



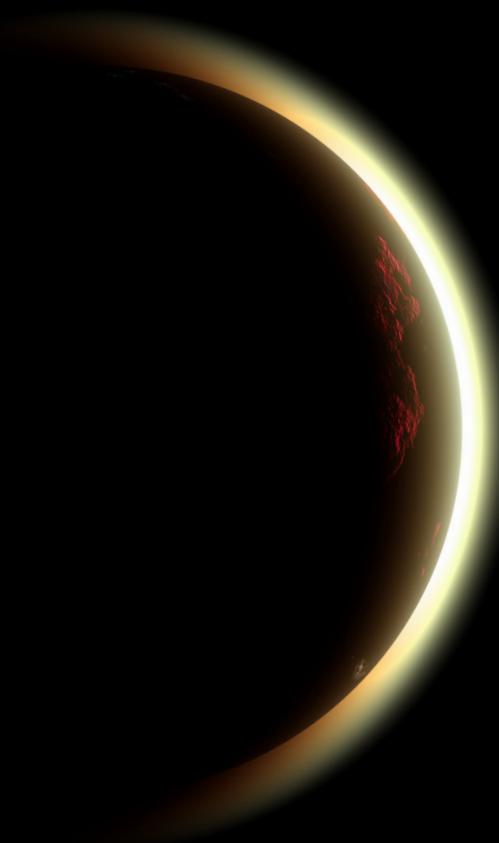
OUR HOME PLANET AT YOUR FINGERTIPS

by Orbital Penguins



TODAY'S AGENDA

- Challenges 01
- Purpose 02
- Roadmap 03
- Data and Pipeline 04
- Backend 05
- Frontend 06
- Result 07
- Future Features 08



CHALLENGES

“Develop a user-friendly application or tool to discover, visualize, and analyze NASA Earth data for monitoring our home planet.”

How can we use the massive amount of science data to visualize and adapt it to the user's needs and interests? There is so much information, but unprocessed there is no chance that an untrained individual can make use of it. So we process the information, provide different data sets and make it accessible for everyone with an intuitive Web app that can be used as an app or website. The software assists users in finding Earth science data that is important to their needs and also to visualize, query, store and download the datasets of your choice from whatever data is available using NASA APIs and to make it more interactive for users of all age range. This can also be great for teachers to show their students information more descriptive, or save the settings for later use.

PURPOSE OF THE PROJECT



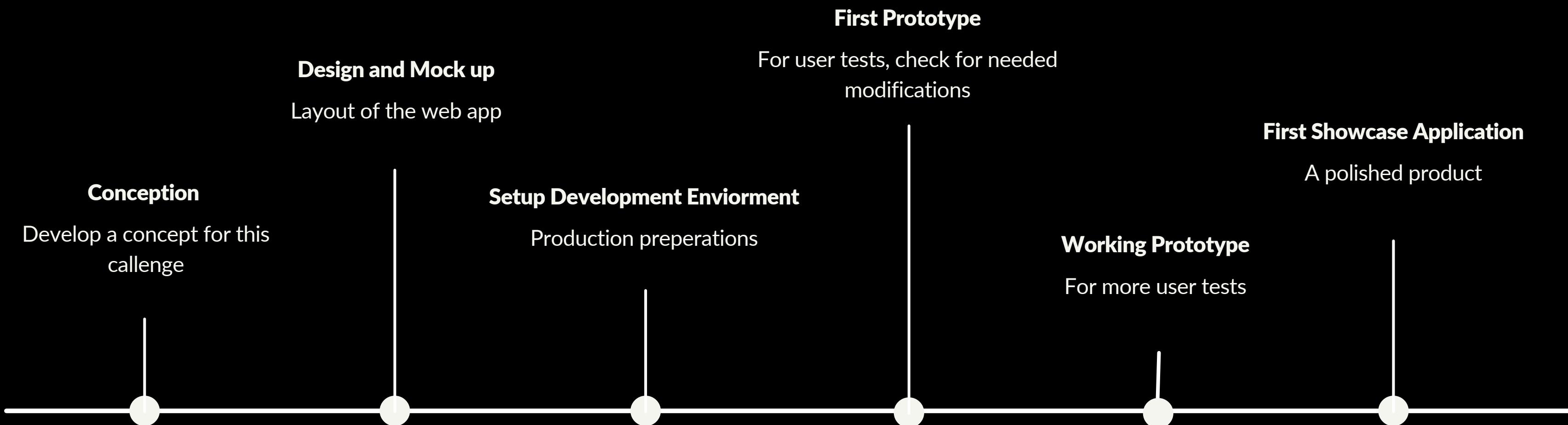
Inspire people to explore the world using data-science

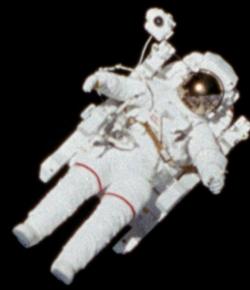
Strengthen awareness of climate crisis

self-explanatory tool for everyone on this beautiful planet

ROADMAP

A brief overview of our Milestones





DIVISION OF WORK

Benjamin

Data Pipelining and Data
Interfacing

Daniela

Data Preparation,
Calculation and Filtering

Elias

Head of Development

Fabian

Backend Operations and
Backend Development

Lena

Frontend Development

Tamara

Design Conception and
Project Management



DATA AND PIPELINE

POST ▾ http://localhost:3000/categories

Send

Query Auth Headers² Body¹ Tests

Text Json XML Form Form-encoded Graphql

Json Content

```
1 {  
2   "categorie": "co2",  
3   "date": "2020-03-13"  
4 }
```

Format

Status: 200 OK Size: 13.66 MB Time: 407 ms

Response Headers⁷ Cookies Test Results

```
1 ↴ [  
2   "date": "2020-03-13",  
3   "data": [  
4     {  
5       "value": 14523100000000000000,  
6       "lng": 153.868,  
7       "lat": -4.688  
8     },  
9     {  
10       "value": 18651900000000000000,  
11       "lng": 154.395,  
12       "lat": -4.34952  
13     },  
14     {  
15       "value": 14375100000000000000,  
16       "lng": 154.426,  
17       "lat": -4.13478  
18     },  
19     {  
20       "value": 13249100000000000000,  
21       "lng": 154.775,  
22       "lat": -4.86558  
23     },  
24     {  
25       "value": 14965700000000000000,  
26       "lng": 154.865,  
27       "lat": -4.23753  
28     },  
29     {  
30       "value": 11037100000000000000,  
31       "lng": 155.195,
```

NASA collects data

Provides data via FTP endpoints

Created ftp client

Downloaded geolocation data sets

Transformed data to fit to the backend

BACKEND

The screenshot shows a Visual Studio Code interface with a dark theme. On the left, there's a sidebar with file navigation and a 'New Request' button. The main area displays a POST request to 'localhost:3000/categories'. The response status is '200 OK', size is '160 Bytes', and time is '49 ms'. The response body is a JSON array of objects, each containing a 'category' field ('co2') and a 'dates' field which is an array of dates from '2019-12-20' to '2019-12-29'. Below the request and response, there's a terminal window showing a Node.js application starting up and catching an unhandled promise rejection.

```
at /home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/index.js:21:15
at Array.forEach (<anonymous>)
at /home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/index.js:18:27
at Layer.handle [as handle request] (/home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/node_modules/express/lib/router/layer.js:95:5)
at next (/home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/node_modules/express/lib/router/route.js:137:13)
at Route.dispatch (/home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/node_modules/express/lib/router/route.js:112:3)
at Layer.handle [as handle _request] (/home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/node_modules/express/lib/router/layer.js:95:5)
at /home/benjo/ausbildung/fhsalzburg/6.Semester/deepData/orbital-penguin/backend/node_modules/express/lib/router/index.js:281:22
(node:335781) UnhandledPromiseRejectionWarning: Unhandled promise rejection. This error originated either by throwing inside of an async function without a catch block, or by rejecting a promise which was not handled with .catch(). To terminate the node process on unhandled promise rejection, use the CLI flag '--unhandled-rejections=strict' (see https://nodejs.org/api/cli.html#cli_unhandled_rejections_mode). (rejection id: 2)
^C
+ backend git:(master) ✘ node index.js
Backend listening to port 3000.
```

Requests

Request that shows all available datasets with their available days

Request that loads all data for a specific day from the server

BACKEND

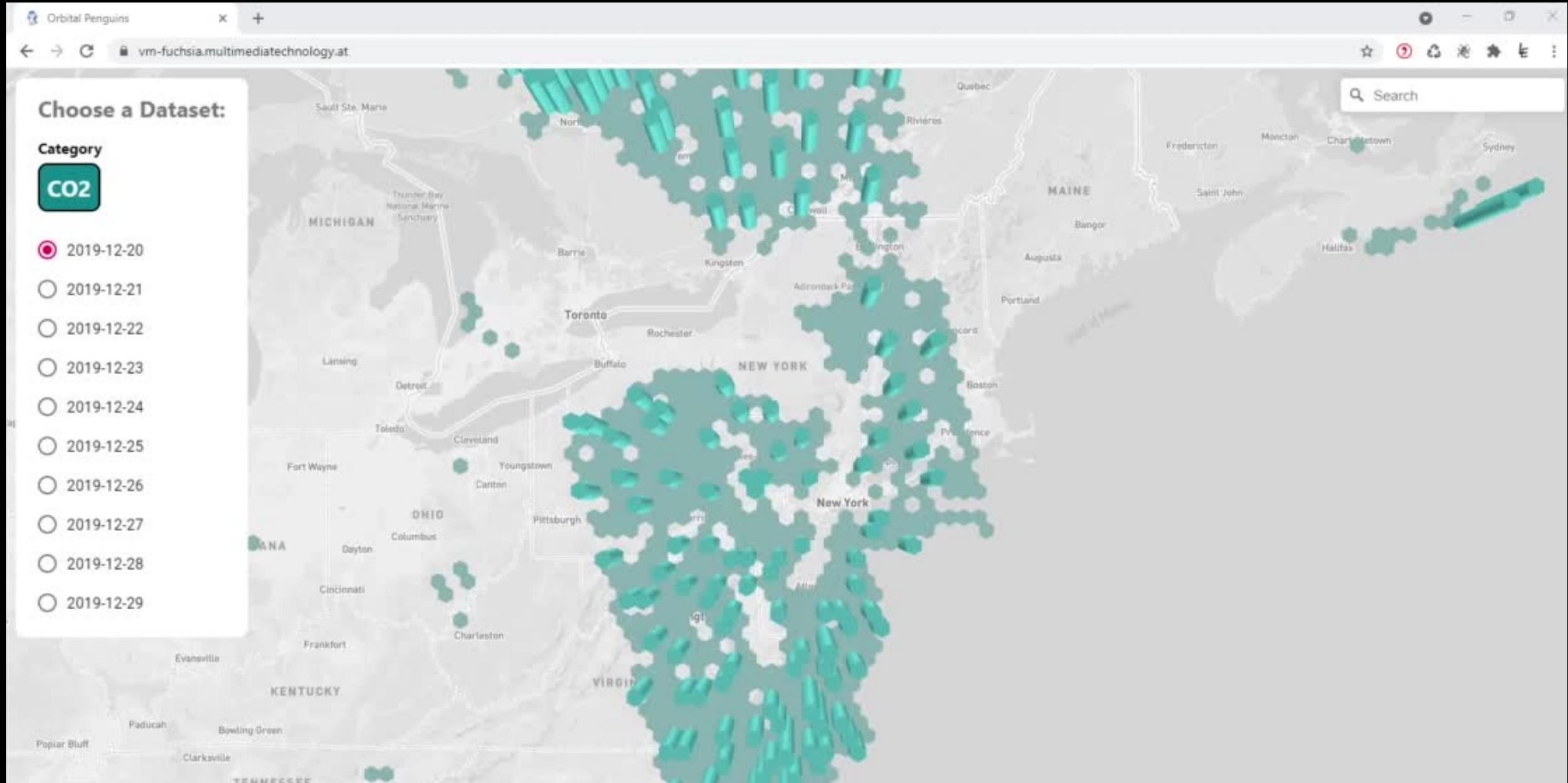
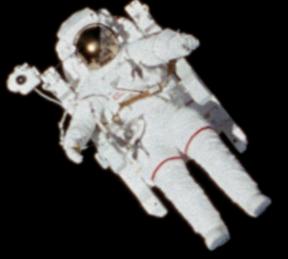
```
~ datasets
  ~ co2
    {} 2019-12-20.json
    {} 2019-12-21.json
    {} 2019-12-22.json
    {} 2019-12-23.json
    {} 2019-12-24.json
    {} 2019-12-25.json
    {} 2019-12-26.json
    {} 2019-12-27.json
    {} 2019-12-28.json
    {} 2019-12-29.json
  > forestFire
  > SurfaceTemp
```

Modular insertion of new datasets

A dataset is a directory on the backend server

Each day is saved as a file

FRONTEND



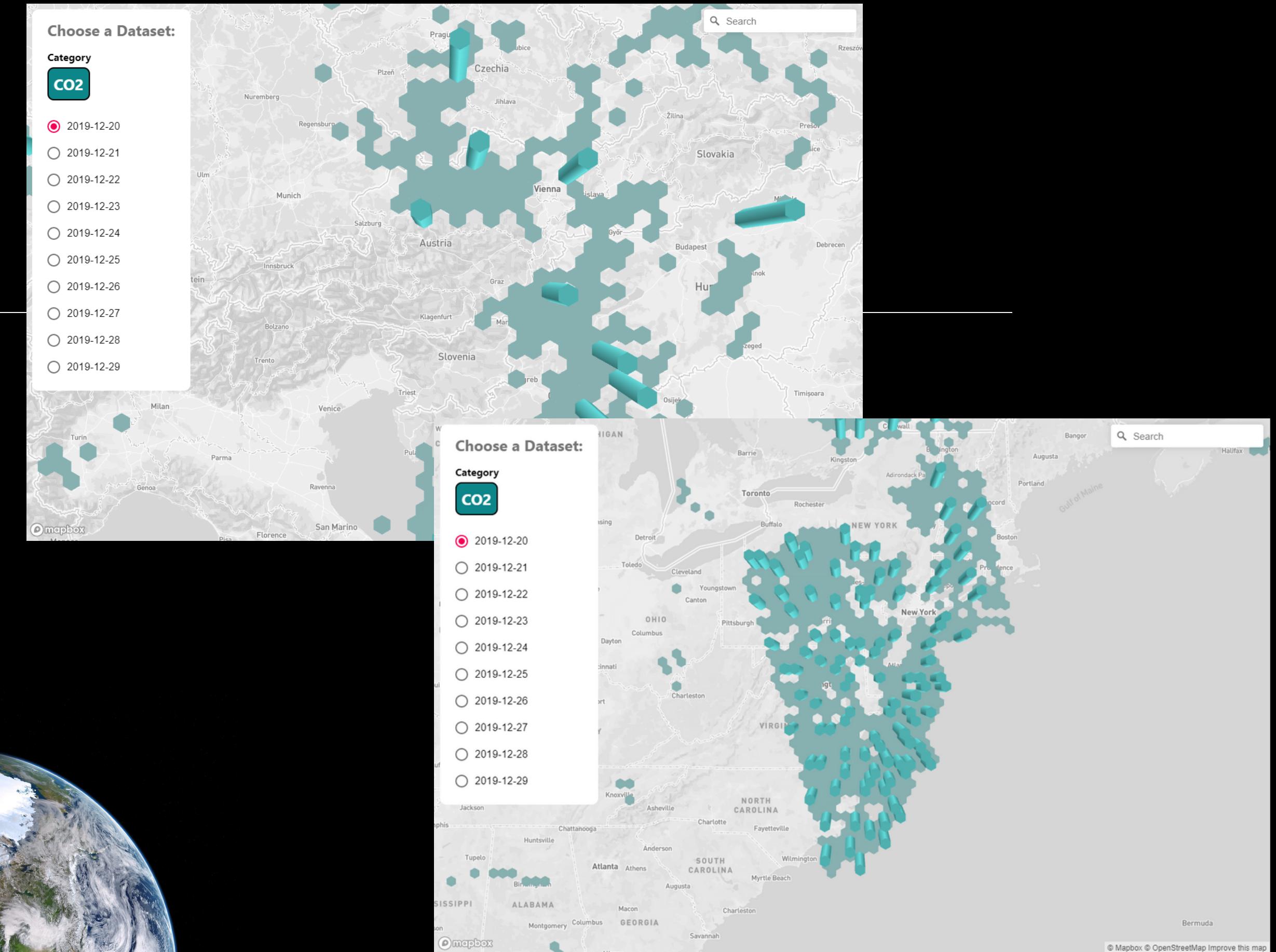
Data gets mapped according its coordinates

and visualised in a component-based Redux-React application

build upon Deck.gl and Mapbox

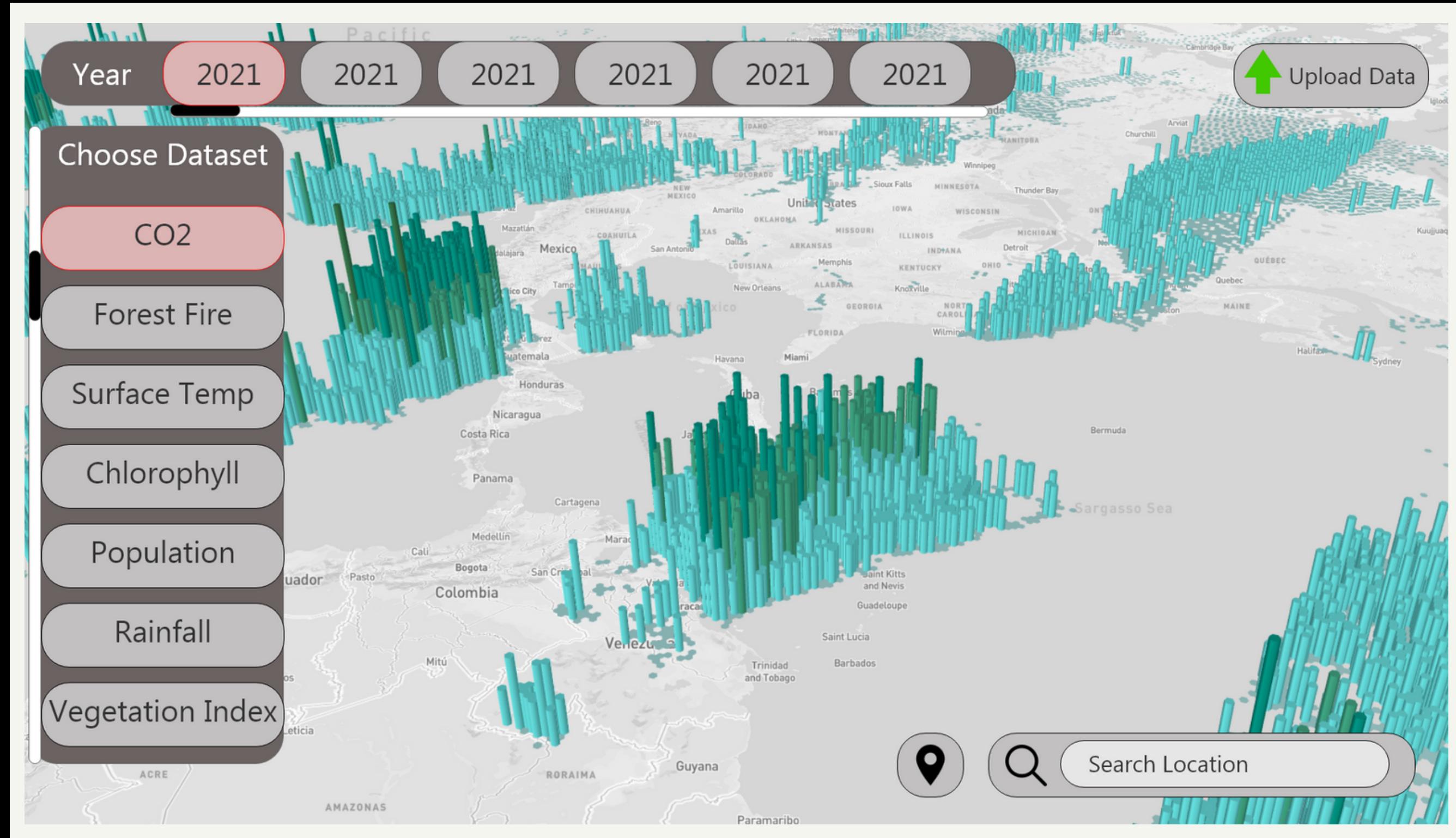


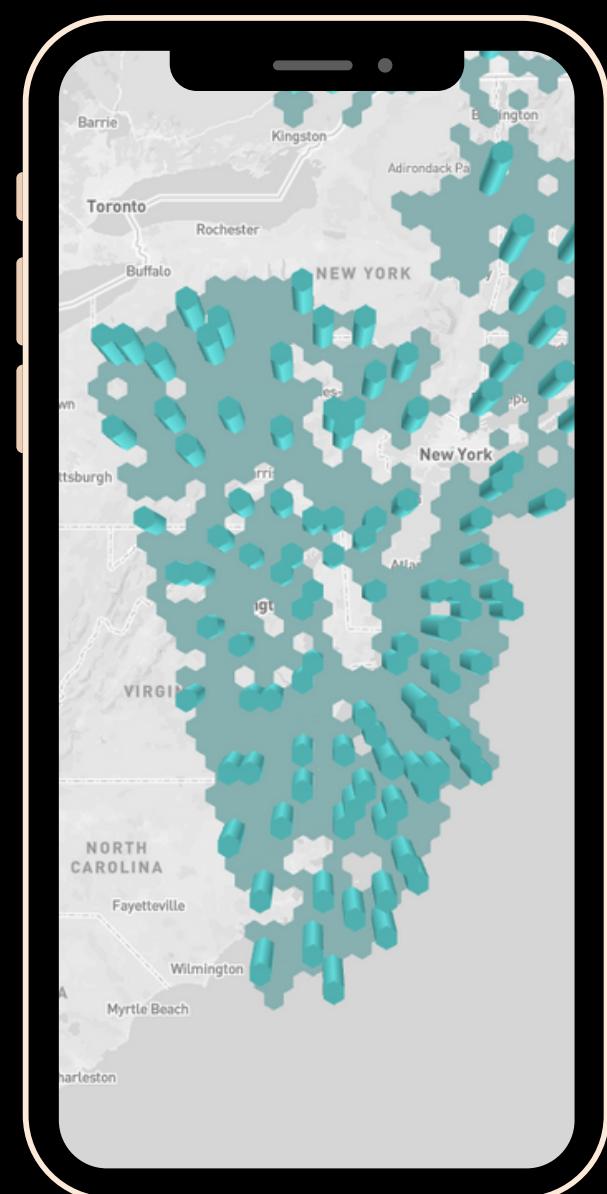
OUR RESULT



<https://vm-fuchsia.multimediatechnology.at/>

FUTURE FEATURES





— THANK YOU —

*Benjamin Halilovic, Daniela Wollmarker, Elias
Bader, Fabian Bliem, Lena Ebner, Tamara Moss*

SOURCES

Git Repository:

<https://github.com/development-with-benjamin/orbital-penguin>

CO2 Data:

ftp://asc-csa.gc.ca/users/OpenData_DonneesOuvertes/pub/

Technologies:

<https://reactjs.org/>

<https://deck.gl/>

<https://www.mapbox.com/>

<https://expressjs.com/de/>

Over and Out now!
THANK YOU for being part of our mission!

