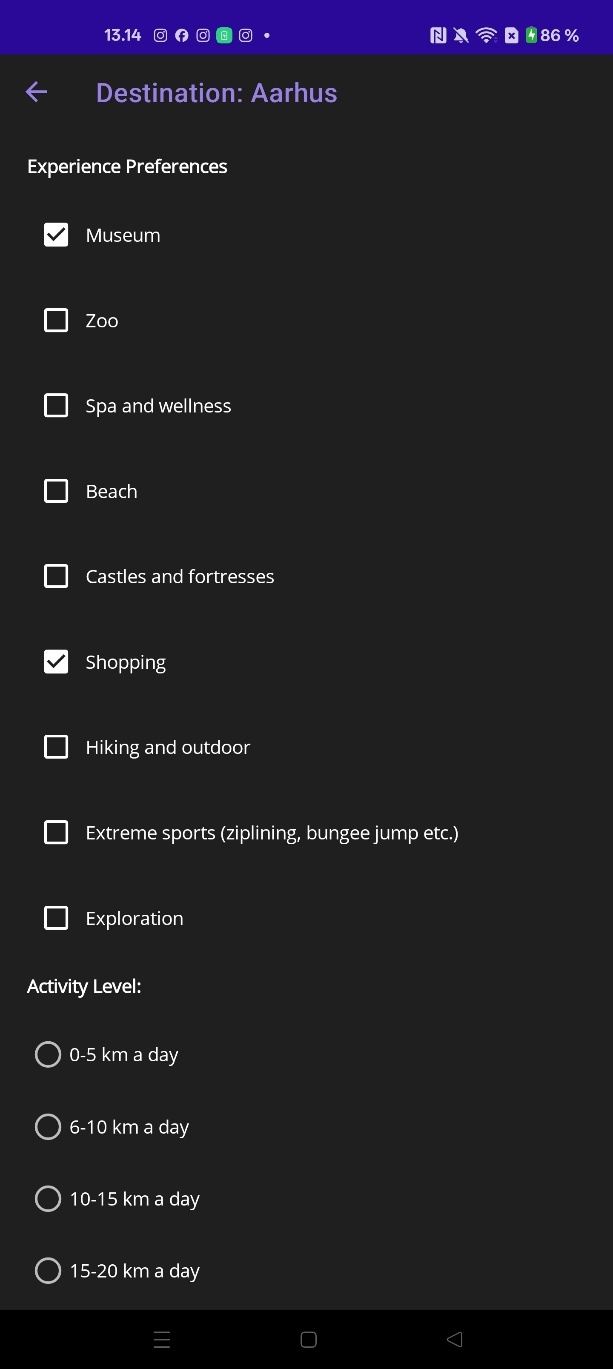


Her starter du med at indtaste destinationen for din rejse og hvilke dage du er afsted. Fordi jeg ikke er særlig kreativ, men har et mindre kendskab til byen, har jeg her bare indtastet Aarhus, og at jeg er afsted fra d. 20/03/2025 – 22/03/25.

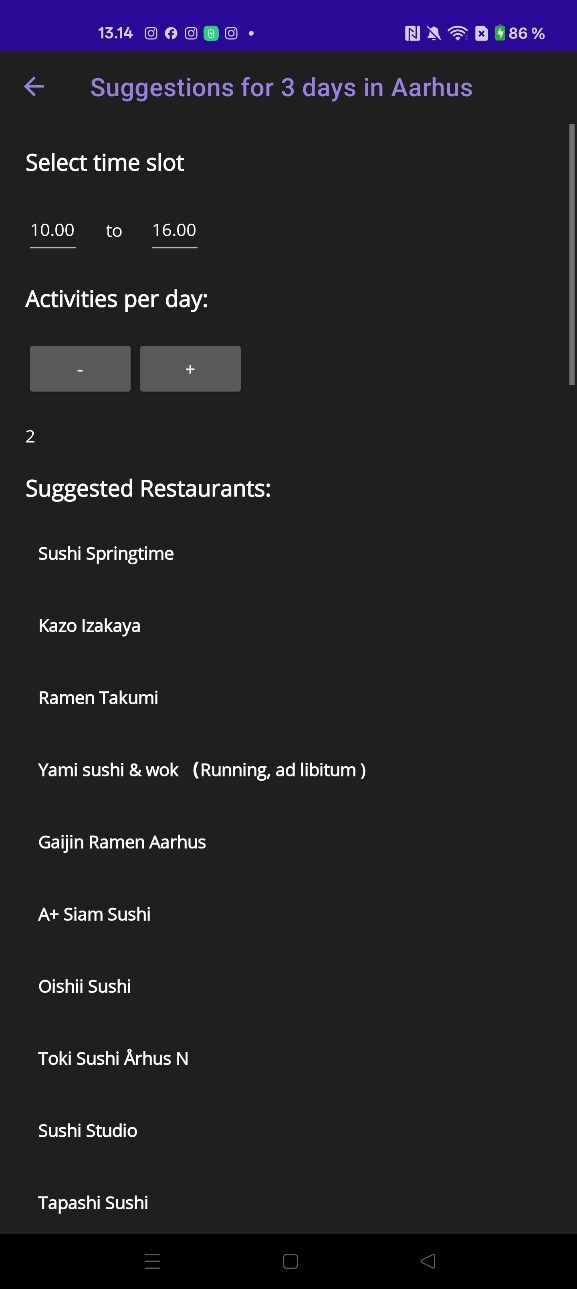
Trykker du på ”Next” bliver du ført til næste side:



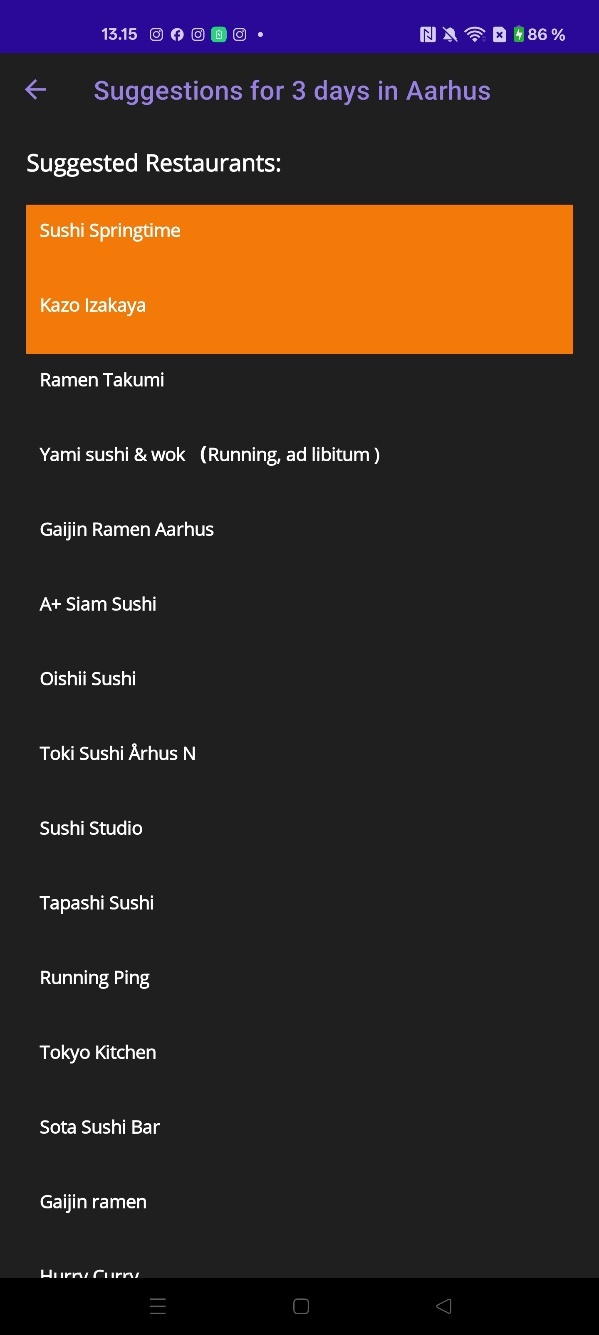
Her starter du med at indtaste antal måltider du regner med at skulle have ude om dagen, og hvilke madpræferencer du har. Længere nede på siden kan du finde valg af aktiviteter, dit energiniveau og knappen til at gemme og fortsætte:



På næste side, skal du først vælge i hvilket tidsrum du ønsker at lægge dagens aktiviteter og antal aktiviteter om dagen.

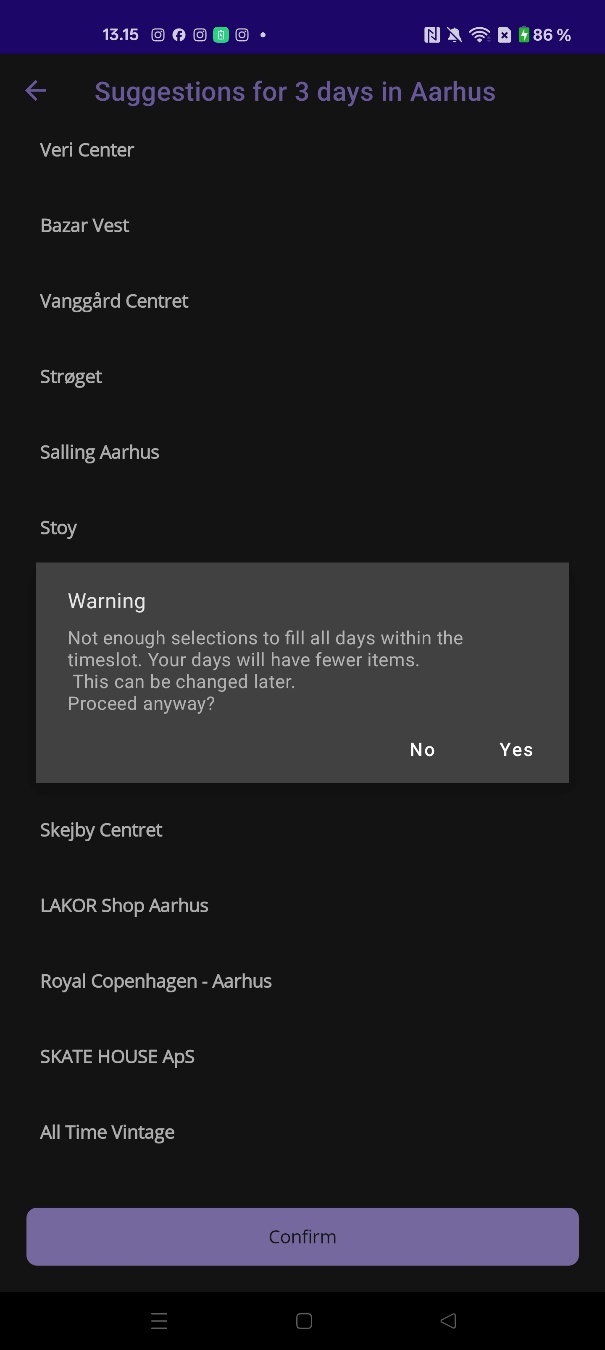


Du bliver på denne side også mødt med en masse forslag, som er baseret på de valg du tog på forrige side. Disse er hentet fra Google. Når du har taget dine valg, kan det se nogenlunde sådan ud:



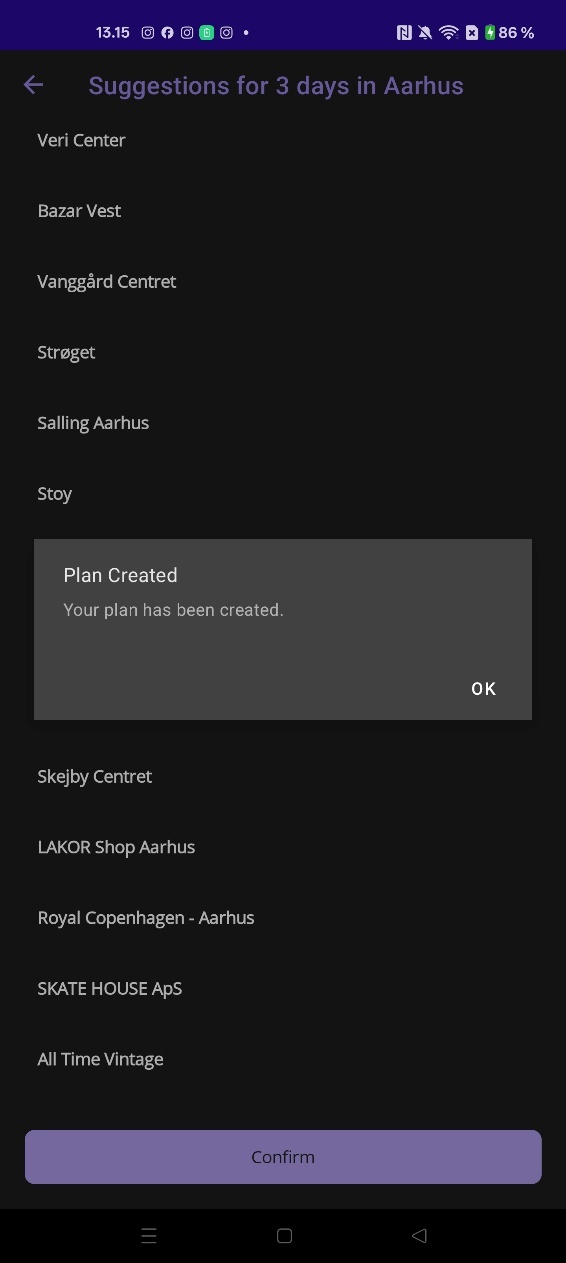
Når du har foretaget dine valg, kan du i bunden af skærmen trykke på ”Confirm” for at bekræfte dine valg.

Alt efter hvor mange dage du har planlagt og hvor mange spisesteder/oplevelser du har valgt inde for dit tidsrum, kan du her blive mødt af denne advarsel:

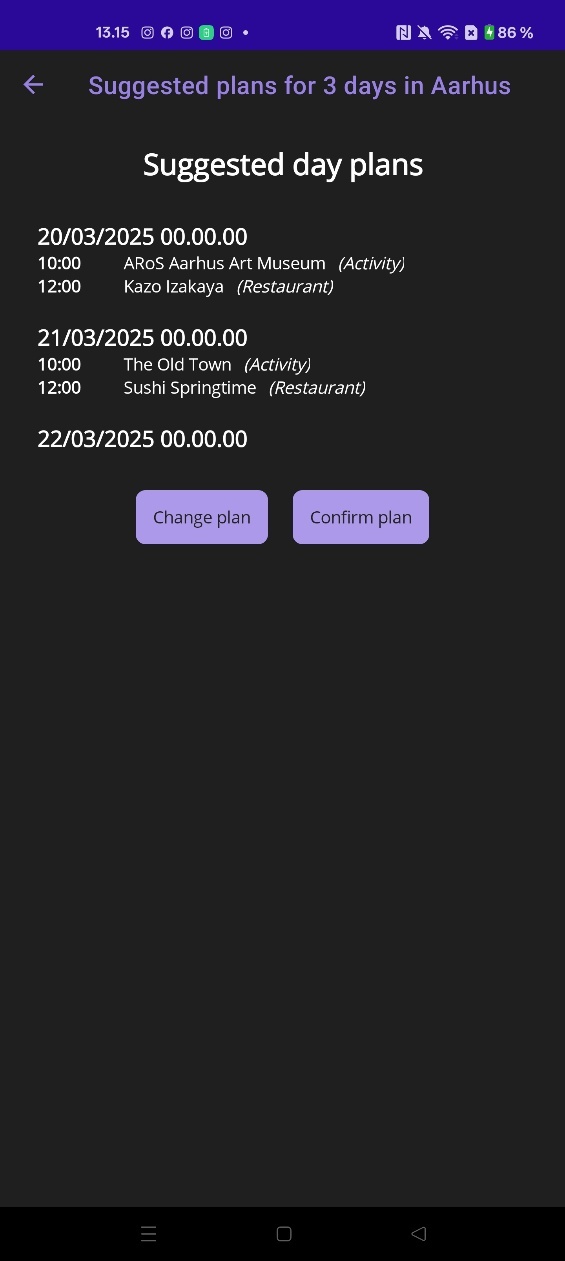


Denne informerer dig bare om at du ikke har taget nok valg til at udfylde al den tid og dage du har valgt tidligere. Men som der står, kan du sagtens ændre det senere. For nu trykker vi bare på ”Yes” og fortsætter.

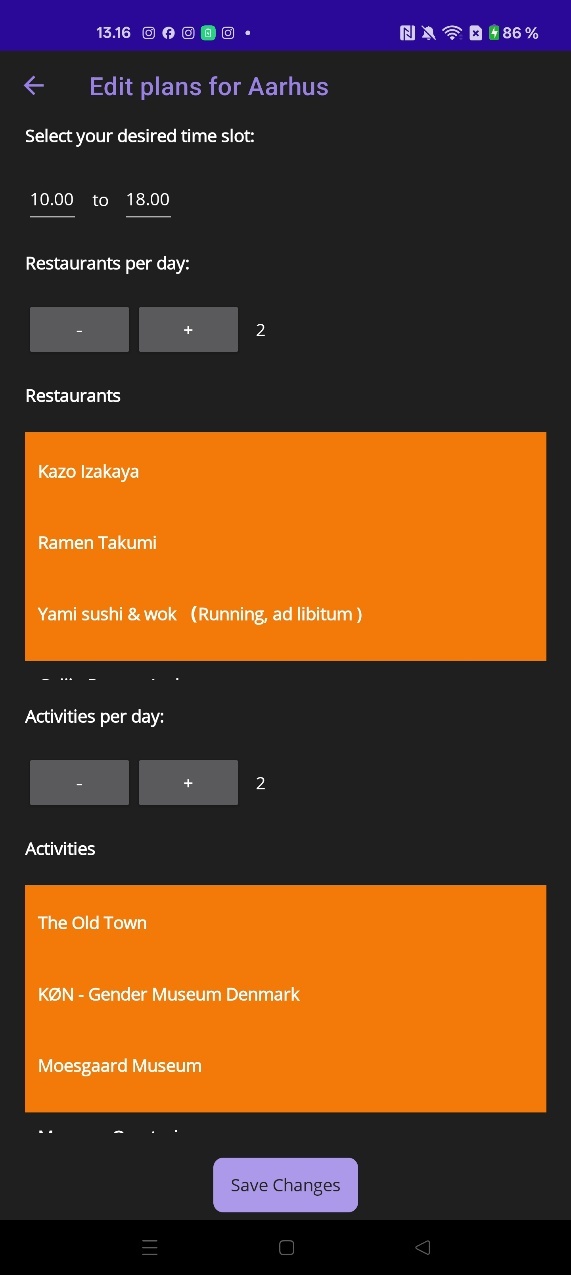
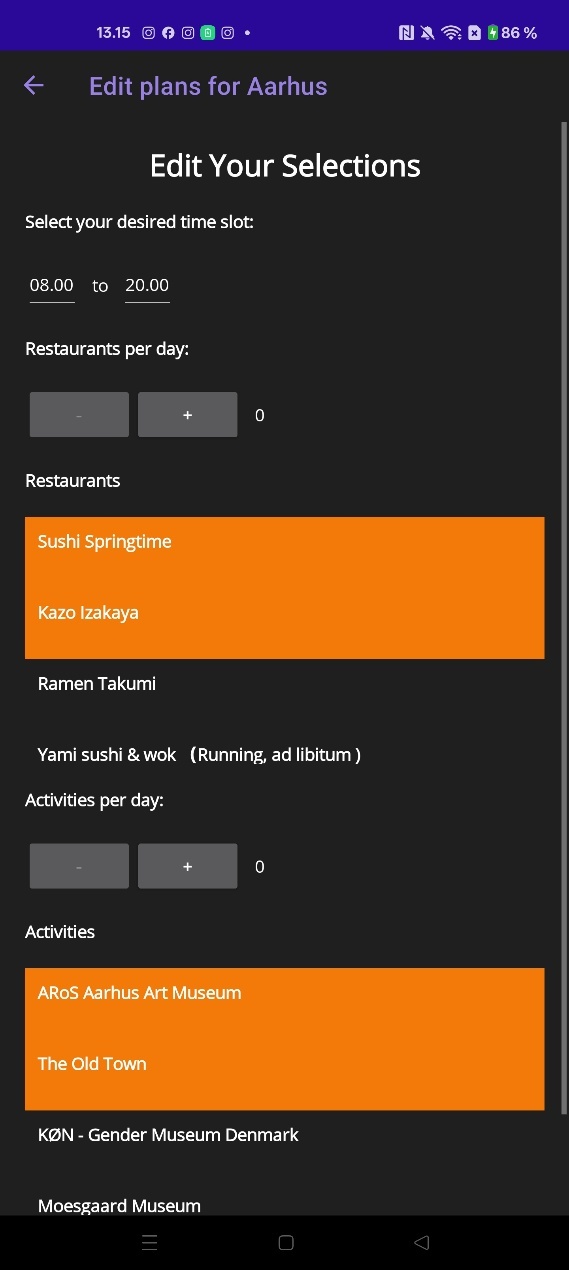
Du får nu dette prompt, som fortæller at din (midlertidige) plan er oprettet:



Efter din plan er blevet oprettet kommer du nu til siden for de foreslåede dagsplaner:

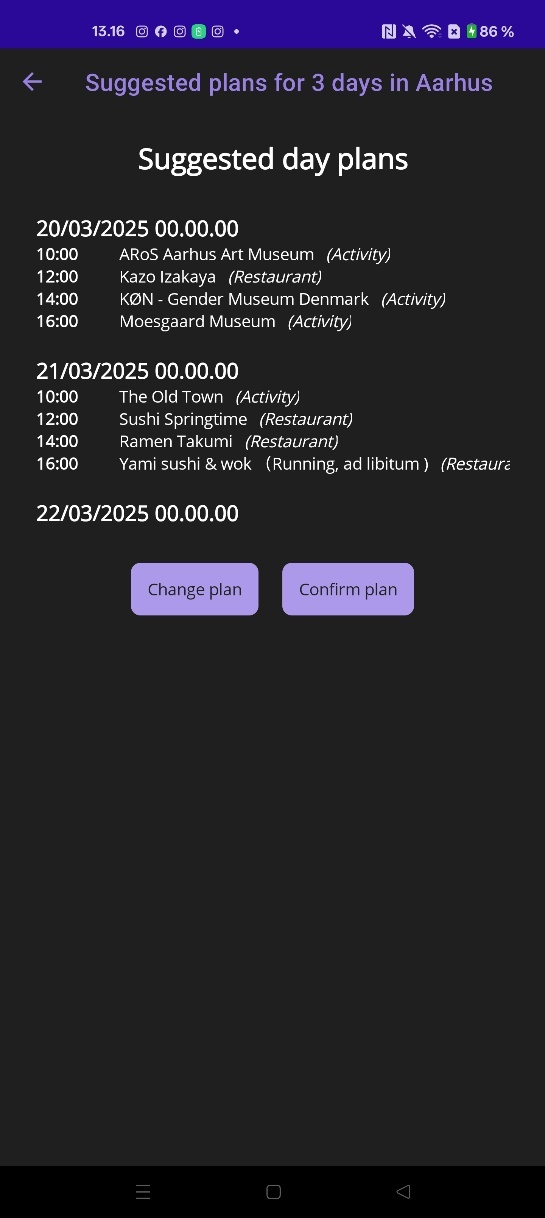
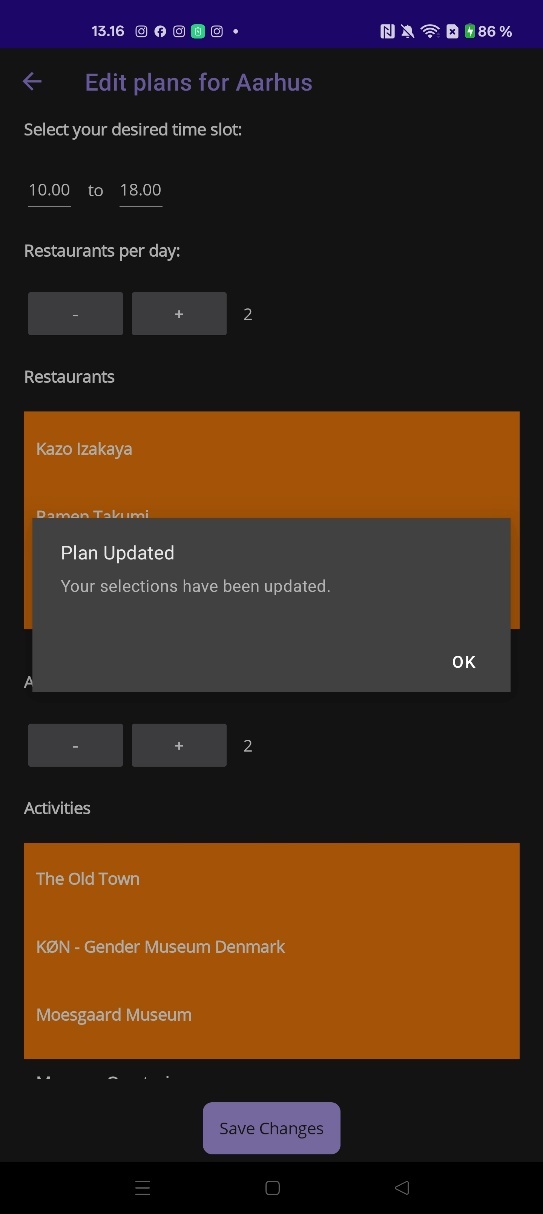


Vi ser her at vi kun har en enkelt aktivitet og spisested sat på de to første dage, og at tredje dag står helt tom. Det kan vi gøre noget ved, ved at trykke på ”Change Plan” så lad os gøre det.



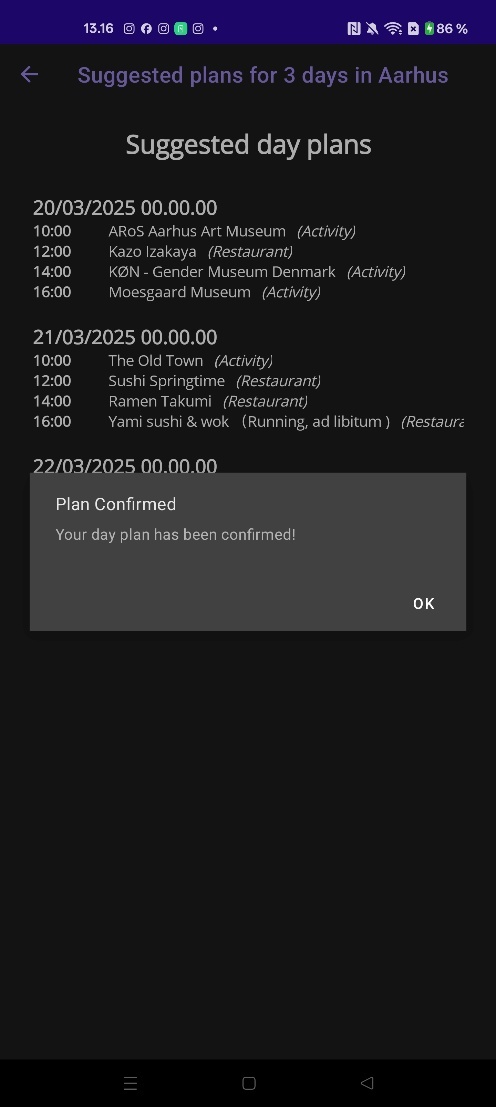
På denne side kan vi redigere alle de valg vi har taget tidligere i appen, som tidsrum, spisesteder pr. dag, oplevelser pr. dag og hvilke spisesteder og oplevelser vi vil tilføje til vores plan. Som standard er vores valg fra tidligere allerede markeret her. Når vi er tilfredse med vores valg, kan vi gemme vores ændringer ved at trykke på ”Save Changes”.

Her får vi en besked med at vores valg er blevet opdateret, før vi bliver ført tilbage til SuggestedDayPlansPage.

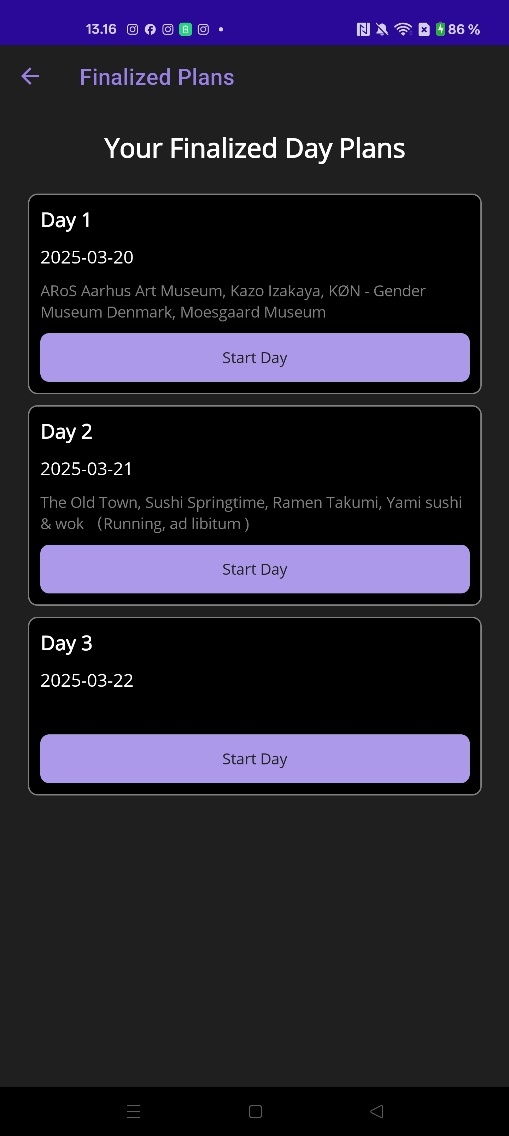


Her kan vi nu se at vores planer er blevet opdateret med vores seneste valg. (Dog havde jeg stadig ikke lavet nok valg, men det er ikke vigtigt for denne gennemgang.)

Hvis du er tilfreds nu, kan du trykke på ”Confirm plan” og får nu en besked om at dine planer er blevet bekræftet:

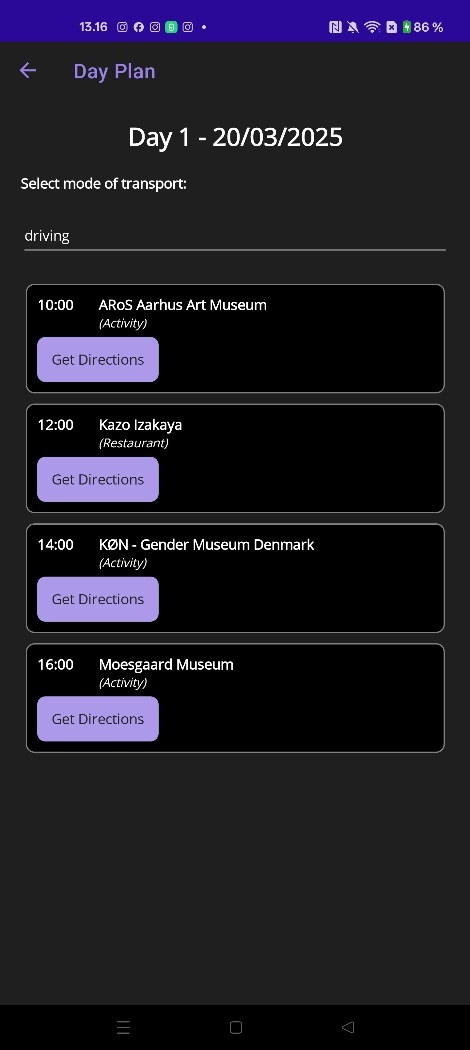


På næste side, kan du nu se en oversigt over de endelige planer for din rejse.

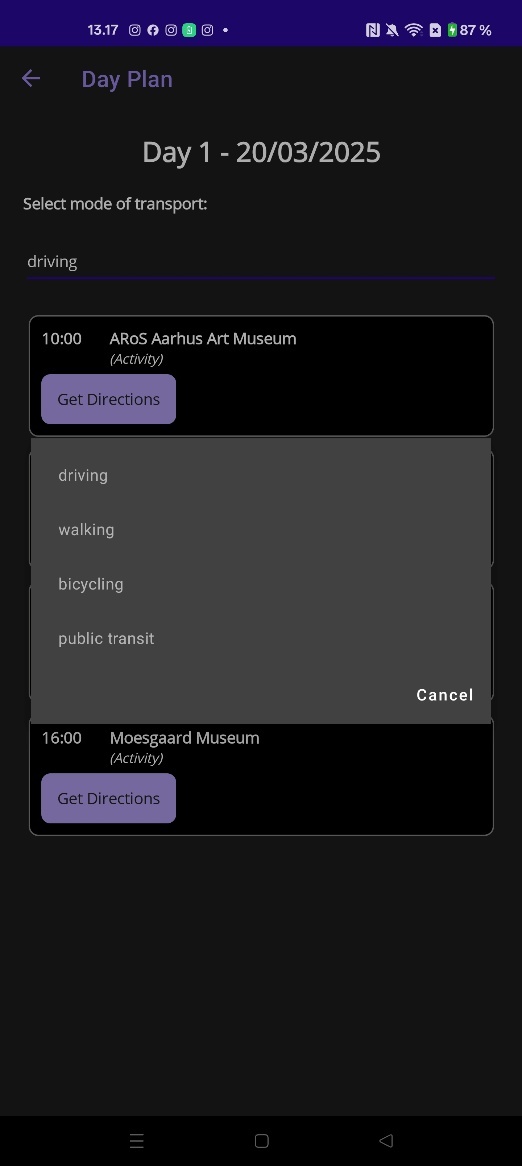


Her er dine dage delt op i enkelte sektioner for hver dag du er afsted. Her kan du se hvilke spisesteder og oplevelser du har valgt for den pågældende dag.

Tryk på ”Start Day” for at begynde din dag.



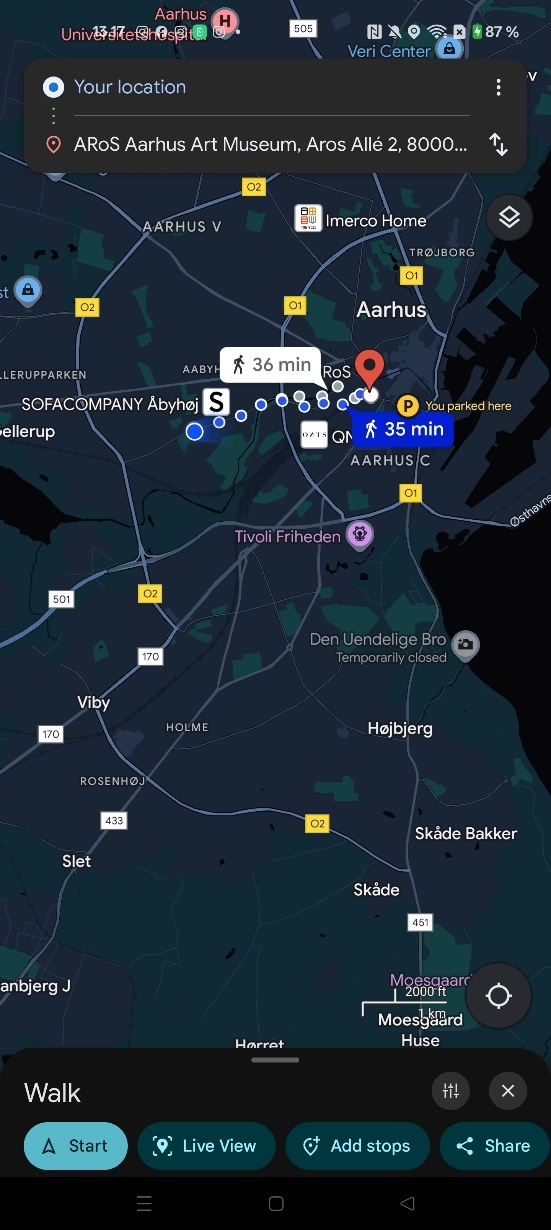
Her ser du en oversigt over de enkelte aktiviteter for den valgte dag. Du kan også vælge din foretrukne transportmetode. Denne kan ændres ved at trykke på feltet, og du bliver her mødt med disse muligheder:



Når du har valgt din foretrukne transportmetode, kan du fortsætte ved at trykke på ”Get Directions”.

Nu kan det ske, at appen spørger om tilladelse til at kende din lokation. Dette step blev skippet for mig, da jeg allerede har givet appen denne tilladelse.

Efter dette, skulle Google Maps-appen gerne åbne automatisk på din app, og vise dig vej fra din nuværende lokation til din aktivitet du har valgt at navigere til.



I mit tilfælde viser den hvor længe det vil tage for mig at gå fra mit hjem til ARoS museet i Århus.

Når du er nået frem og har spist eller er færdig med den pågældende aktivitet, kan åbne Roameo igen og få vist vej til din næste aktivitet eller spisested.

Dette er den komplette guide til hvordan man bruger Roameo.

# Test og validering

I dette afsnit vil jeg beskrive to test-cases, deres formål, metode og hvordan de måler mod kravspecifikationerne.

## Case 1: Indtastning af destination og navigationsflow gennem appen

**Formål:** Verificér at man ved at taste en gyldig destination, start og slutdato og trykker ”Next” på destinationsvalgssiden fører brugeren videre til præferencevalgssiden.

* Input:
  + Destination: ”Aarhus”
  + Start date: ”20/03/2025”
  + End date: ”22/03/2025”
* Metode:
  + Åbn Roameo og start på første side, som er destinationsvalgssiden.
  + Indtast ovenstående og tryk på ”Next”
* Forventet resultat:
  + Appen validerer brugerens input og navigerer videre til præferencevalgssiden.

**Opfyldelse af kravsspecifikation:** Her opflyder appen de funktionelle krav om at brugeren skal kunne indtaste en destination og sætte en start- og slutdato for sin rejse.

## Case 2: Alternering mellem spisesteder og oplevelser på planer

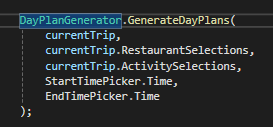
**Formål:** Validér at appen tager højde for at spisesteder og oplevelser er to forskellige entiteter og derfor så vidt muligt undgår at der bliver placeret eksempelvis to spisesteder efter hinanden i en foreslået/endelig dagsplan.

* Input:
  + Restaurants: ”Rest. A”, ”Rest. B”
  + Experiences: “Exp. A”, “Exp. B”
  + Time slot: Start = 10.00, End = 18.00
  + Trip dates: [Hentet fra destinationsvalgssiden]
* Metode:
  + Kald GenerateDayPlans metoden med indputs (vises nedenunder)
* Forventet resultat:
  + Hver genereret dagsplan vil sætte en oplevelse først, og derefter en aktivitet. Herfra vil der skiftevis blive sat oplevelser og spisesteder ind på de genererede dagsplaner. Hvis der er valgt flere oplevelser end spisesteder, vil der kunne være flere oplevelser efter hinanden efter sidste spisested. Den samme aktivitet vil heller ikke kunne ses på flere dage.

**Opfyldelse af kravsspecifikation:** Brugeren kan sætte sit ønskede tidsrum. Og oplever ikke at blive foreslået de samme aktiviteter på samme plan.

### GenerateDayPlans-metoden.

Når der trykkes på ”Confirm” på siden for de forslåede aktiviteter, aktiveres bl.a. dette stykke kode:



Der bliver kaldt til min DayPlanGenerator class, der har en metode der hedder GenerateDayPlans. Denne tager imod de inputs som brugeren indtaster og genererer en dagsplan ud fra disse. Hele metoden kan ses i bilag 3.

# Konklusion

Jeg synes at jeg nåede godt i mål på rigtig mange punkter. Appen gør det hurtigt og nemt at finde inspiration til oplevelser og spisesteder. Dette kan især være positivt, hvis man rejser til et sted, hvor man ikke rigtig er kendt.

Integrationen af Googles API’er for Geocoding, Places og Maps, gør også at appen har et næsten uendeligt forråd af idéer til at inspirere brugeren. Brugeren kan også nemt komme fra sted til sted.

Der er dog også en del mangler, som jeg gerne vil adressere også:

Jeg er ked af at jeg ikke fik implementeret brugerhåndtering på en god måde. De sociale features jeg havde tiltænkt, som deling af sin rejse, tilføjelse af rejsemakkere osv. blev desværre ikke til noget.

Jeg synes også at appens flow kunne optimeres. I stedet for at du vælger antal ønskede måltider, oplevelser og ønsket tidsrum på forskellige sider, kunne dette sagtens være gjort på en enkelt, samlet side.

## Muligheder for videreudvikling

Skulle jeg videreudvikle på projektet, ville brugerhåndtering helt klart være en førsteprioritet. Dette ville muliggøre deling af rejseplaner, tilføjelse af rejsemakkere og generelle sociale features jeg havde tiltænkt.

Appens flow i forhold til opsætning af præferencer og valg at tidsrum, rejsedage, spisesteder og oplevelser kunne også forbedres en hel del.

UI’et kunne også godt bruge en overhaling, så det ikke bare er Microsofts standard XAML-elementer der vises.

Jeg tænker også det kunne have gjort appen meget bedre, hvis man selv kunne vælge sin foretrukne navigationsservice, og at denne var integreret direkte i appen, så man ikke skulle forlade den for at gå til navigation.

Som appen står lige nu, kan brugeren heller ikke rette sine præferencer. Dette skal også gøres muligt.

# 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

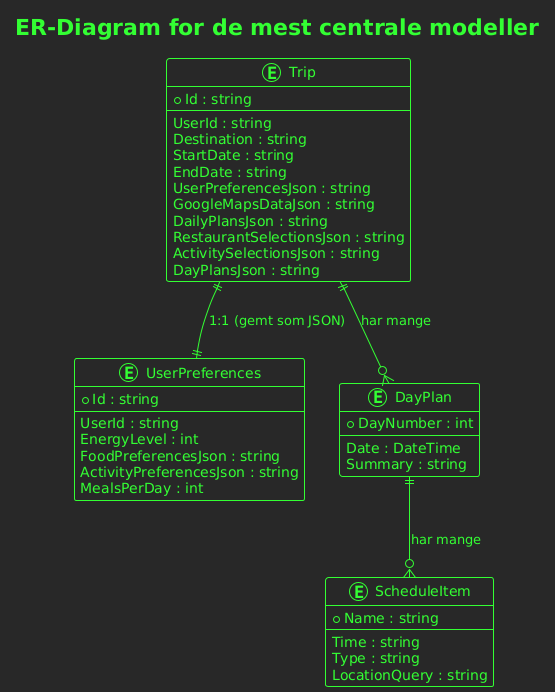
## Grafik

Michelle Wittstrøm Petersen – Mockups, app ikon og forsidebillede.

# Bilag

## Bilag 1 – Flowchart

## Bilag 2 – ER Diagram over de mest centrale modeller



## Bilag 3 – GenerateDayPlans-metoden.

