Charlie Tango x MMD

Frontend Case

Charlie Tango

Who are we?

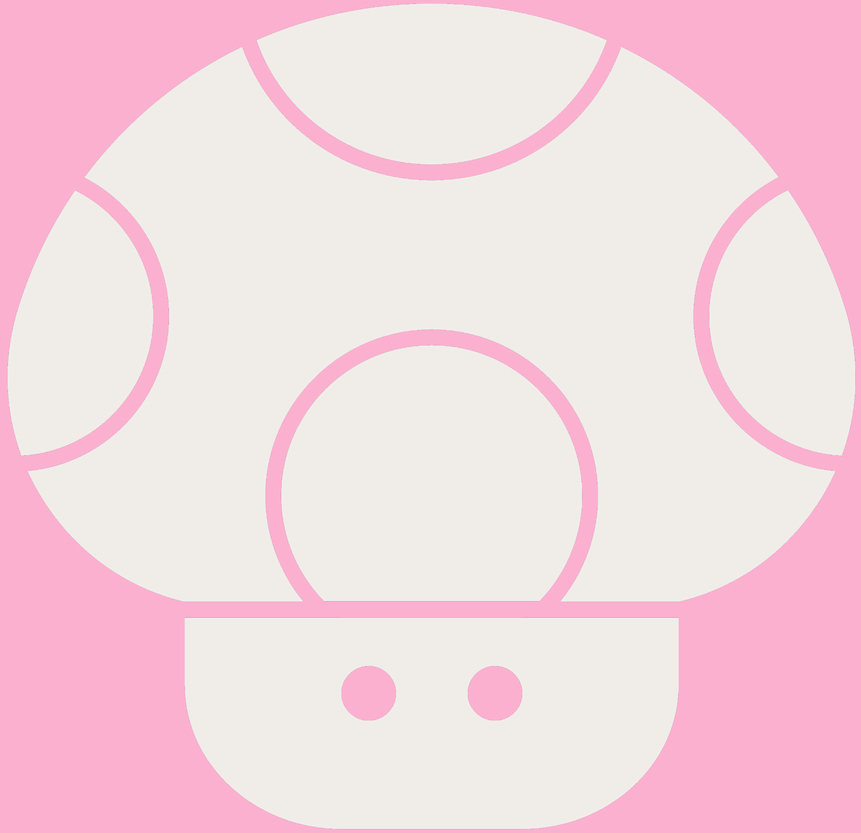
01


CHARLIE
TUPAC.

Who am I?

Daniel Schmidt Lead Engineer

- MMD in 2007
- Started as a Flash Developer ⚡
- Working at In2media/Charlie Tango since 2011
- “Switched” to HTML in 2014
- Frontend Team Lead in 2020
- Lead Engineer in 2023



Maria

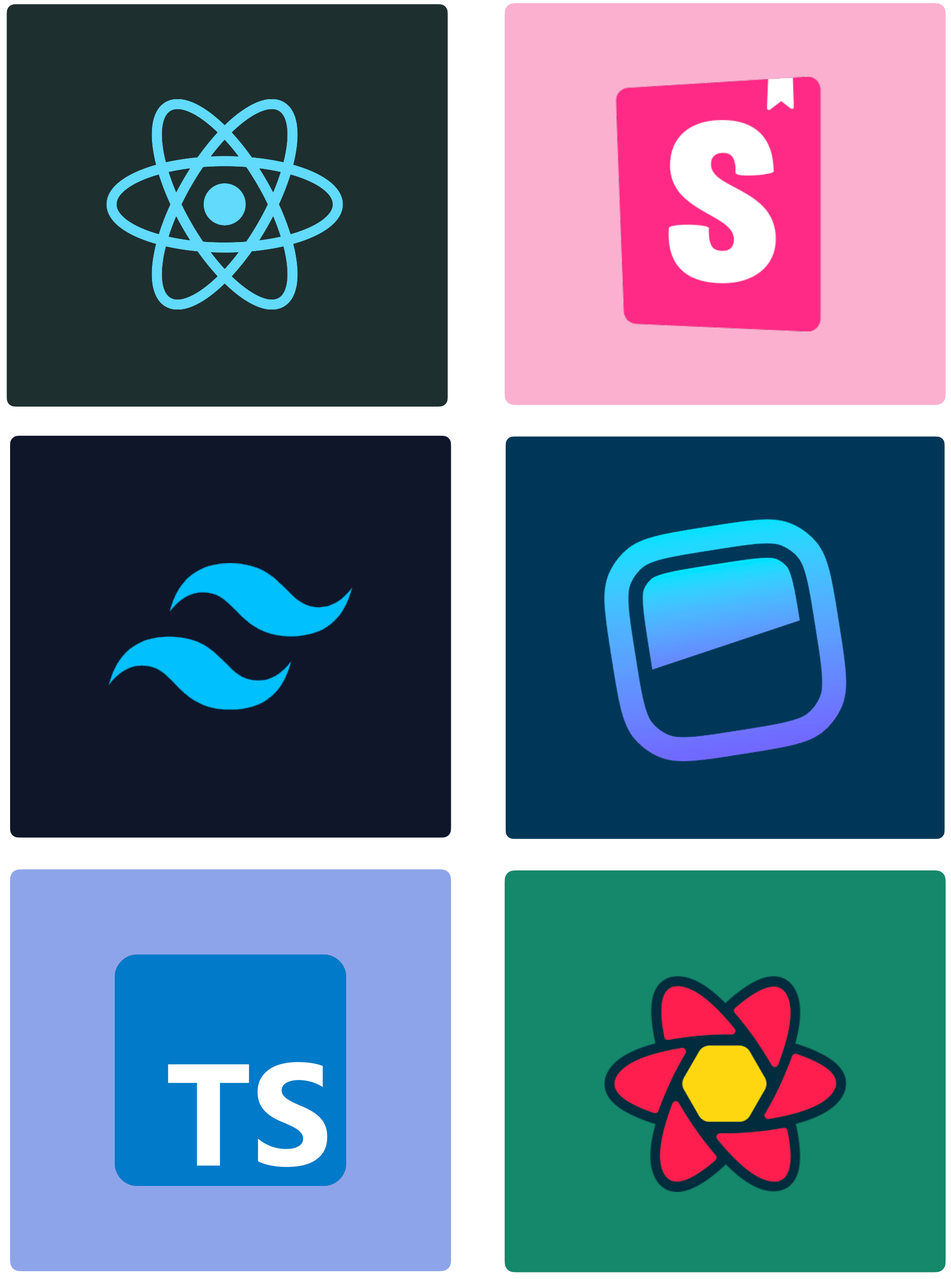
How I became a Frontender

Maria Lundqvist Frontend Developer✨

- Content Marketer at Sunweb Group
- Started MMD to become a fancy designer ✨
- Finishing MMD in 2023 🤞
- Frontend Intern at Charlie Tango in 2023

Now I know...

- Tailwind CSS
- React Query
- TypeScript
- Storybook
- React Hooks



Some CT Takeaways

ASK

Ask all your stupid questions.

PLAN

A good plan saves you **so much time.**

SHARE

Share your victories with each other.

What we do

Digital transformation

Key areas for the modern organisation

01

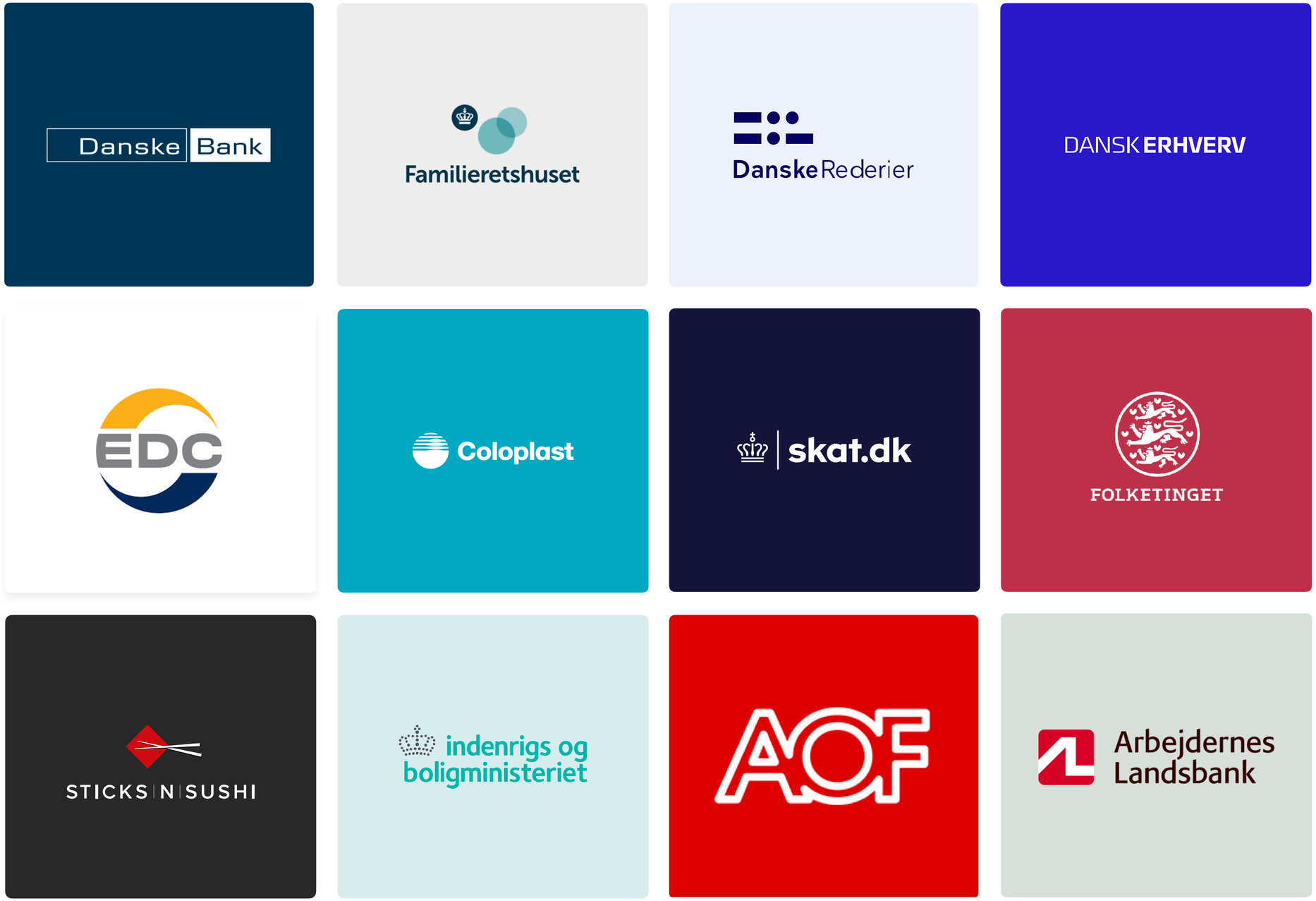
Experience
Design

02

Platform Strategy
& Implementation

03

Data-driven
decision-making



A selection of projects we have worked on recently

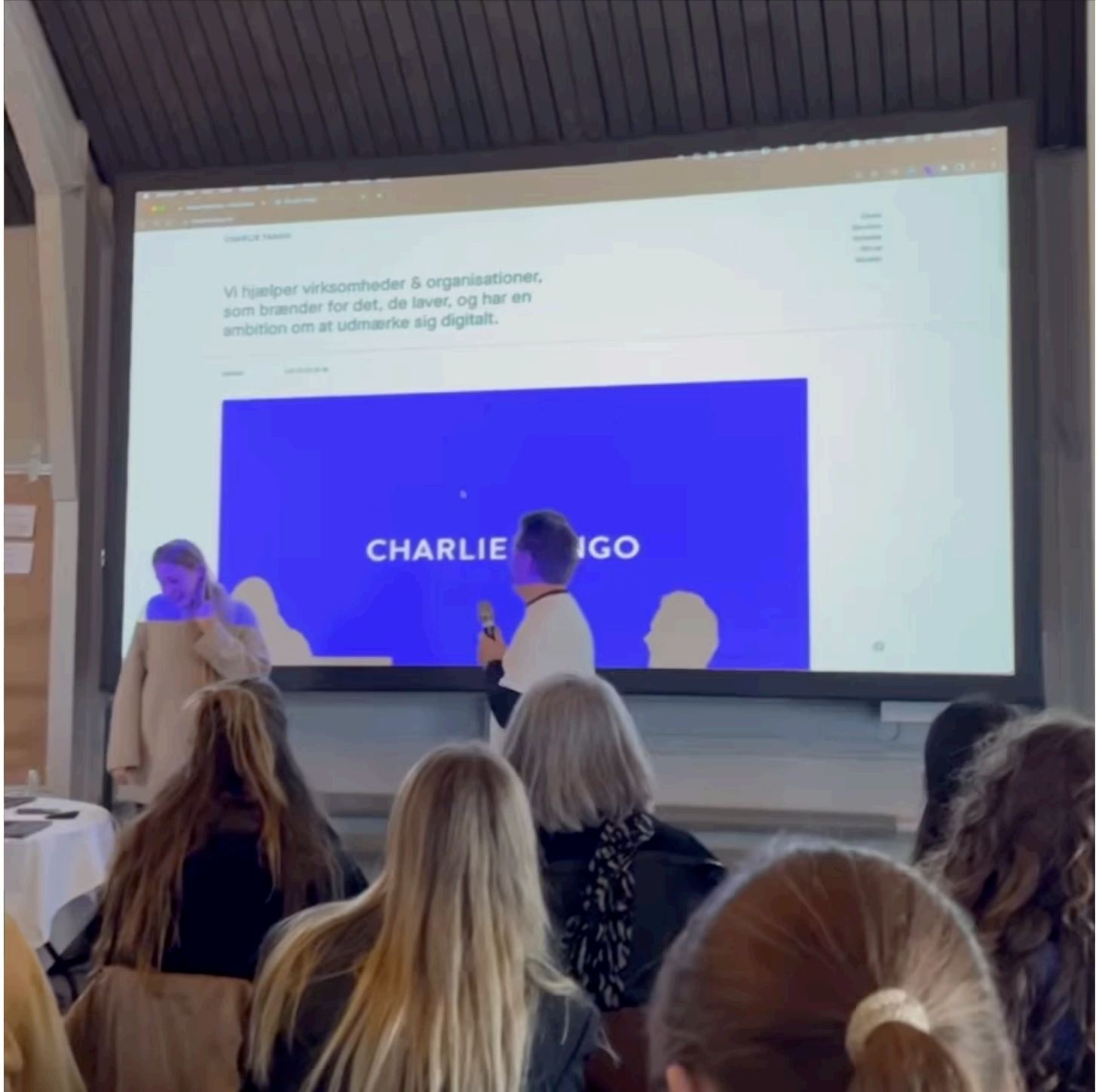
CHARLIE TANGO

New charlietango.dk - went live last week!

- Next.js
- Tailwind CSS
- DatoCMS
- Made by MMD interns 💪

Check it out!

(after the presentation 😄)



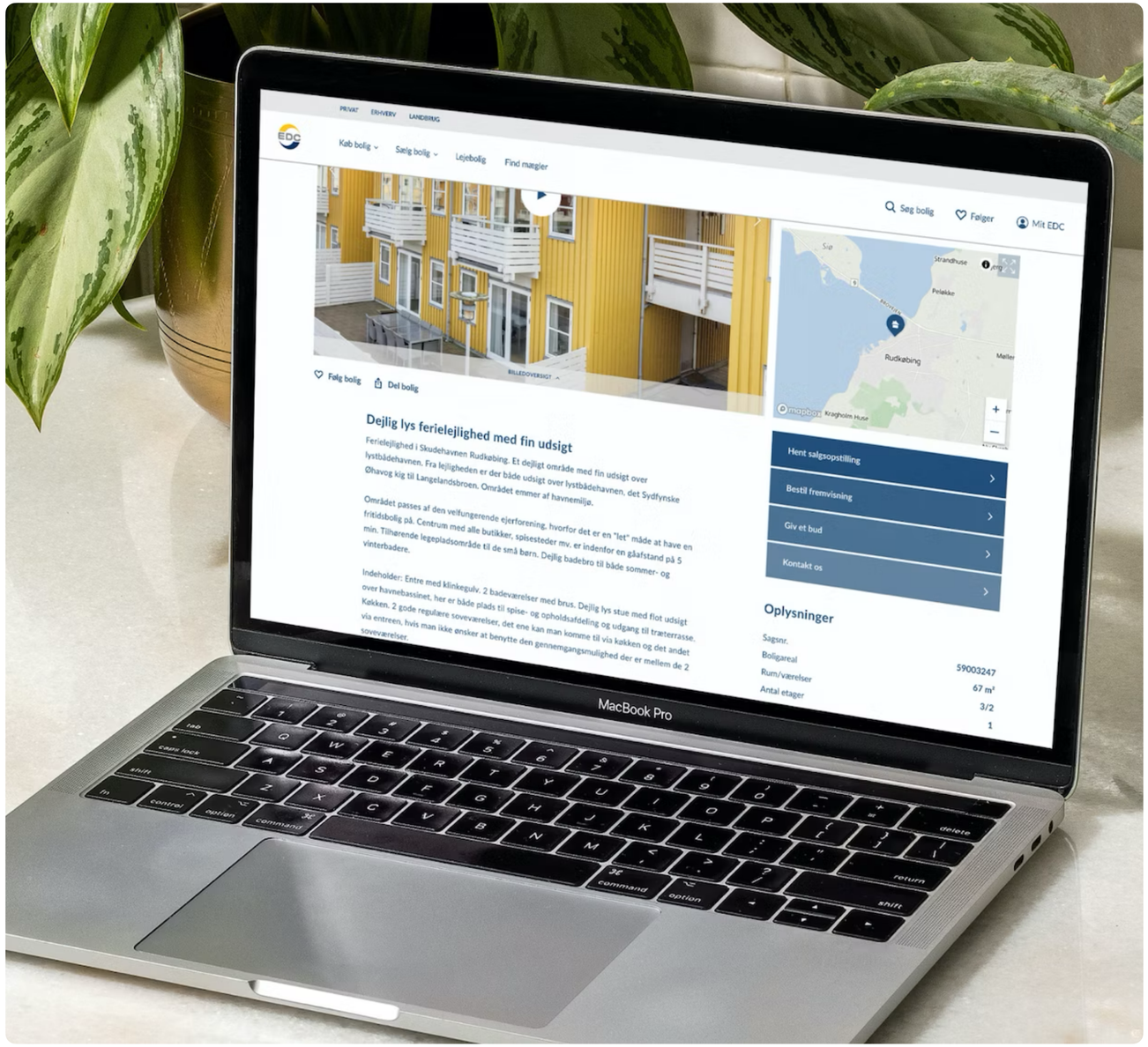
Enough about us

Your Client

Introducing...

02


EDC



EDC is Denmark’s largest real estate agency chain, consisting of independent real estate agents spread across 230 local stores.

PRIVAT ERHVERV LANDBRUG

EDC

Køb bolig Sælg bolig Lejebolig Find mægler

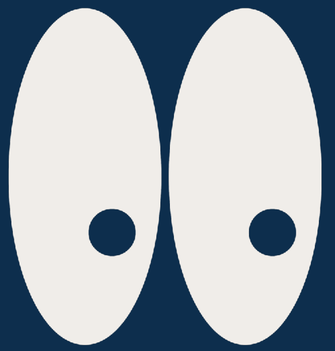
Søg bolig Følger Mit EDC

Hvor ønsker du at bo?

HVOR ØNSKER DU AT BO?

Tilføj by, adresse eller kommune

SØG BOLIGER



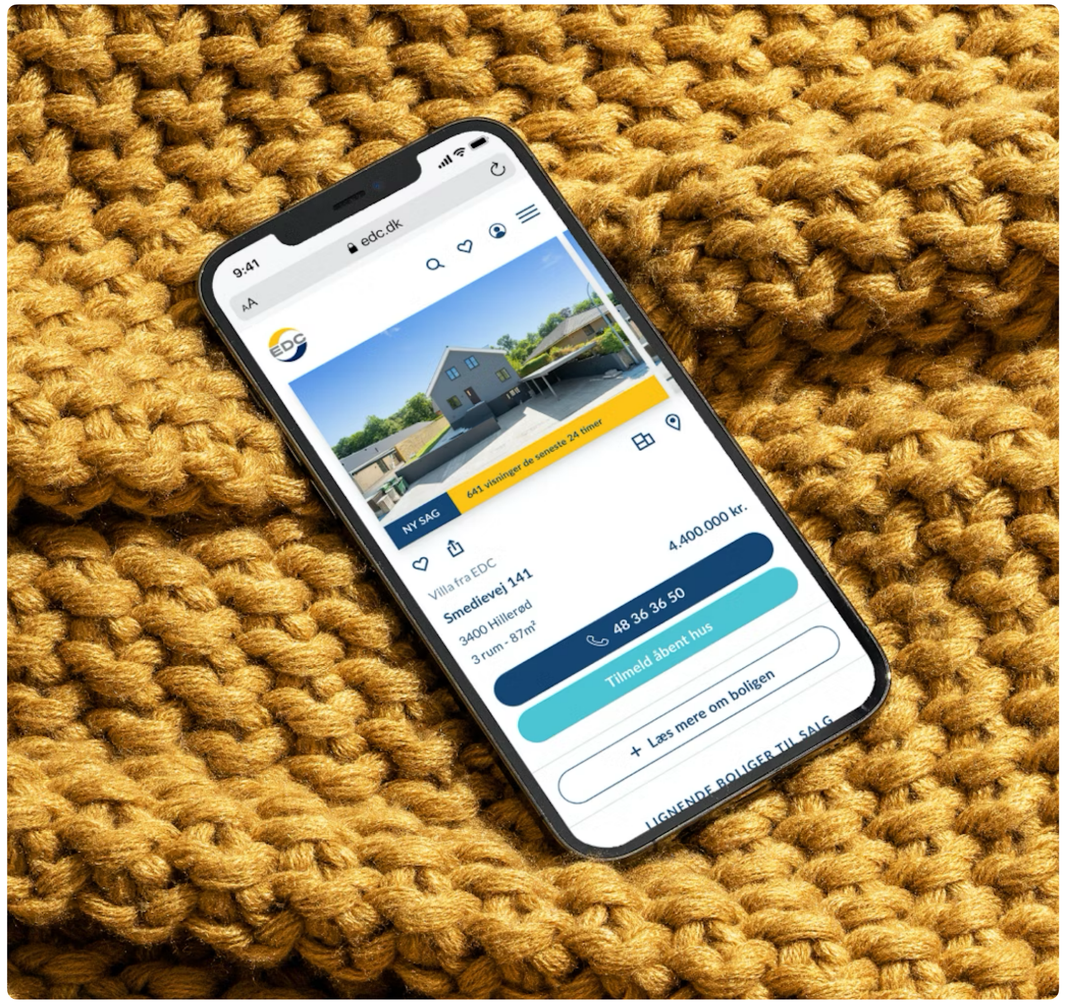
The case

Find Buyers site

The goal of this case is to help people who are thinking of selling their property, to get an idea of the market before they work with a real estate agent.

This will help them understand the value of their property and the potential buyers in their area before beginning the process or engaging with an agent.

If they find a potential buyer, EDC can start the process of contacting them and helping them sell their property.



What you'll be working with

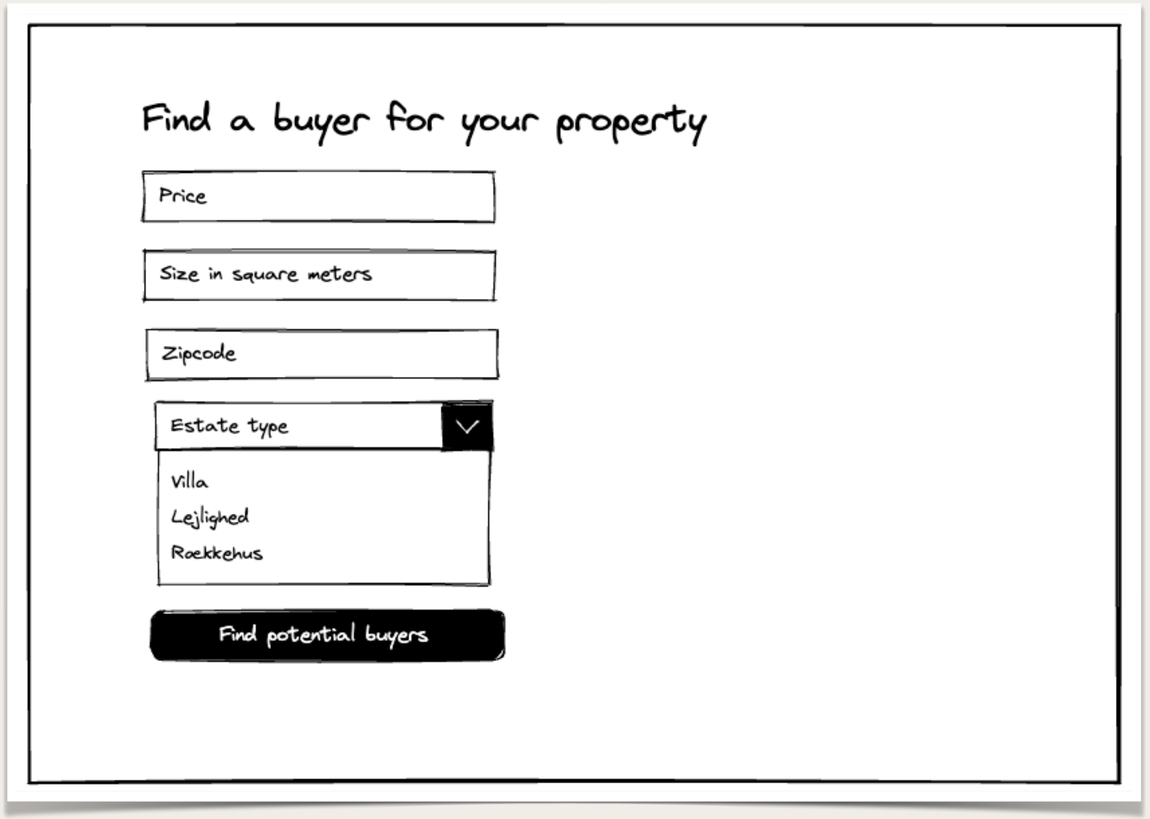
Focus areas

- Creating UX flows
- Building a Next.js application
- Forms
- Data fetching

Suggested flow

Screens

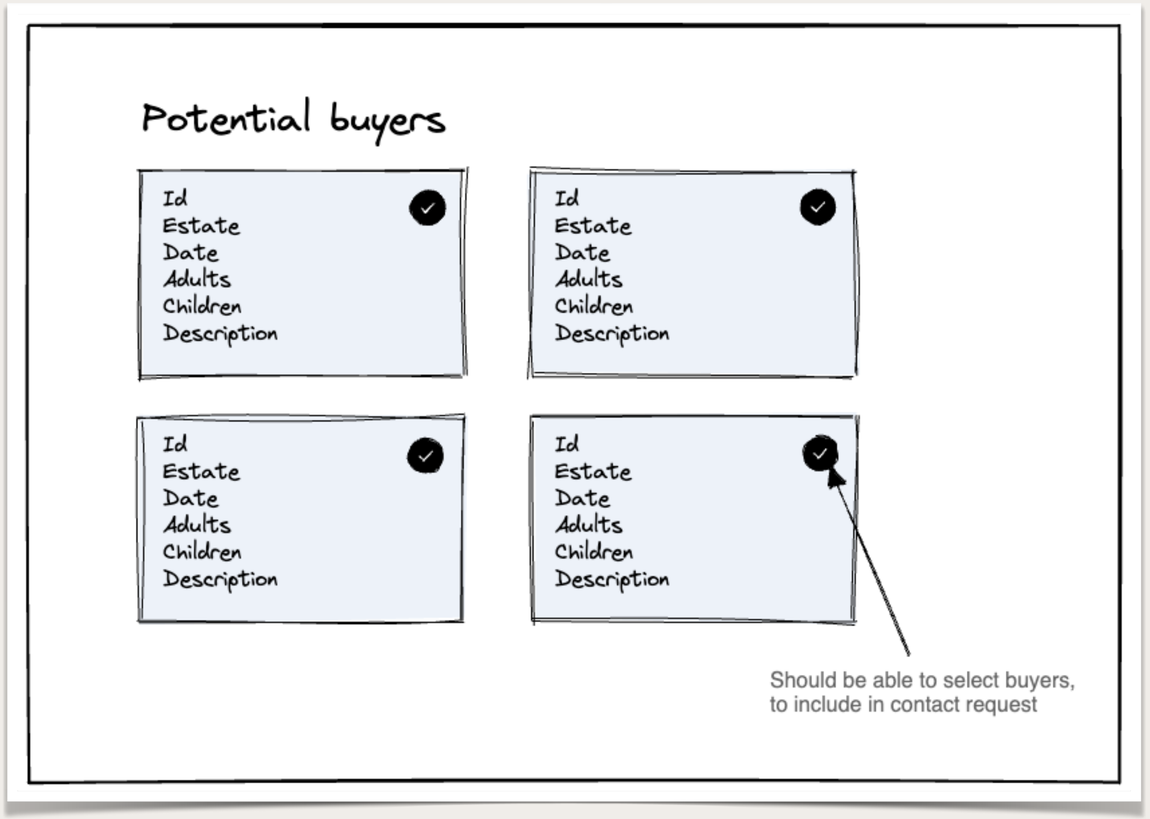
Step 1 - Property details



Requirements

- Prompt the user to enter a few details about the property they want to sell.
- The expected price (in Danish kroner)
- You'll need to know the area the property is located in, with either:
  - Zipcode
  - Full address
- Estate type is a fixed list of properties
- Submitting the form, should send the user to the next page.

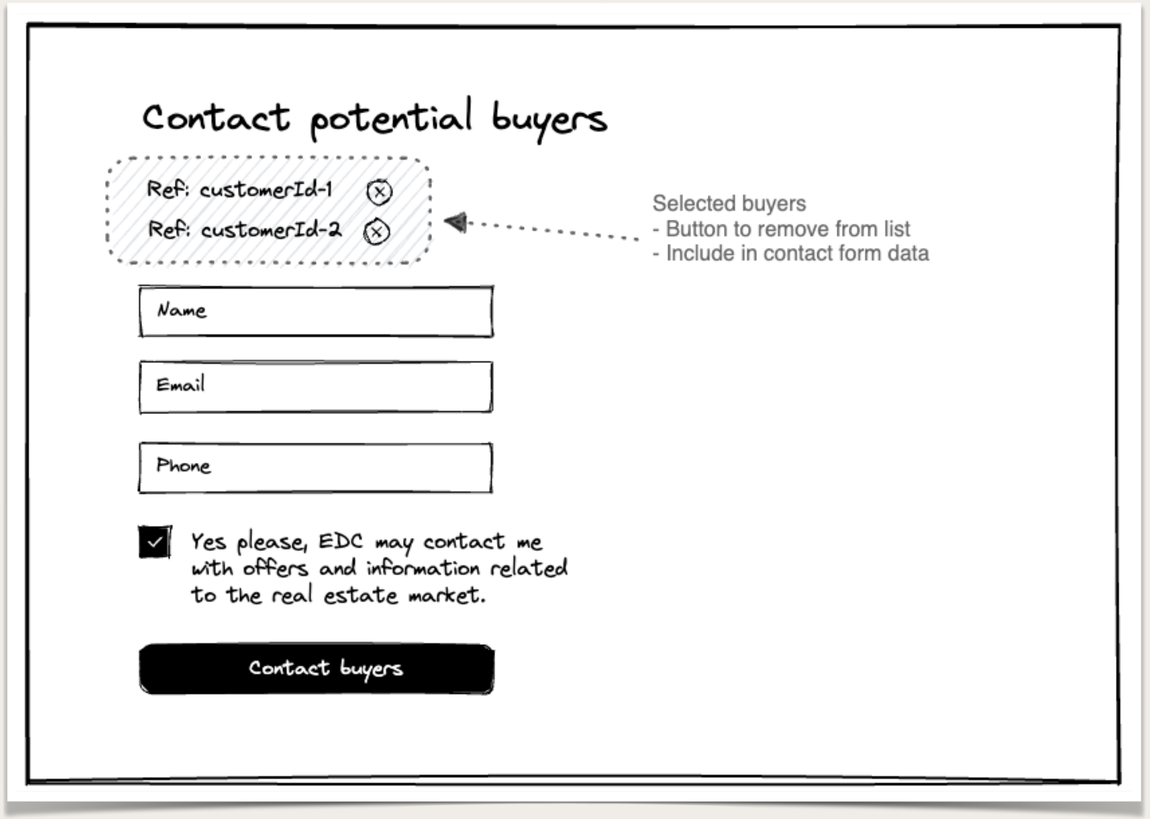
Step 2 - Results



Requirements

- Display a list of potential buyers, based on the zip code, and other details, entered by the user.
- Allow the user to select buyers they find interesting

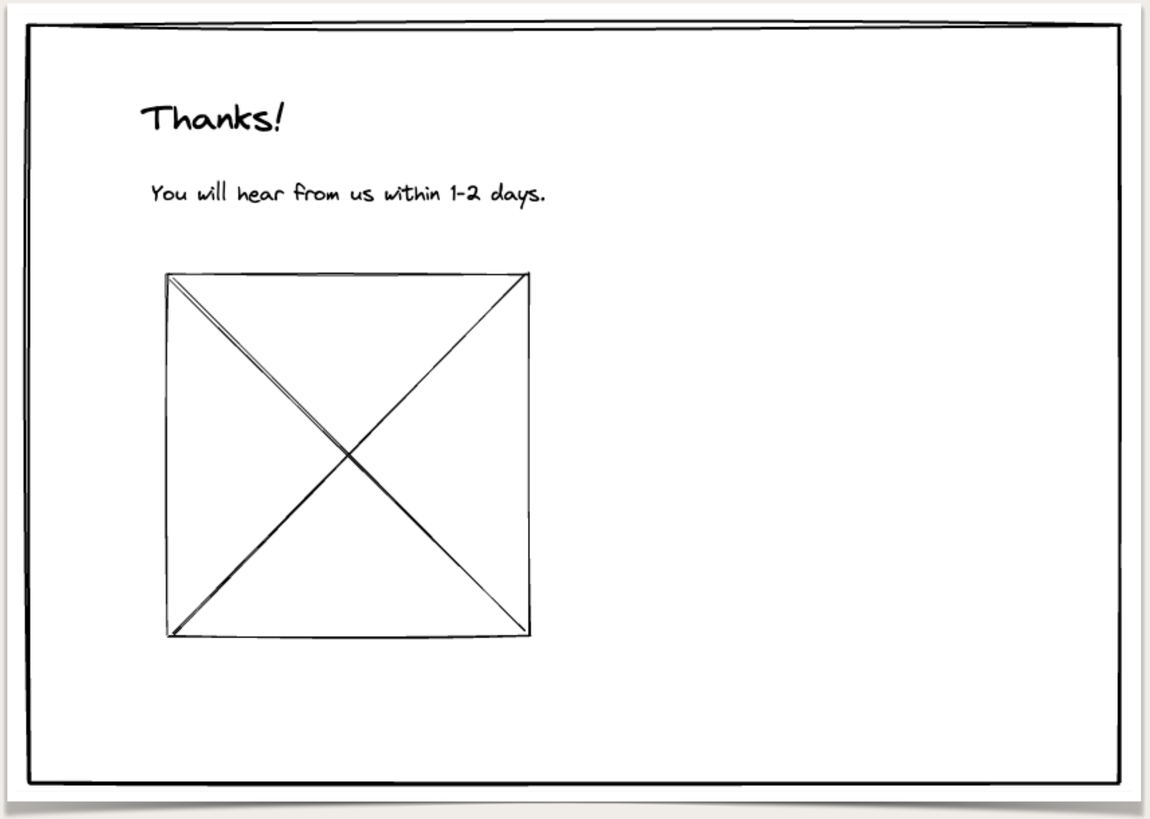
Step 3 - Contact



Requirements

- Allow the user to fill in their contact details, so EDC can contact the potential buyers on their behalf.
- Include the buyers the user selected on Step 2
- Submit the data to a **Supabase** database

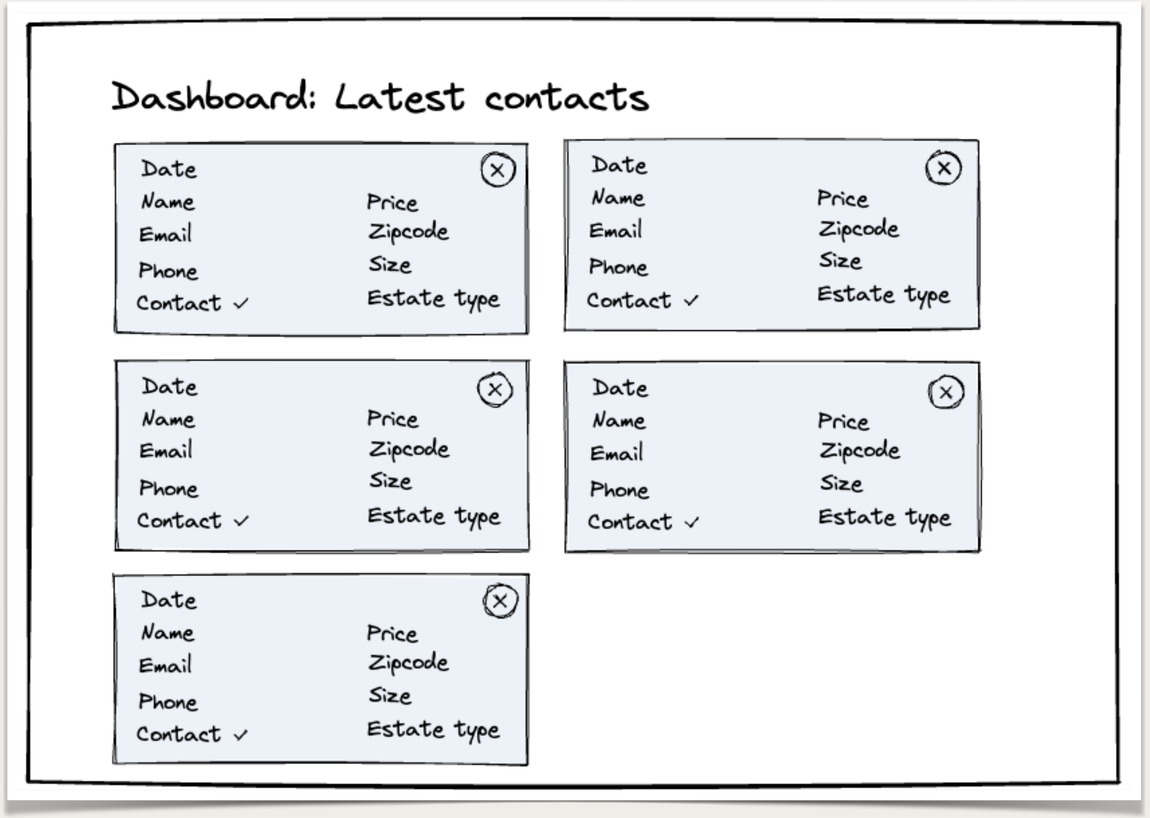
Step 4 - Confirmation



Requirements

- User should get confirmation, that their request has been received, and will be processed

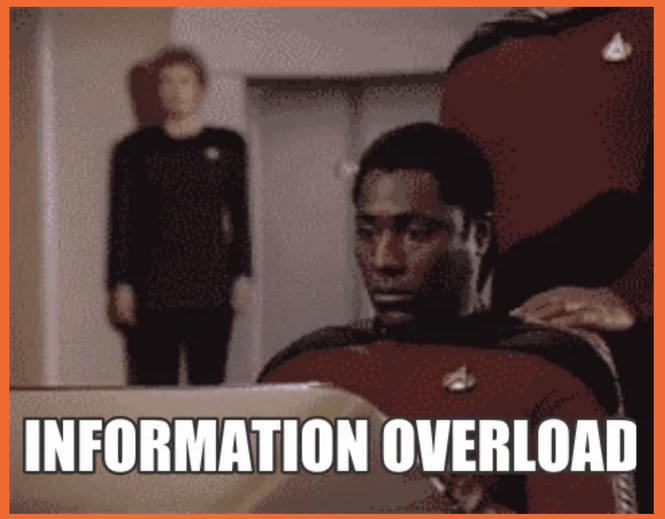
Admin dashboard



Requirements

- Display latest requests from the **Supabase** database
- It should be located at `/dashboard`
- Email and phone should be displayed as links, so the employee can click to call or email the user
- The list should be sorted by the date the request was created, with the most recent requests at the top
- You should be able to delete a request from the list, by clicking a delete button
- It doesn't need to be secured with login

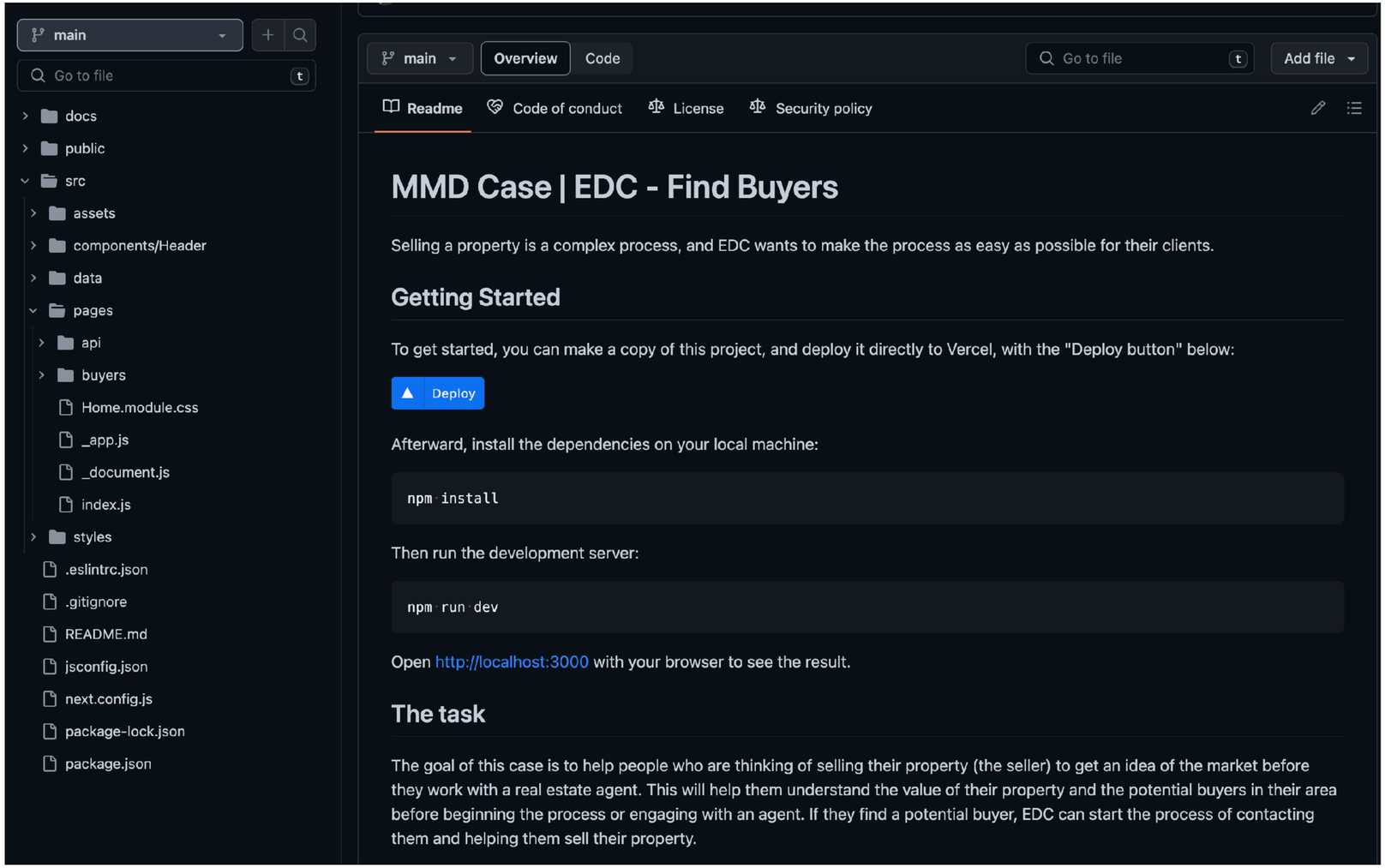
That’s it!



Next.js template

Hit the ground running

03



Template project

https://github.com/charlie-tango/mmd-case/

You're almost done.

Please follow the steps to configure your Project and deploy it.

mmd-case

- Create Git Repository
- Deploy

GIT REPOSITORY

charlie-tango/mmd-case

main

./

Import a different Git Repository →

Browse Templates →

Create Git Repository

To ensure you can easily update your project after deploying it, a Git repository must be created. Every push to that Git repository will be deployed automatically.

Git Scope: thebuilder

Repository Name: charlie-tango-case

Create private Git Repository

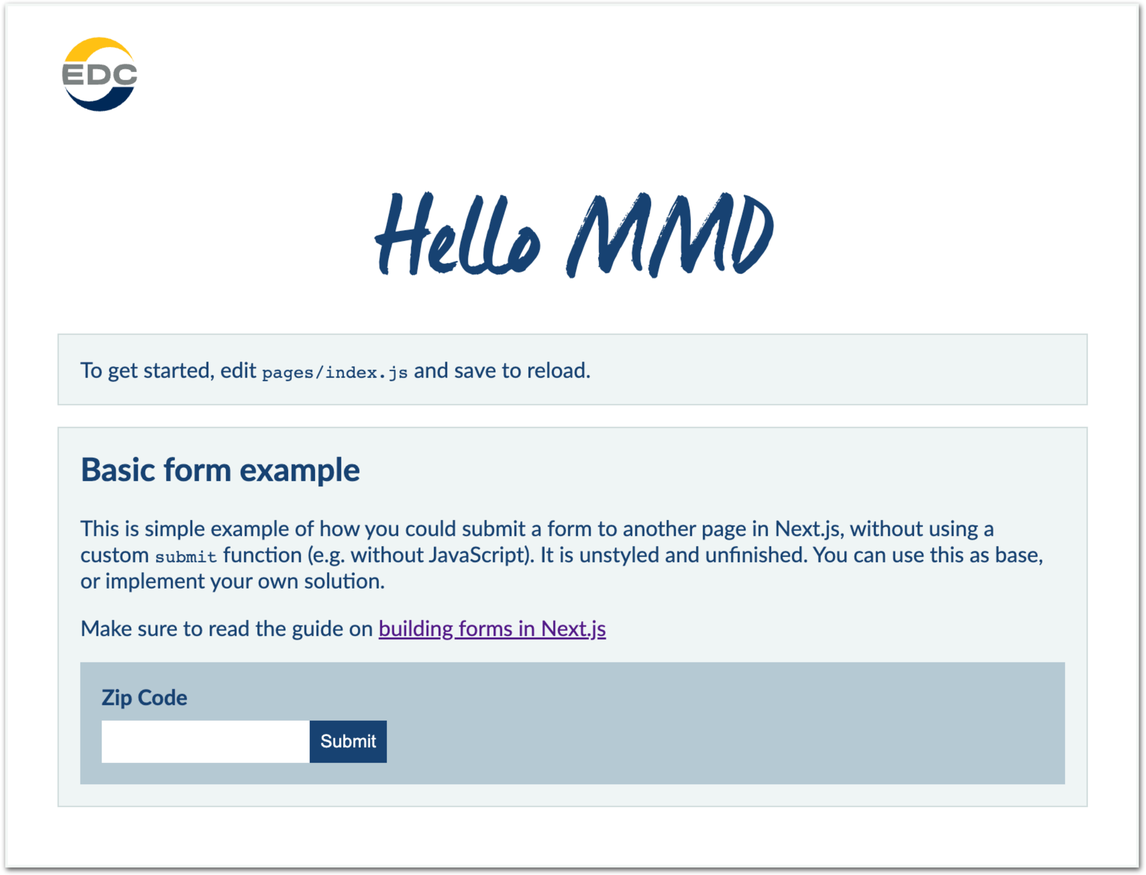
Create

Uncheck

Deploy

Preparing Git Repository.

Cloning charlie-tango/mmd-case


Hello MMD
To get started, edit pages/index.js and save to reload.
Basic form example
This is simple example of how you could submit a form to another page in Next.js, without using a custom submit function (e.g. without JavaScript). It is unstyled and unfinished. You can use this as base, or implement your own solution.
Make sure to read the guide on building forms in Next.js
Zip Code
Submit

Styling

```
/* CSS Variables
 * `./src/styles/global.css`
 */
:root {
  --dark-blue: #0d2e4d;
  --blue: #164573; /* Primary color; */
  --hover: #2985cc;
  --dusty-blue: #5e7c9a;
  --dusty-blue-light: #93b0cd;
  --dusty-blue-lighter: #bbc8d7;
  --aqua: #4dc4d4; /* Secondary color; */
  --dusty-aqua: #d2dfe0;
  --dusty-aqua-light: #f3f7f7;
  --grey: #e5e5e5;
  --dark-grey: #a2a2a2;
  --status-light-red: #e18c8c;
  --status-red: #f24040; /* Error color; */
  --yellow: #fec315; /* Accent color; */
  --white: #fff;
  --black: #000;

  /* Fonts */
  --font-sans: "Lato", sans-serif;
  --font-serif: "Manus", serif;
}
```

- Project configured to use CSS modules
- EDC colors and fonts have been added as CSS Custom Properties (variables)

SVG Icons

- A few EDC style icons you can use
- Located in `./src/assets/`

Forms

- Don’t worry too much about validation
  - Native HTML input validation is fine
- Make sure the forms are accessible
  - `<input>` + `<label>`
  - Keyboard
  - Submit button

DAWA Challenge

Postnummeropslag

Modtag postnummer med id.

```
GET https://api.dataforsyningen.dk/postnumre/{nr}
```

Parametre

NAVN	BESKRIVELSE
`nr`	Postnummer. 4 cifre.
`landpostnumre`	Hvis denne parameter er sat vil den returnerede geometri for postnumre være afgrænset af kyster.
`struktur`	Struktur parameteren angiver om der ønskes en flad eller en nestet svarstruktur. Default er nestet for JSON, og flad for GeoJSON.

Eksempler

Hent postnummer for København NV

```
https://api.dataforsyningen.dk/postnumre/2400
```


```
{
  "href": "https://api.dataforsyningen.dk/postnumre/2200",
  "nr": "2200",
  "navn": "København N",
  "stormodtageradresser": null,
  "bbox": [
    12.5248903,
    55.6816636,
    12.57220713,
    55.71214273
  ],
  "visueltcenter": [
    12.54914572,
    55.6940569
  ],
```


- **OPTIONAL** - If you're bored one night
- Do zipcode input validation with the DAWA API
- Present the matching city in the UI
- Free public API
  - https://dawadocs.dataforsyningen.dk/

Data fetching

- Read up on the different ways of getting data
  - `getStaticProps()`
  - `getServersideProps()`
  - **API route** that can be fetched from the client
- Consider which method is best for a given page
  - Does the data change?
  - Is it user/session specific?
  - Is a “mutation”? (POST that changes data)
- https://nextjs.org/docs/basic-features/data-fetching

```
export default function handler(req, res) {
  // Find the zip code from the query parameters, and use it to generate a list of (fake) buyer profiles.
  const zipCode = parseInt(req.query.zipCode || "2100");
  const profilesForZipCode = generateBuyerProfiles({
    zipCode,
  });

  // Set the cache headers, so that the response can be cached
  res.setHeader("Cache-Control", "s-maxage=86400, stale-while-revalidate");

  // Make sure to filter out profiles based on the other query parameters. e.g. minSize, maxPrice, etc.
  return res.status(200).json(profilesForZipCode);
}
```

Buyer profiles

- Anonymous data - No names etc.
- Most relevant information to determine a match
- `./src/data/buyerProfiles.js`
  - Random mockdata, generated by Faker
  - Seeded by Zipcode
- Example API endpoint: `/api/find-buyers`

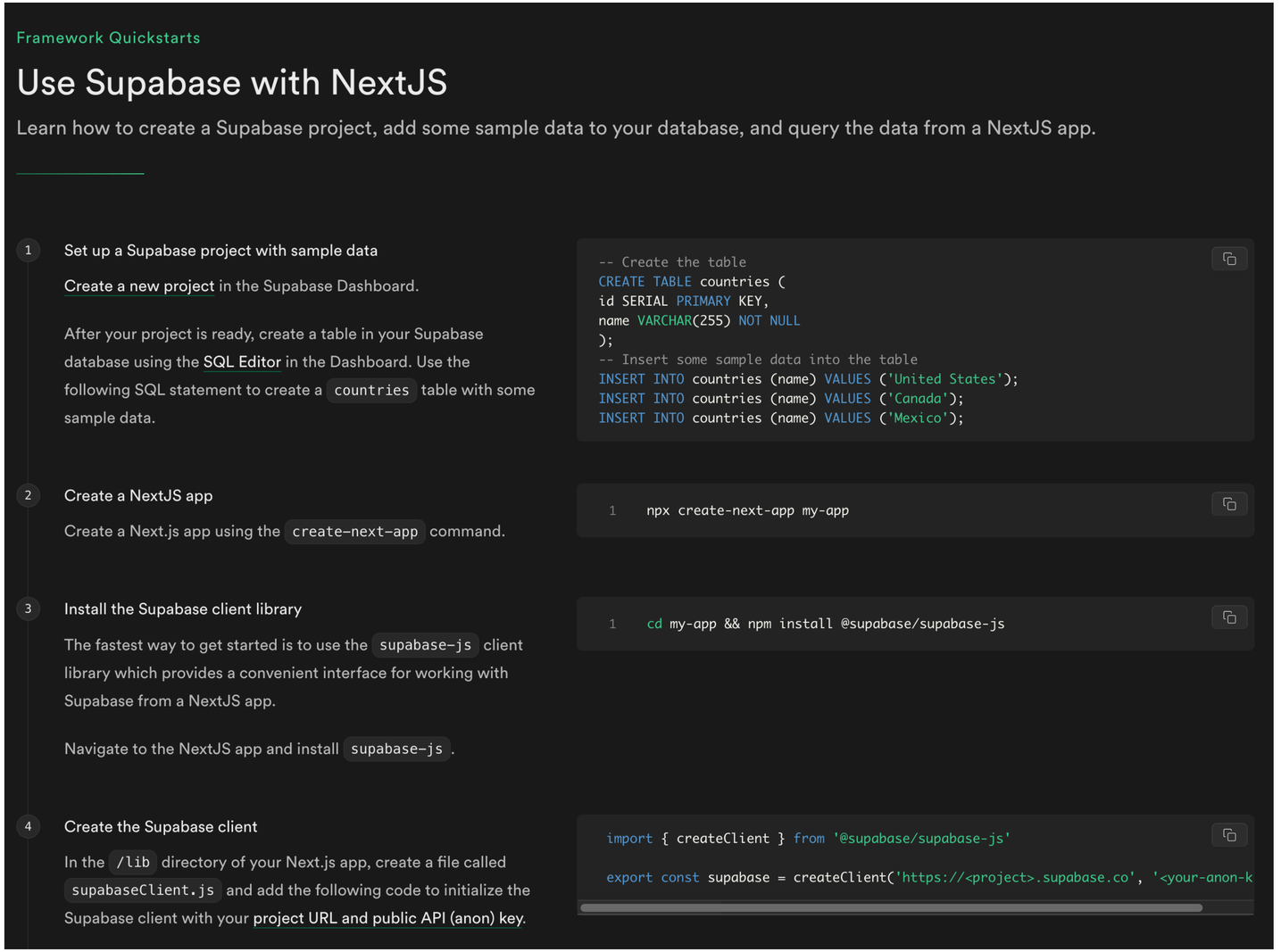

```
[
  {
    "id": "f0118597",
    "maxPrice": 6500000,
    "minSize": 74,
    "adults": 2,
    "children": 3,
    "description": "Family is looking for a Fritidshus with a minimum…”,
    "estateType": "5",
    "takeoverDate": "2023-07-05"
  },
  {
    "id": "a2a94863",
    "maxPrice": 7200000,
    "minSize": 104,
    "adults": 1,
    "children": 3,
    "description": "Single parent is looking for a Villa with a minimum…”,
    "estateType": "1",
    "takeoverDate": "2023-06-16"
  }
  …
]
```

Estate types

```
export const estateTypes = [
  {
    name: "Villa",
    id: "1",
  },
  {
    name: "Villalejlighed",
    id: "2",
  },
  {
    name: "Rækkehus",
    id: "3",
  },
  {
    name: "Ejerlejlighed",
    id: "4",
  },
  {
    name: "Fritidshus",
    id: "5",
  },
  {
    name: "Fritidsgrund",
    id: "6",
  },
  {
    name: "Helårsgrund",
    id: "7",
  },
  {
    name: "Andelsbolig",
    id: "8",
  },
  {
    name: "Landejendom",
    id: "9",
  },
];
```

- Different types of housing
- `./src/data/estateTypes.js`
- Make sure you use the `id` when submitting

Supabase Database



Framework Quickstarts

Use Supabase with NextJS

Learn how to create a Supabase project, add some sample data to your database, and query the data from a NextJS app.

1 Set up a Supabase project with sample data

Create a new project in the Supabase Dashboard.

After your project is ready, create a table in your Supabase database using the SQL Editor in the Dashboard. Use the following SQL statement to create a `countries` table with some sample data.

```
-- Create the table
CREATE TABLE countries (
id SERIAL PRIMARY KEY,
name VARCHAR(255) NOT NULL
);
-- Insert some sample data into the table
INSERT INTO countries (name) VALUES ('United States');
INSERT INTO countries (name) VALUES ('Canada');
INSERT INTO countries (name) VALUES ('Mexico');
```

2 Create a NextJS app

Create a Next.js app using the `create-next-app` command.

```
npx create-next-app my-app
```

3 Install the Supabase client library

The fastest way to get started is to use the `supabase-js` client library which provides a convenient interface for working with Supabase from a NextJS app.

Navigate to the NextJS app and install `supabase-js`.

```
cd my-app && npm install @supabase/supabase-js
```

4 Create the Supabase client

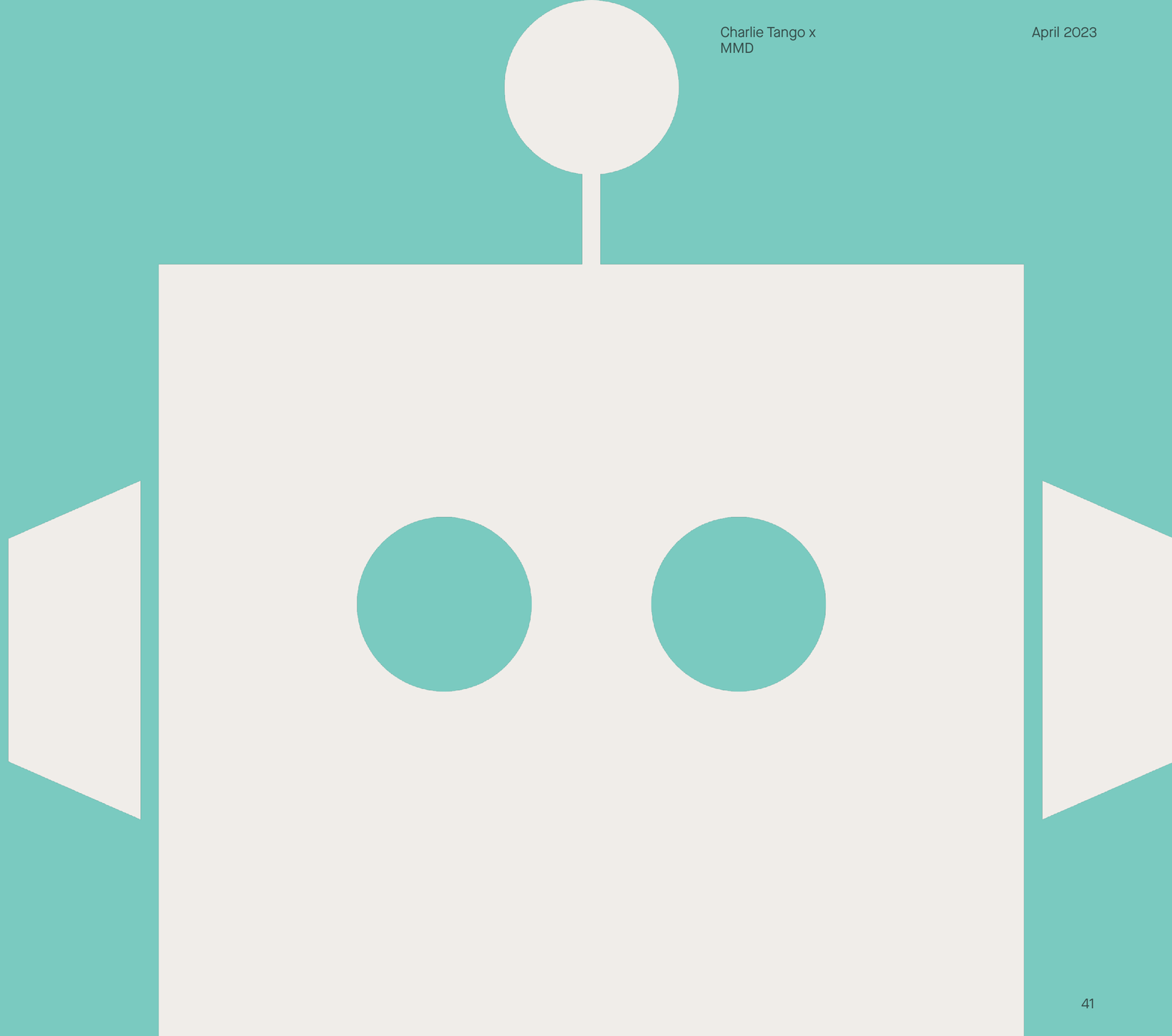
In the `/lib` directory of your Next.js app, create a file called `supabaseClient.js` and add the following code to initialize the Supabase client with your project URL and public API (anon) key.

```
import { createClient } from '@supabase/supabase-js'

export const supabase = createClient('https://<project>.supabase.co', '<your-anon-k
```

- Fetch and write to database using server methods
  - **API route** or **getServerSideProps**
  - Keeps your database safe and secure

Questions?




YOU GOT
THIS

Good luck!

https://github.com/charlie-tango/mmd-case/

Daniel Schmidt
Lead Engineer
dsc@charlietango.dk

charlietango.dk
70 22 15 16

Find us on
Linkedin and Instagram

CHARLIE TANGO