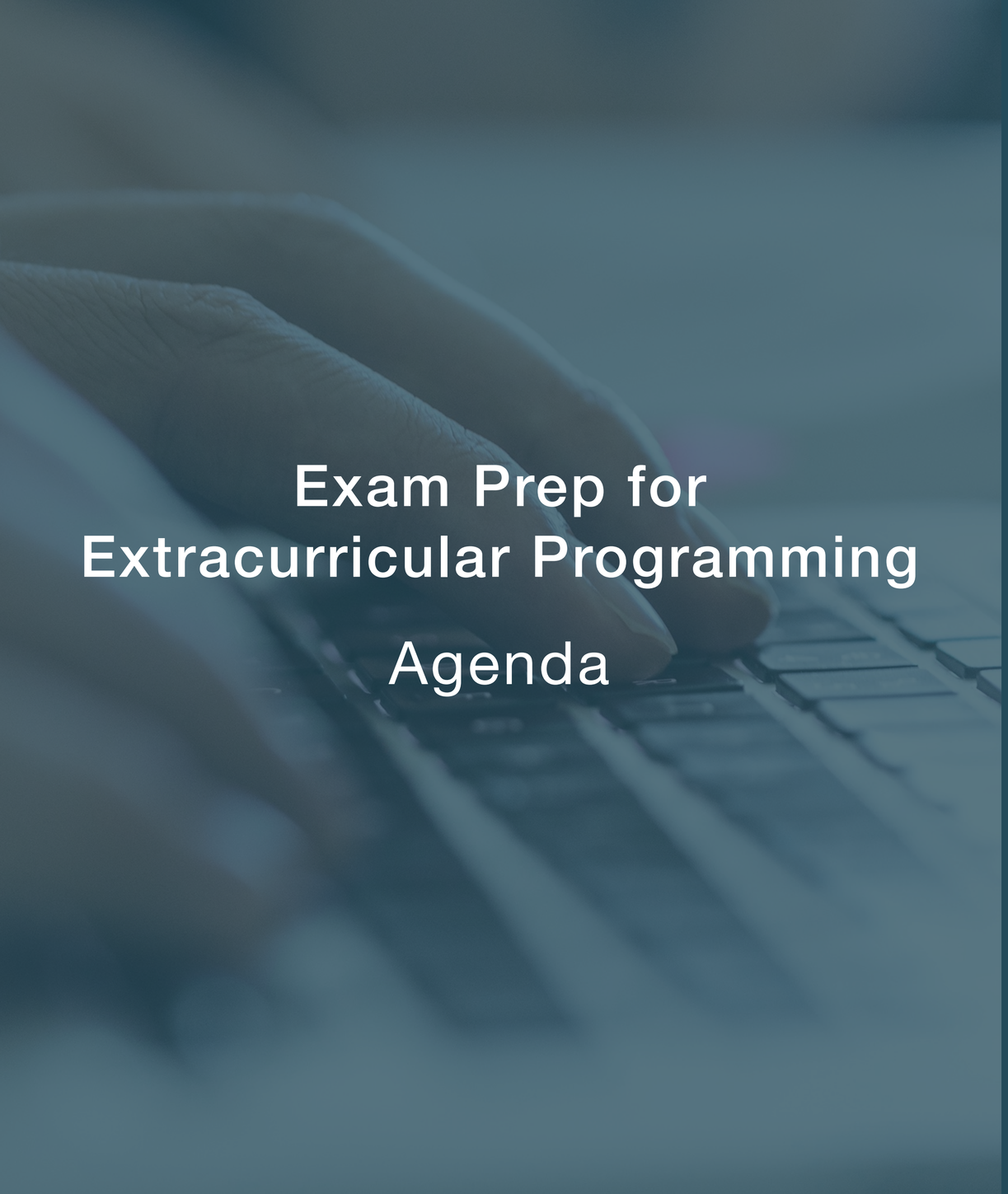
ERHVERVSAKADEMI AARHUS
BUSINESS ACADEMY AARHUS

Exam Prep for Extracurricular Programming

Welcome Back Programming 🥳



Exam Prep for Extracurricular Programming

Agenda

1. Client Server
2. Fetch, JSON & DOM Manipulation
3. Headless WordPress
4. Recap on JS terms & concepts
5. Exam Prep

Teachers - WP Headless

127.0.0.1:5501/index.html

Incognito

Teachers - WP Headless

Maria Louise Bendixen

Senior Lecturer

mlbe@eaaa.dk

Martin Aagaard Nøhr

Lecturer

mnor@eaaa.dk

Rasmus Cederdorff

Senior Lecturer & Web App Developer

race@eaaa.dk

Teachers ‹ Headless WordPre

headless.cederdorff.dk/wp-admin/edit.php?post_type=teacher

Headless WordPress - RACE 0 New Performance

Site Assistant
Dashboard
Posts
Media
Pages
Comments
Teachers
All Teachers
Add New Teacher
Appearance
Plugins
Users
Tools
Settings
ACF
Performance
Collapse menu

Teachers

Add New Teacher

All (3) | Published (3)

Bulk actions Apply All dates Filter

Title

Maria Louise Bendixen

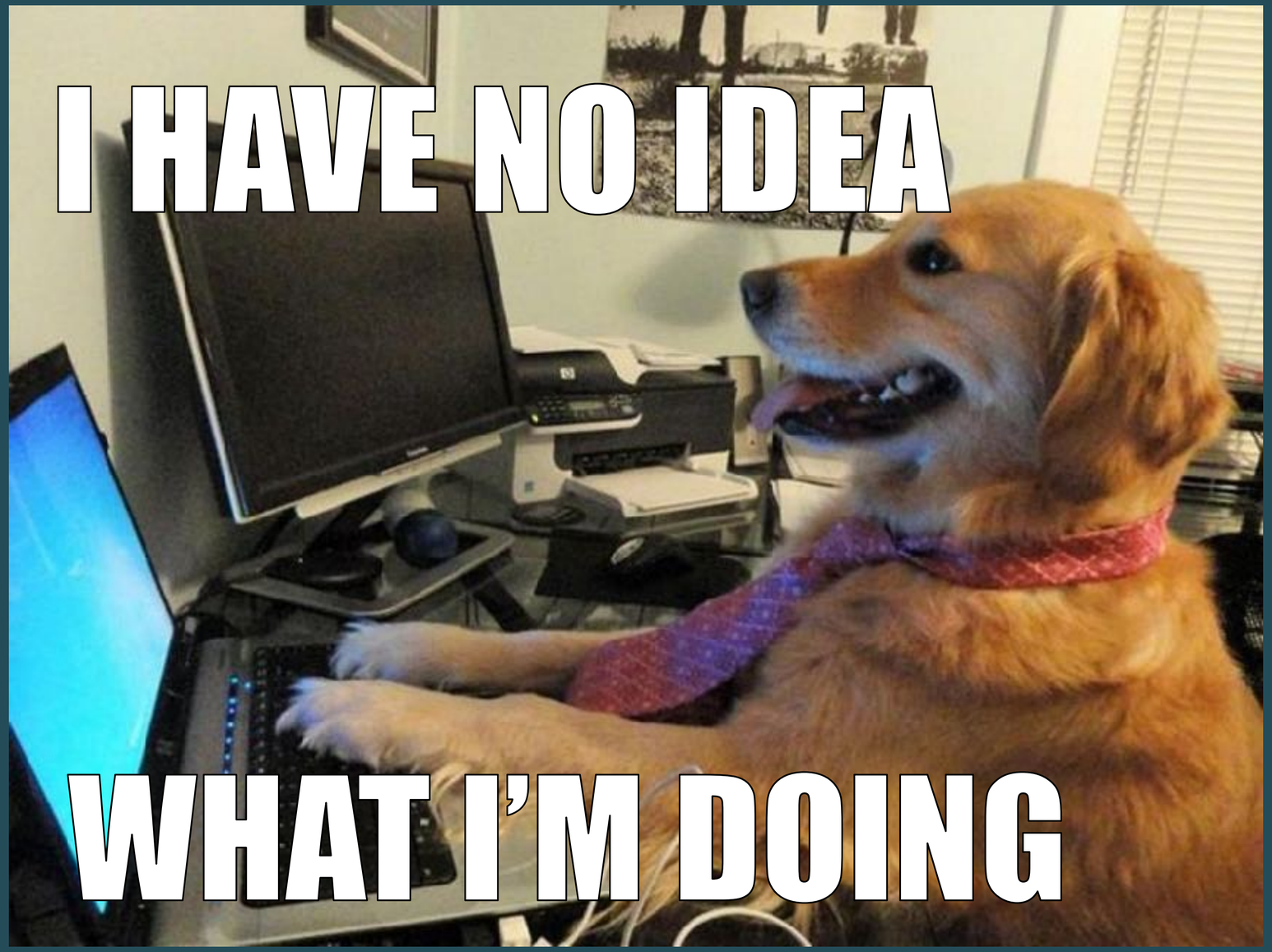
Rasmus Cederdorff

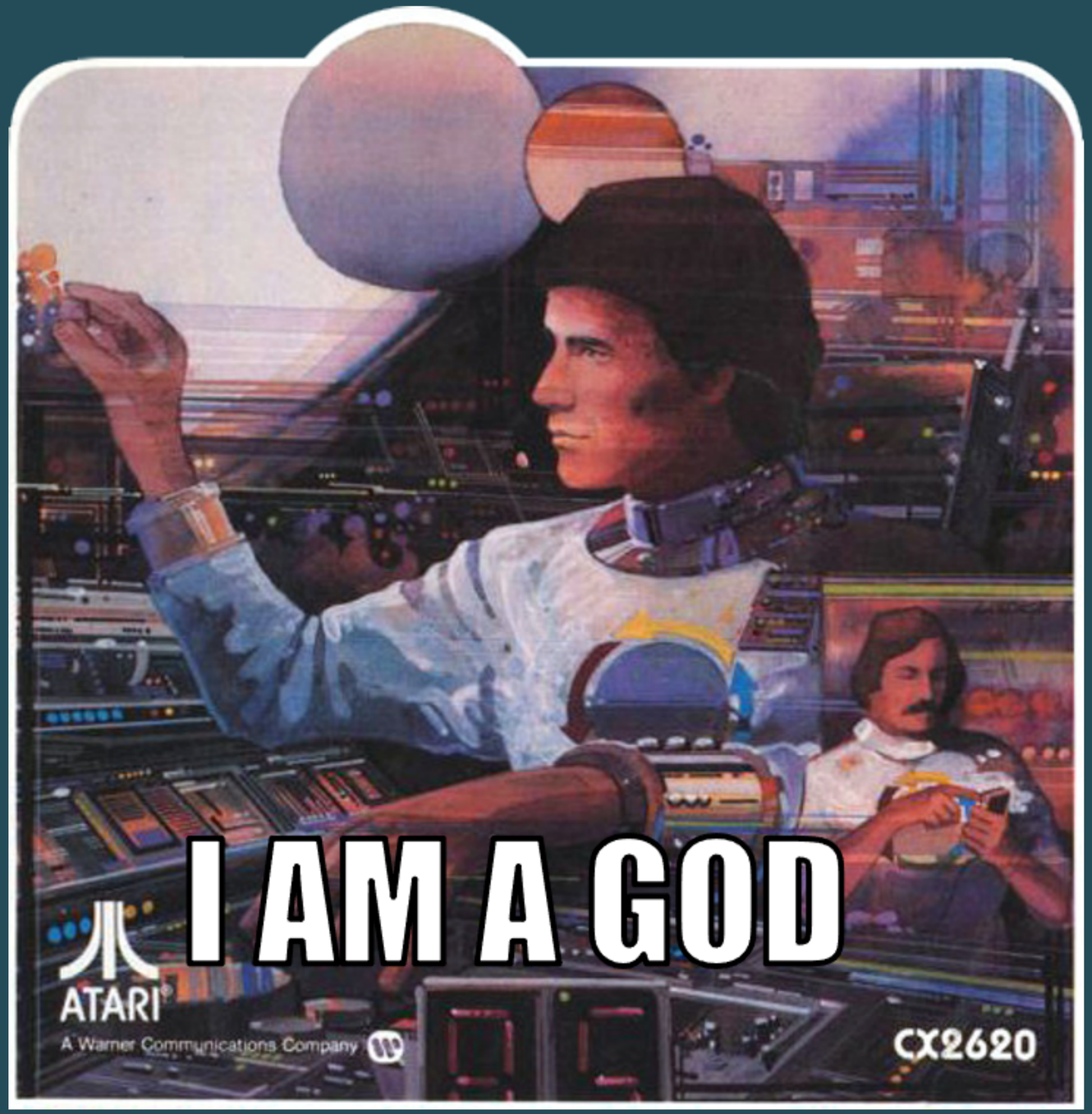
Martin Aagaard Nøhr

Title

Bulk actions Apply

Thank you for creating with WordPress.

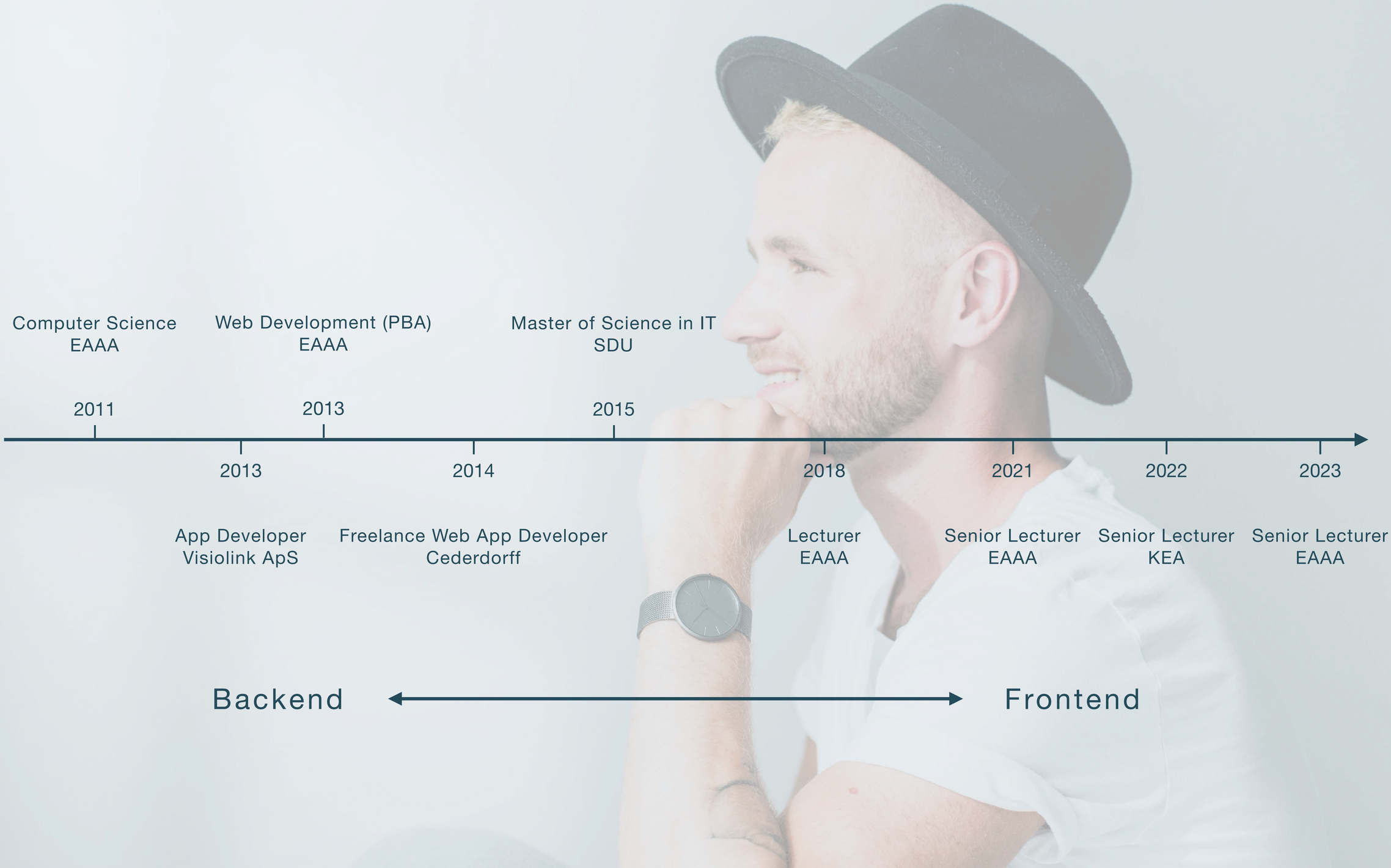

I HAVE NO IDEA
WHAT I'M DOING


I AM A GOD
ATARI
A Warner Communications Company
CX2620

I’m Rasmus Cederdorff (RACE)

Senior Lecturer
Freelance Web App Developer

- Programming with an eye for UI and UX.
- Web Development, JavaScript, React, BaaS & Node.js
- “I speak JavaScript”.
- Websites, Webshops, Web Apps, Mobile Apps, Server App and BaaS.


Computer Science
EAAA
2011
Web Development (PBA)
EAAA
2013
Master of Science in IT
SDU
2015
2013
2014
2018
2021
2022
2023
App Developer
Visiolink ApS
Freelance Web App Developer
Cederdorff
Lecturer
EAAA
Senior Lecturer
EAAA
Senior Lecturer
KEA
Senior Lecturer
EAAA
Backend
Frontend

I’m Rasmus Cederdorff

Aarhus (Tilst)

Alicia & Ida

From Holstebro

I’m into sports

Love (🍎) gadgets, to take pictures & interior design projects

What’s with my arm?

Client Server

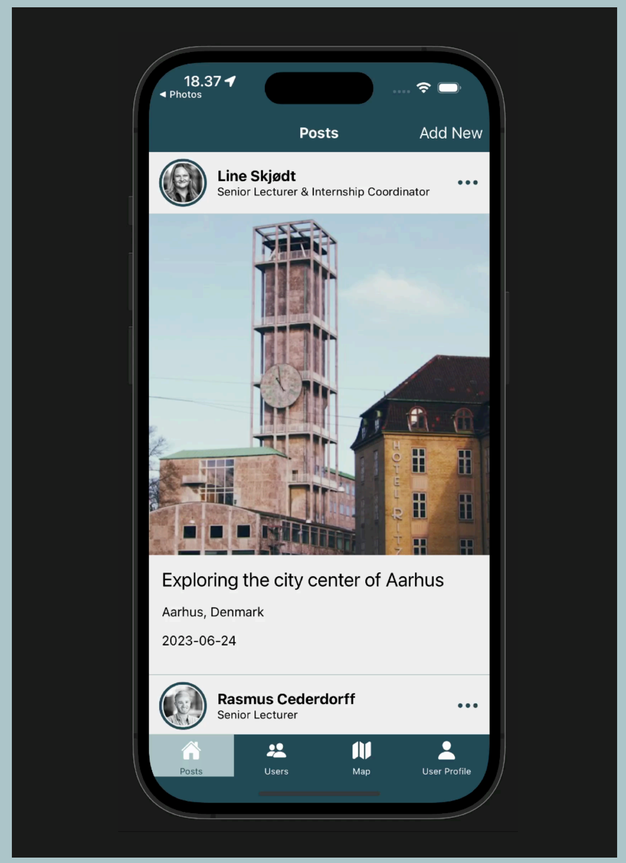
The Basic Architecture of the Web

Client Server Architecture

APP

REST API

DATA SOURCE



FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

JSON

CLIENT

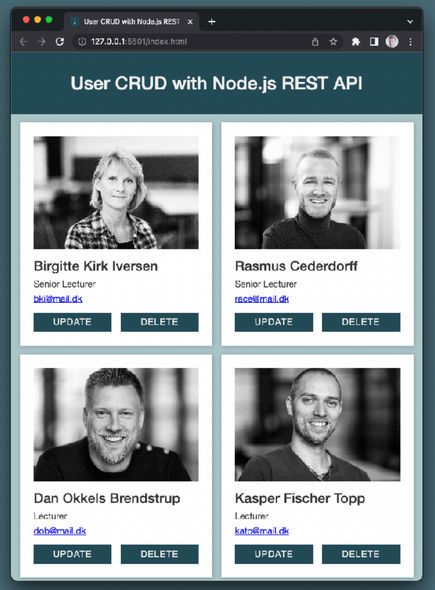
SERVER

Web Development

WEB APP

REST API

DATA SOURCE



cederdorff.com/dat-js-crud-intro

FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

posts

users

CLIENT

SERVER

Webudvikling

FRONTEND

REST API

BACKEND

FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

Express

Client-Server Model

Communication between web **clients** and web **servers**.

CLIENT

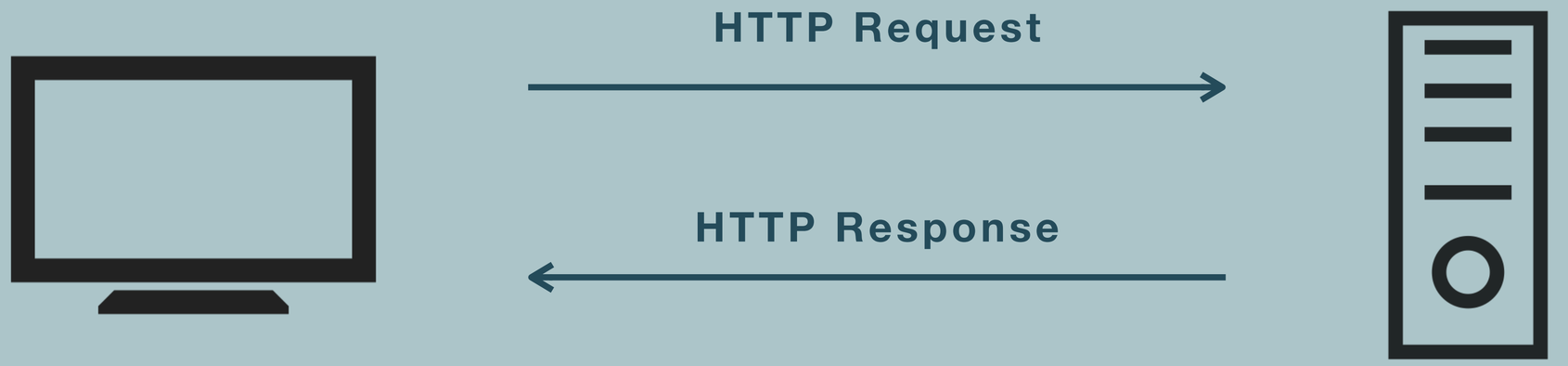
SERVER

Client-Server Model

Communication between web **clients** and web **servers**.

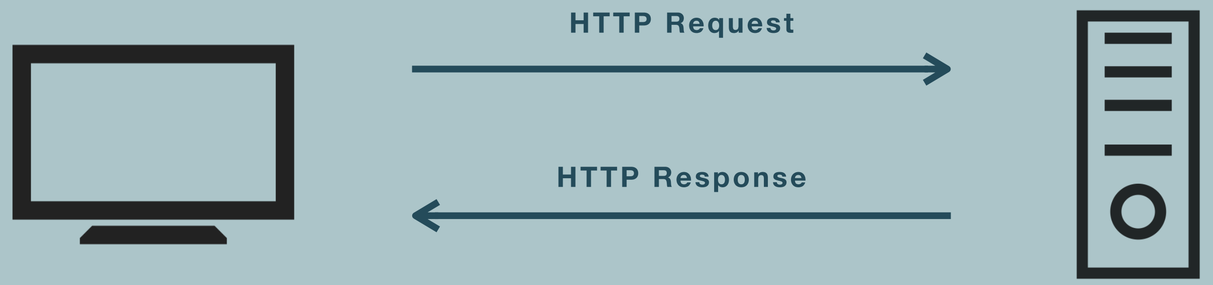
Client-Server Model

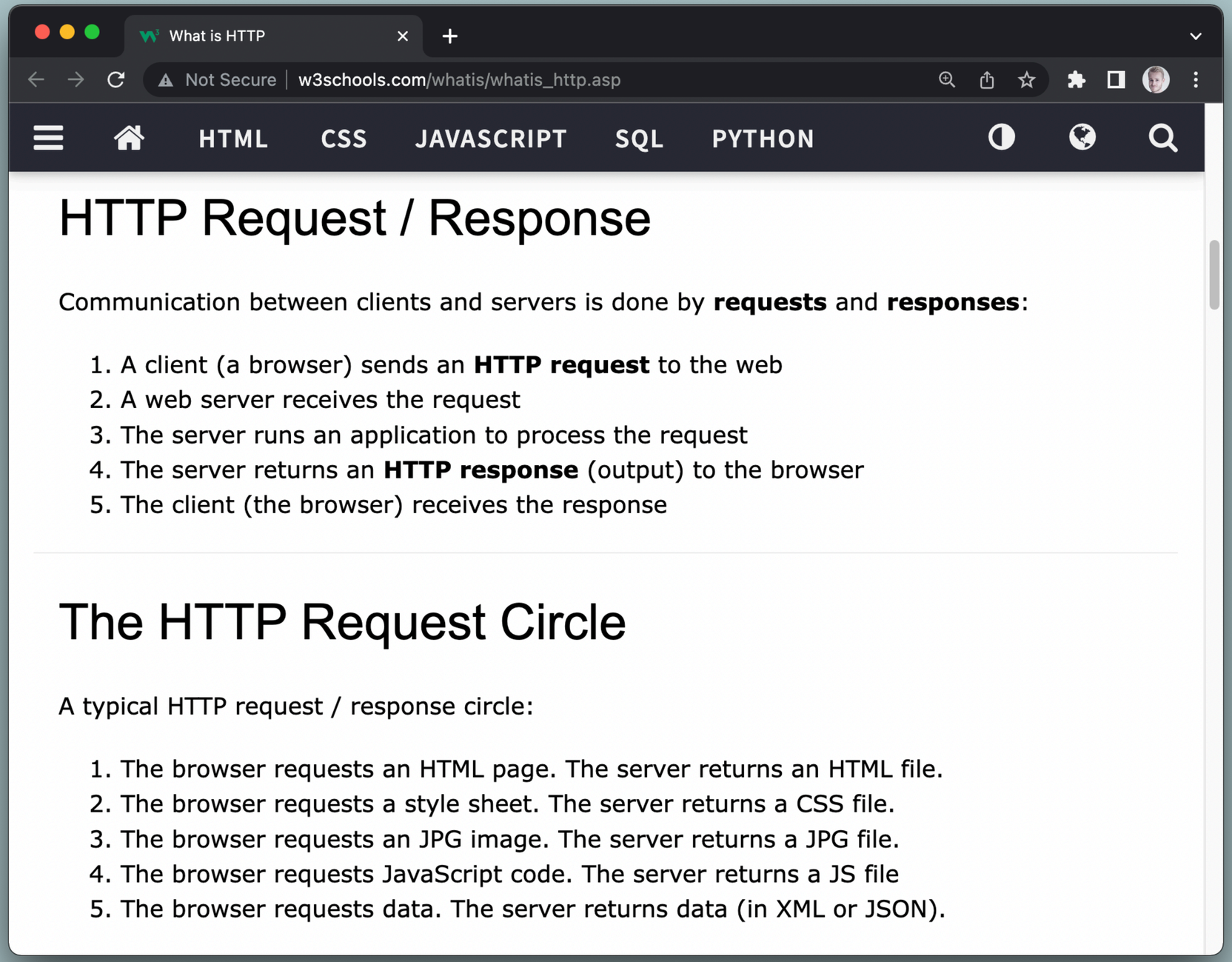
Communication between web **clients** and web **servers**.



Hyper Text Transfer Protocol

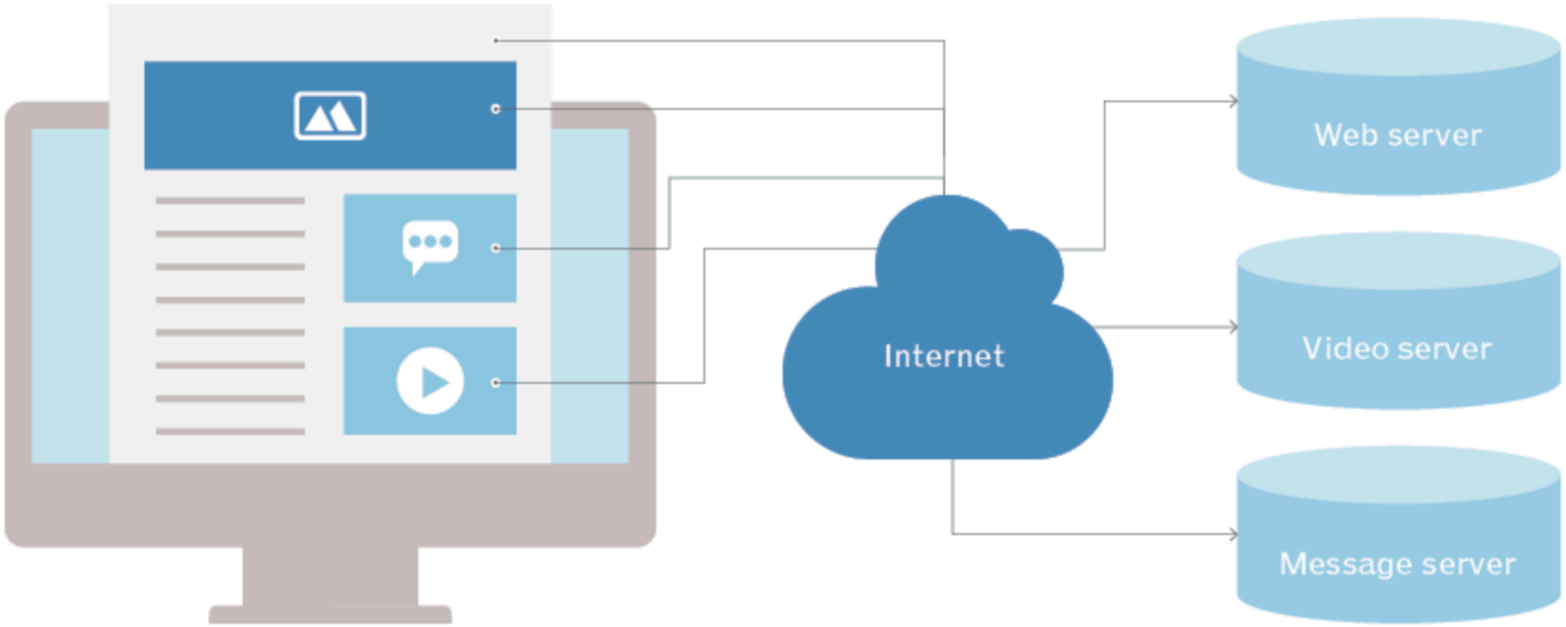
- A protocol and standard for fetching data, HTML and other resources (text, images, videos, scripts, JSON).
- The foundation of the web.





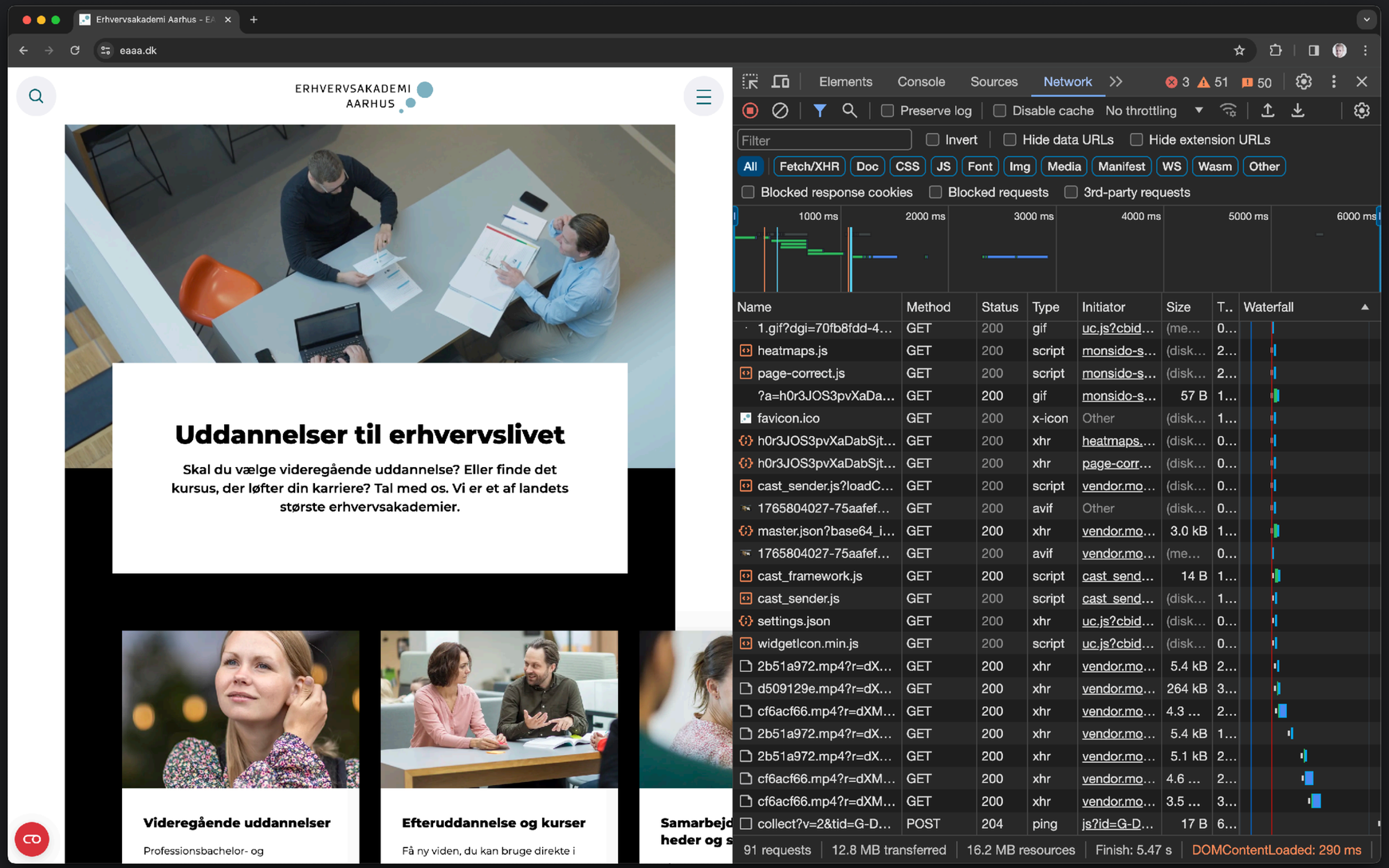
https://www.w3schools.com/whatis/whatis_http.asp

How HTTP Works

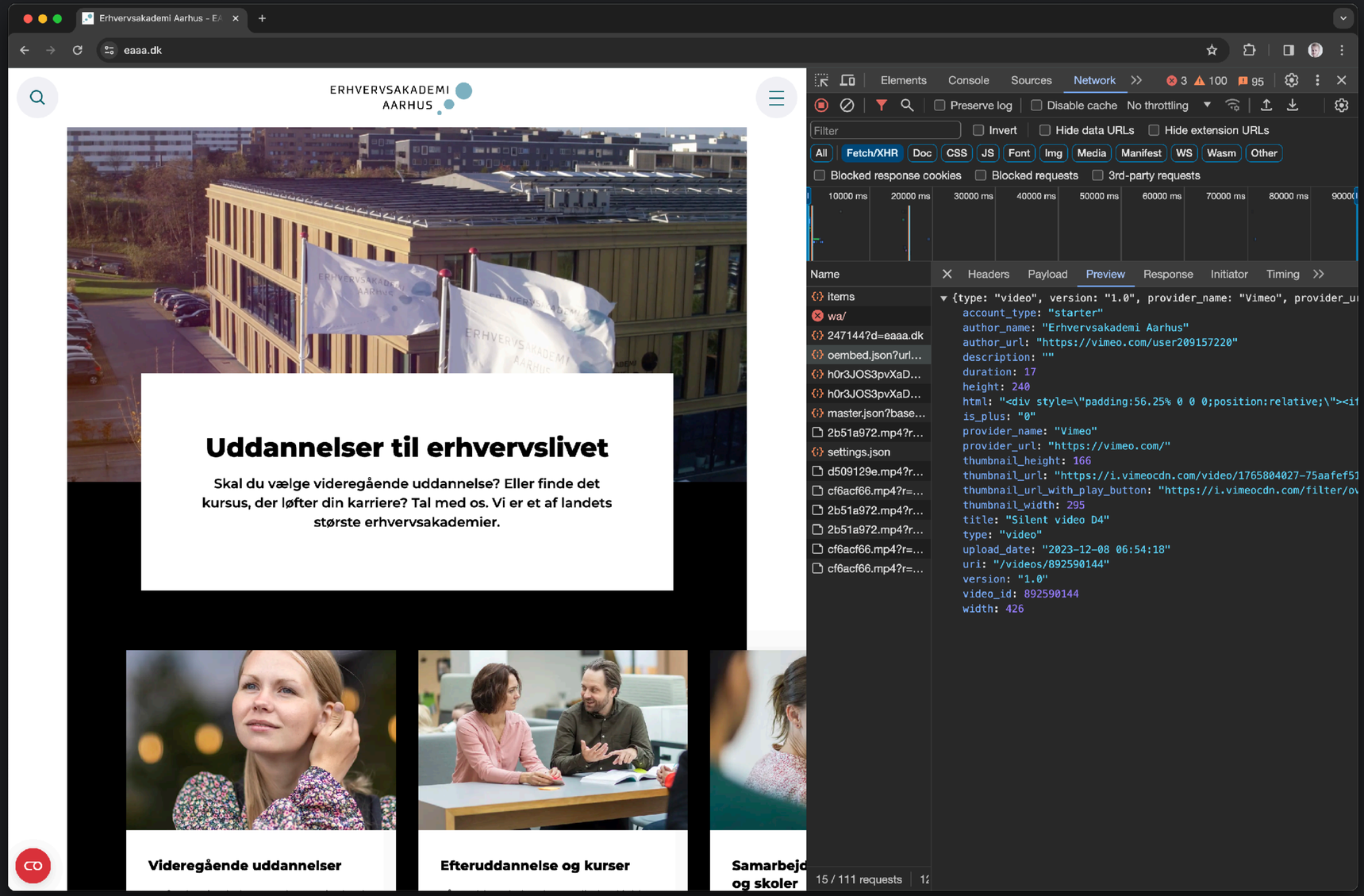


https://www.techtarget.com/whatis/definition/HTTP-Hypertext-Transfer-Protocol

Network Tab


Erhvervsakademi Aarhus - EA
eaaa.dk
ERHVERVSAKADEMI AARHUS
Uddannelser til erhvervslivet
Skal du vælge videregående uddannelse? Eller finde det kursus, der løfter din karriere? Tal med os. Vi er et af landets største erhvervsakademier.
Videregående uddannelser
Professionsbachelor- og
Efteruddannelse og kurser
Få ny viden, du kan bruge direkte i
Elements
Console
Sources
Network
3 51 50
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
1000 ms
2000 ms
3000 ms
4000 ms
5000 ms
6000 ms
Name
Method
Status
Type
Initiator
Size
T..
Waterfall
1.gif?dgi=70fb8fdd-4...
heatmaps.js
page-correct.js
?a=h0r3JOS3pvXaDa...
favicon.ico
h0r3JOS3pvXaDabSjt...
h0r3JOS3pvXaDabSjt...
cast_sender.js?loadC...
1765804027-75aafef...
master.json?base64_i...
1765804027-75aafef...
cast_framework.js
cast_sender.js
settings.json
widgetIcon.min.js
2b51a972.mp4?r=dX...
d509129e.mp4?r=dX...
cf6acf66.mp4?r=dXM...
2b51a972.mp4?r=dX...
2b51a972.mp4?r=dX...
cf6acf66.mp4?r=dXM...
cf6acf66.mp4?r=dXM...
collect?v=2&tid=G-D...
91 requests
12.8 MB transferred
16.2 MB resources
Finish: 5.47 s
DOMContentLoaded: 290 ms

Network Tab


Erhvervsakademi Aarhus - EA
eaaa.dk
ERHVERVSAKADEMI AARHUS
Uddannelser til erhvervslivet
Skal du vælge videregående uddannelse? Eller finde det kursus, der løfter din karriere? Tal med os. Vi er et af landets største erhvervsakademier.
Videregående uddannelser
Efteruddannelse og kurser
Samarbejd og skoler
Elements
Console
Sources
Network
Preserve log
Disable cache
No throttling
Filter
Invert
Hide data URLs
Hide extension URLs
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Blocked response cookies
Blocked requests
3rd-party requests
10000 ms
20000 ms
30000 ms
40000 ms
50000 ms
60000 ms
70000 ms
80000 ms
Name
items
wa/
247144?d=eaaa.dk
oembed.json?url...
h0r3JOS3pvXaD...
h0r3JOS3pvXaD...
master.json?base...
2b51a972.mp4?r...
settings.json
d509129e.mp4?r...
cf6acf66.mp4?r=...
2b51a972.mp4?r...
2b51a972.mp4?r...
cf6acf66.mp4?r=...
cf6acf66.mp4?r=...
15 / 111 requests
Headers
Payload
Preview
Response
Initiator
Timing
{type: "video", version: "1.0", provider_name: "Vimeo", provider_u
account_type: "starter"
author_name: "Erhvervsakademi Aarhus"
author_url: "https://vimeo.com/user209157220"
description: ""
duration: 17
height: 240
html: "<div style=\"padding:56.25% 0 0 0;position:relative;\"><if
is_plus: "0"
provider_name: "Vimeo"
provider_url: "https://vimeo.com/"
thumbnail_height: 166
thumbnail_url: "https://i.vimeocdn.com/video/1765804027-75aafef51
thumbnail_url_with_play_button: "https://i.vimeocdn.com/filter/ov
thumbnail_width: 295
title: "Silent video D4"
type: "video"
upload_date: "2023-12-08 06:54:18"
uri: "/videos/892590144"
version: "1.0"
video_id: 892590144
width: 426

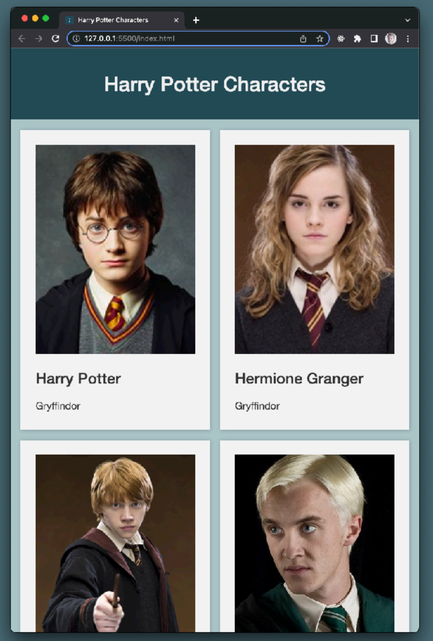
Network Tab

- Use the Network tab to investigate a website (e.g. kea.dk, dr.dk, google.dk, react-rest-and-auth.web.app or any other).
- Open the website in Google Chrome (or another browser).
- Open the Developer Tool (see: How to open the dev tool in your browser) and go to Netværk/Network.
- Reload the page while you are in the Network tab and check which resources are being downloaded.
- What type(s) of resources are they?
- Consider how the resources are being downloaded and why I tell you to do all this.

Fetch, HTTP Request & Response

WEB APP

DATA SOURCE



FETCH

REQUEST (GET)

JSON DATA

RESPONSE

potters

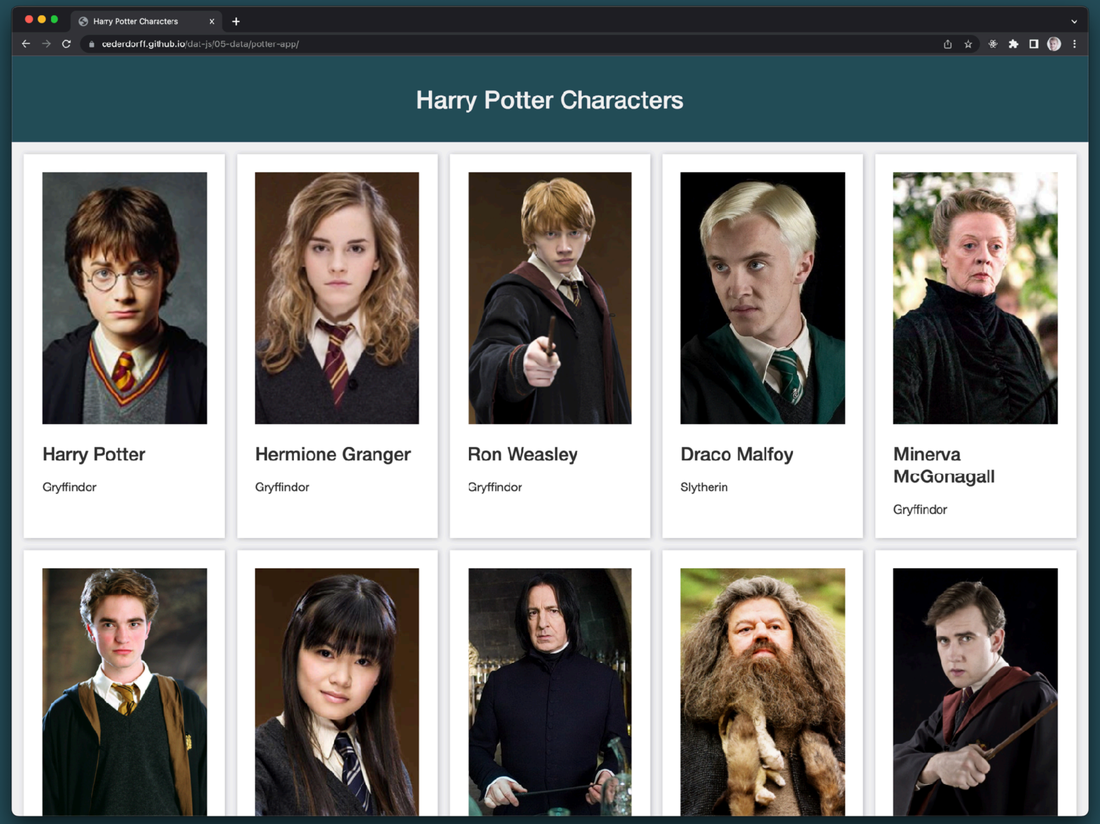
pokemons

southpark

CLIENT

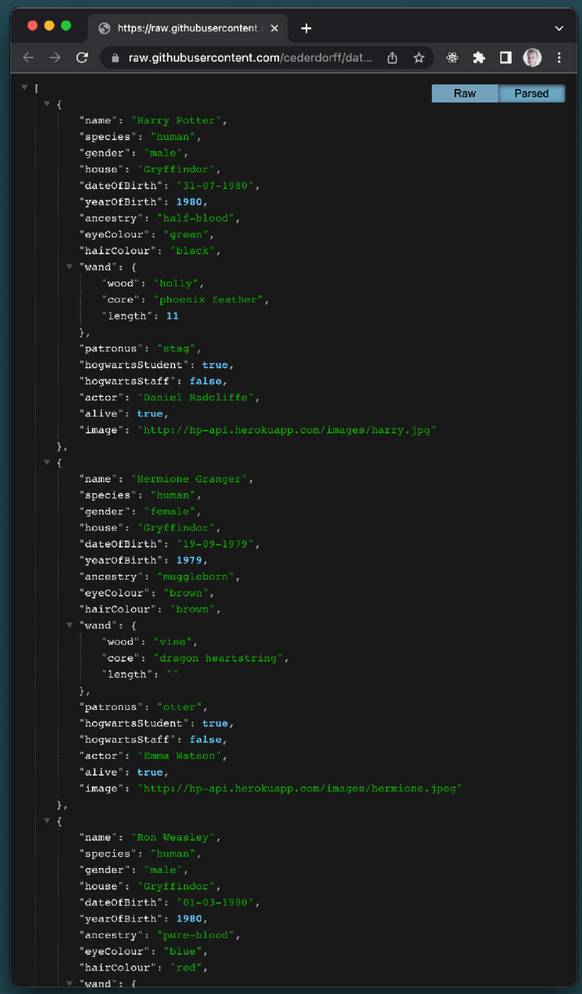
SERVER

Frontend (client)



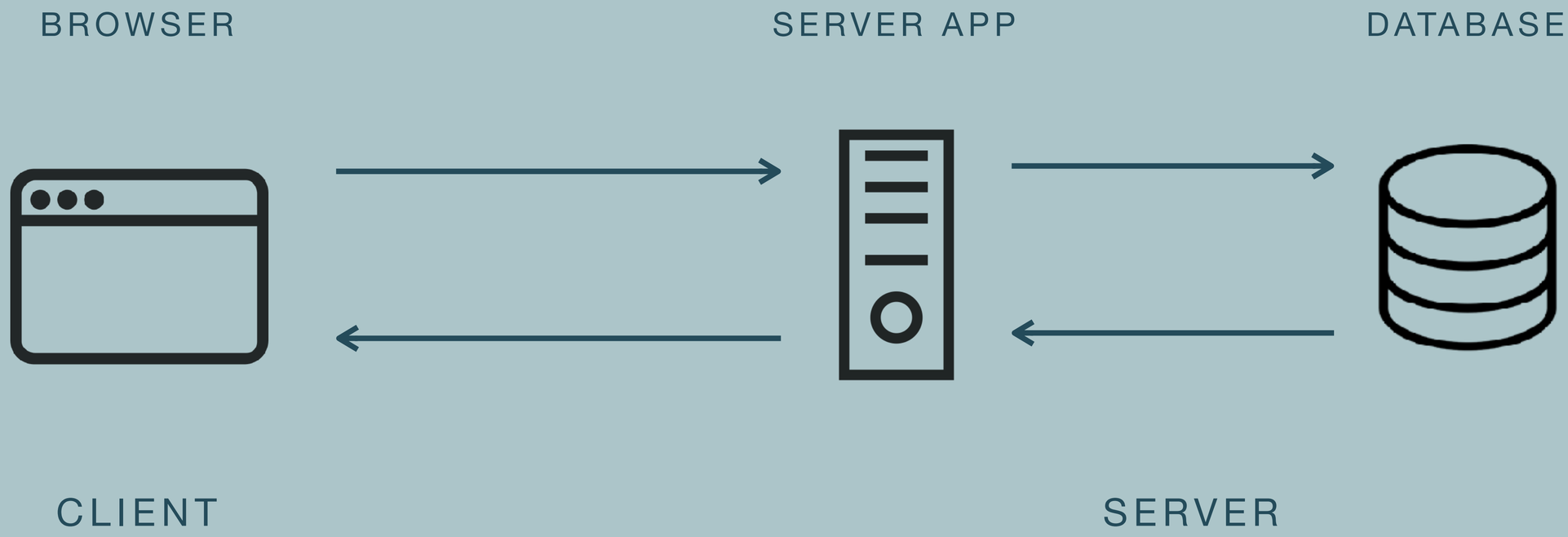
https://cederdorff.com/potter-app/

JSON Data Source (Server)



https://raw.githubusercontent.com/cederdorff/dat-js/main/data/potter.json

Web Dev Architecture

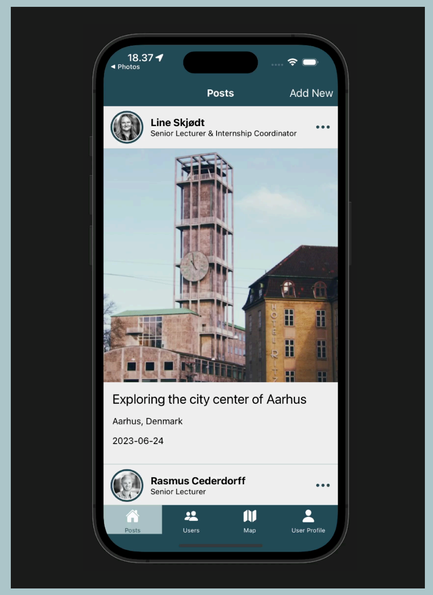


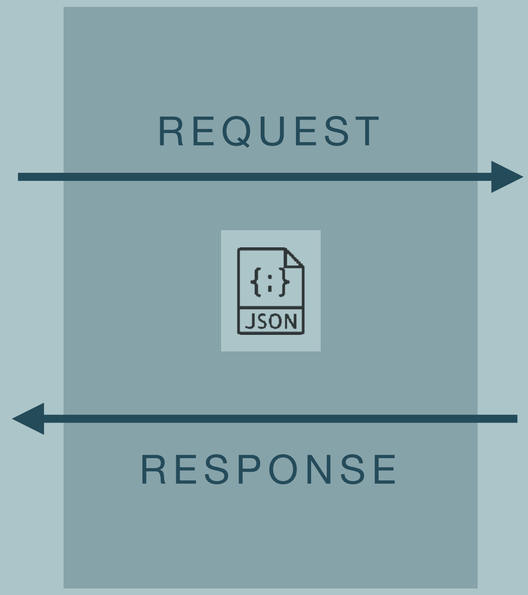
Web Dev Architecture

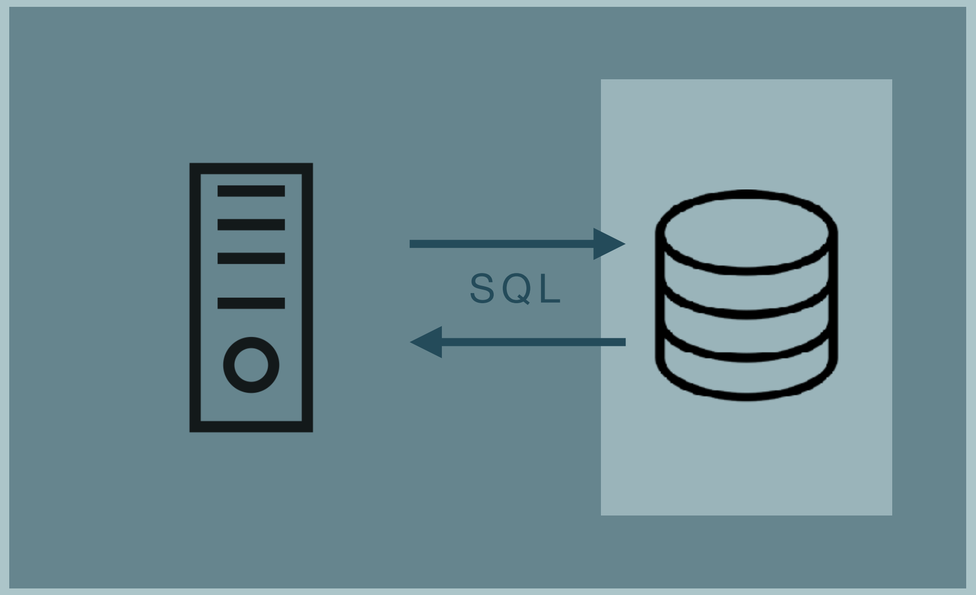
FRONTEND

REST API

BACKEND







CLIENT

SERVER

Payment Gateways

Social Media Integration

Healthcare Integration

Maps and Location Services

Weather Data

IoT (Internet of Things)

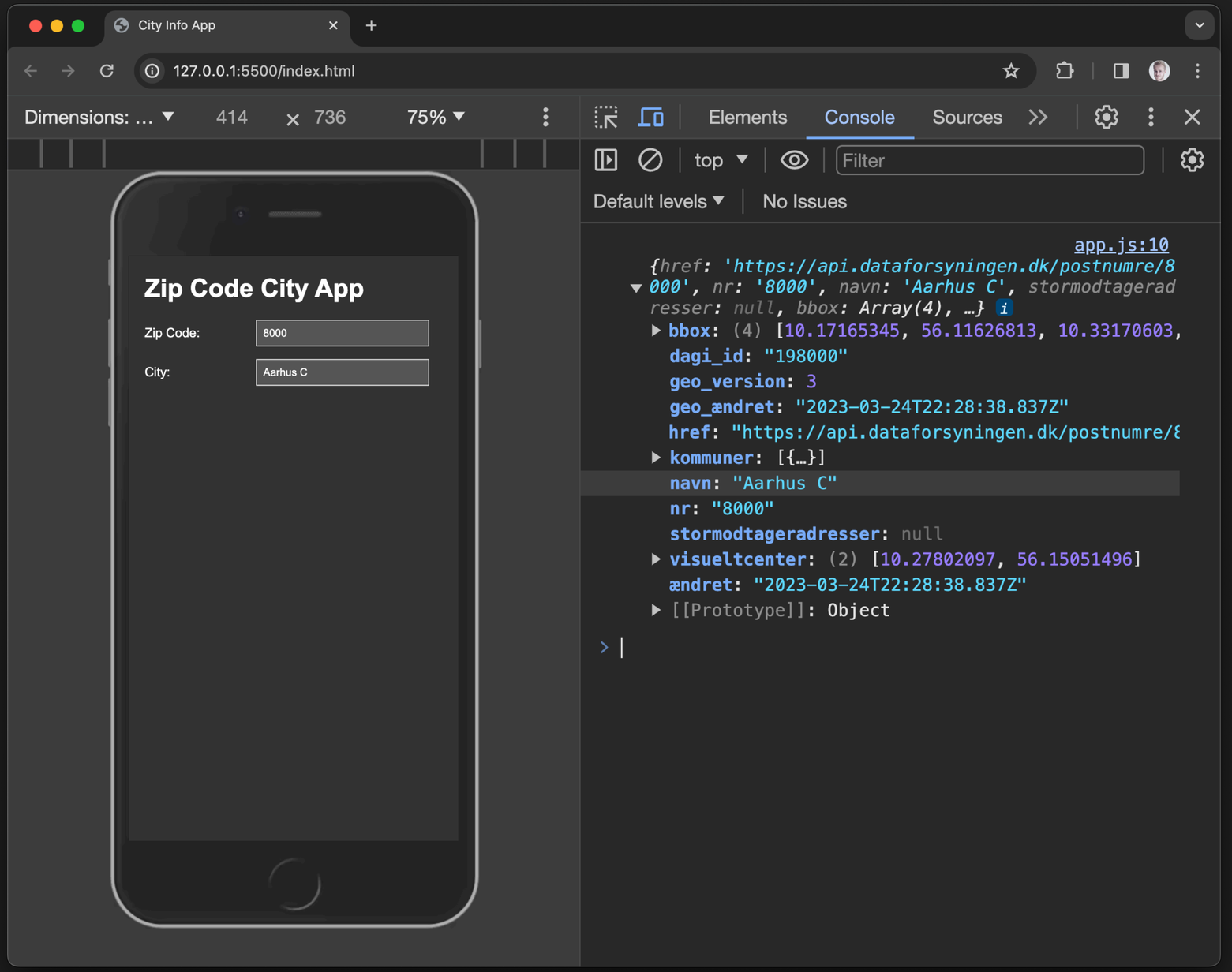
API

Cloud Services

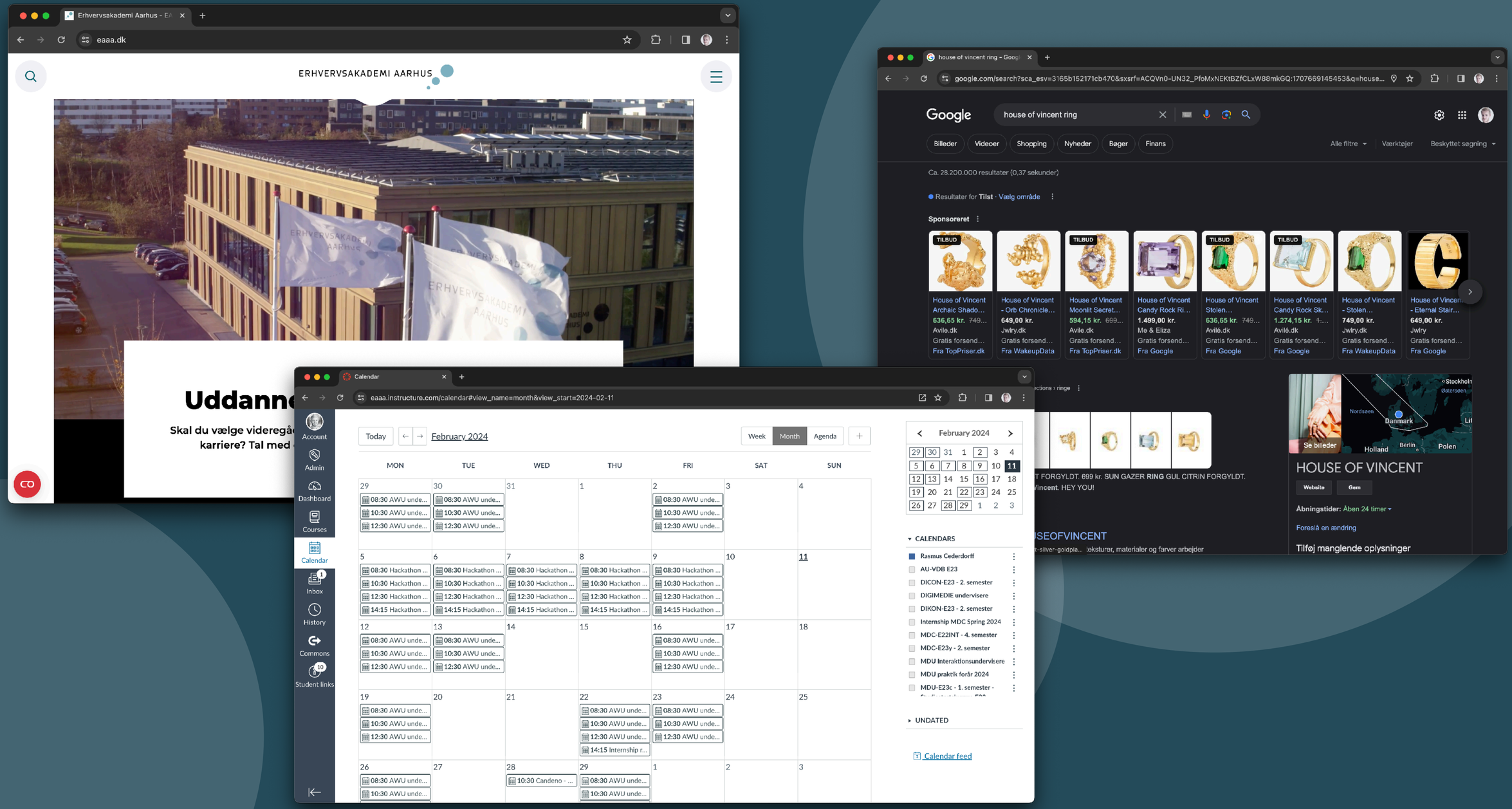
News and Content Aggregation

Financial Data

E-commerce Platforms



https://cederdorff.com/city-zip-code/


Erhvervsakademi Aarhus - EA
eaaa.dk
ERHVERSVAKADEMI AARHUS
Uddanne
Skal du vælge videregå
karriere? Tal med
Calendar
eaaa.instructure.com/calendar#view_name=month&view_start=2024-02-11
Account
Admin
Dashboard
Courses
Calendar
Inbox
History
Commons
Student links
Today
February 2024
Week
Month
Agenda
MON
TUE
WED
THU
FRI
SAT
SUN
CALENDARS
Rasmus Cederdorff
AU-VDB E23
DICON-E23 - 2. semester
DIGIMEDIE undervisere
DIKON-E23 - 2. semester
Internship MDC Spring 2024
MDC-E22INT - 4. semester
MDC-E23y - 2. semester
MDU Interaktionsundervisere
MDU praktik forår 2024
MDU-E23c - 1. semester -
UNDATED
Calendar feed
house of vincent ring - Google
google.com/search?sca_esv=3165b152171cb470&sxsrf=ACQVn0-UN32_PfoMxNEKtBZfCLxW88mkGQ:1707669145453&q=house...
Google
house of vincent ring
Billeder
Videoer
Shopping
Nyheder
Bøger
Finans
Alle filtre
Værktøjer
Beskyttet søgning
Ca. 28.200.000 resultater (0,37 sekunder)
Resultater for Tilst · Vælg område
Sponsoreret
TILBUD
House of Vincent Archaic Shado...
636,65 kr. 749...
Avile.dk
Gratis forsend...
Fra TopPriser.dk
House of Vincent - Orb Chronicle...
649,00 kr.
Jwlry.dk
Gratis forsend...
Fra WakeupData
House of Vincent Moonlit Secret...
594,15 kr. 699...
Avile.dk
Gratis forsend...
Fra TopPriser.dk
House of Vincent Candy Rock Ri...
1.499,00 kr.
Mø & Eliza
Gratis forsend...
Fra Google
House of Vincent Stolen...
636,65 kr. 749...
Avilé.dk
Gratis forsend...
Fra Google
House of Vincent Candy Rock Sk...
1.274,15 kr. 1...
Avilé.dk
Gratis forsend...
Fra Google
House of Vincent - Stolen...
749,00 kr.
Jwlry.dk
Gratis forsend...
Fra WakeupData
House of Vincent - Eternal Stair...
649,00 kr.
Jwlry
Gratis forsend...
Fra Google
T FORGYLDT. 699 kr. SUN GAZER RING GUL CITRIN FORGYLDT.
/incent. HEY YOU!
SEOFVINCENT
Se billeder
Stockholm
Østersøen
Nordsøen
Danmark
Holland
Berlin
Polen
HOUSE OF VINCENT
Website
Gem
Åbningstider: Åben 24 timer
Foreslå en ændring
Tilføj manglende oplysninger

Twitter API

Stripe API

Microsoft HealthVault API

Google Maps API

API

OpenWeatherMap API

Fitbit API

Amazon S3 API

The New York Times API

Alpha Vantage API

Shopify API

Fetch

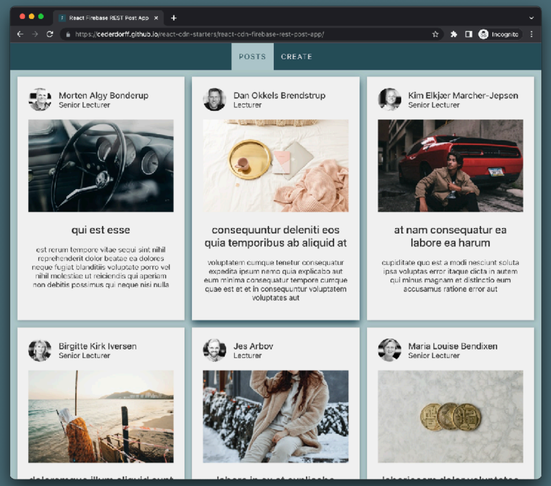
HTTP Requests in JavaScript

Fetch, HTTP Request & Response

WEB APP

FIREBASE REST API

DATA SOURCE



FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

JSON

Firebase
Realtime Database

posts

users

fetch(...)

HTTP Requests in JavaScript.

A way to get and post data from and to data sources.

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/ca
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
//
//
// or with async/await
const response = await fetch("https://cederdorff.gi
const data = await response.json();
console.log(data);
```

fetch(...)

HTTP Requests in JavaScript.

A way to get and post data from and to data sources.

`getCharacters` fetches a list of characters from a JSON data source, parses the JSON to JS and returns the data.

```
async function getCharacters() {
    const response = await fetch(url);
    const data = await response.json();
    console.log(data);
    return data;
}
```

fetch(...)

... get & post data from and to a data source
... can perform network requests to a server

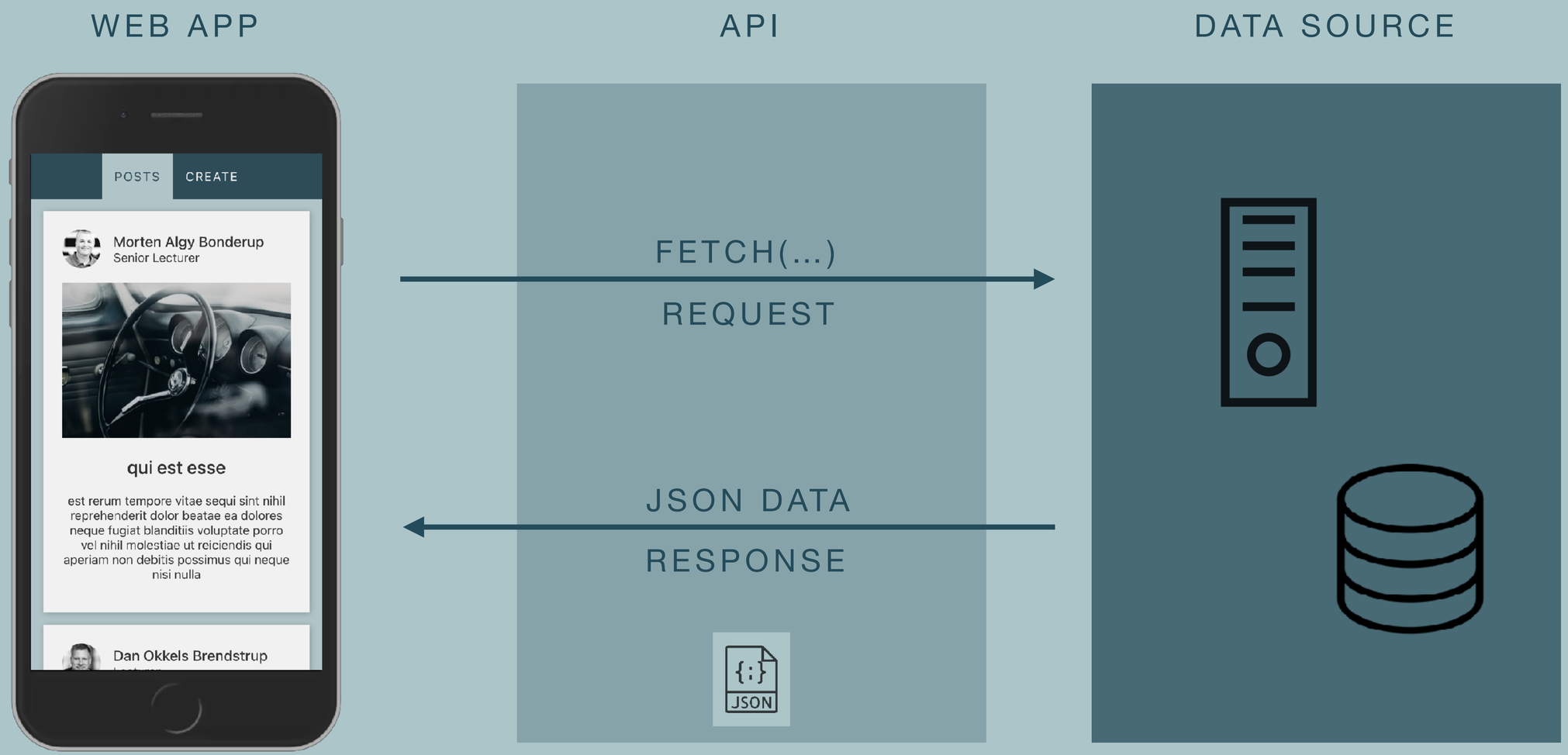
```
let promise = fetch(url, [options])
```

- `url` – the URL to access.
- `options` – optional parameters: method, headers etc.

Without `options`, this is a simple GET request, downloading the contents of the `url`.

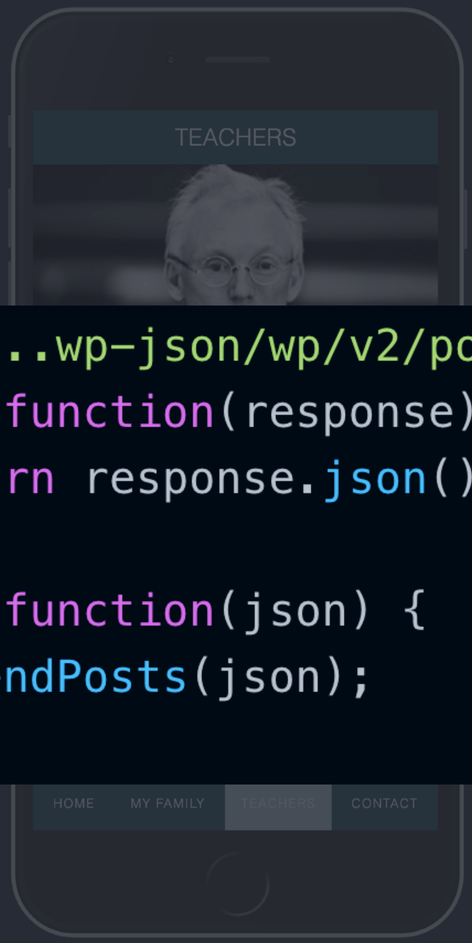
https://javascript.info/fetch

Fetch, fetch, fetch



Fetch

WEB APP




```
fetch("...wp-json/wp/v2/posts")
  .then(function(response) {
    return response.json();
  })
  .then(function(json) {
    appendPosts(json);
  });
```

REQUEST (fetch)

RESPONSE (then)

DATA SOURCE

```js
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

```json
[
  {
    "name": "Peter Madsen",
    "age": 52,
    "hairColor": "blonde",
    "relation": "dad",
    "img": "img/dad.jpg"
  },
  {
    "name": "Ane Madsen",
    "age": 51,
    "hairColor": "brown",
    "relation": "mom",
    "img": "img/ane.jpg"
  },
  {
    "name": "Rasmus Madsen",
    "age": 28,
    "hairColor": "blonde",
    "relation": "brother",
    "img": "img/IMG_0526_kvadrat.jpg"
  },
  {
    "name": "Mie Madsen",
    "age": 25,
    "hairColor": "brown",
    "relation": "blonde",
    "img": "img/mie.jpg"
  },
  {
    "name": "Mads Madsen",
    "age": 18,
    "hairColor": "dark",
    "relation": "blonde",
    "img": "img/mads.jpg"
  },
  {
    "name": "Jens Madsen",
    "age": 14,
    "hairColor": "blonde",
    "relation": "uncle",
    "img": "img/jenspeter.jpg"
  }
]
```

fetch-family-members

```js
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

REQUEST (fetch)

RESPONSE (then)

```json
[
  {
    "name": "Peter Madsen",
    "age": 52,
    "hairColor": "blonde",
    "relation": "dad",
    "img": "img/dad.jpg"
  },
  {
    "name": "Ane Madsen",
    "age": 51,
    "hairColor": "brown",
    "relation": "mom",
    "img": "img/ane.jpg"
  },
  {
    "name": "Rasmus Madsen",
    "age": 28,
    "hairColor": "blonde",
    "relation": "brother",
    "img": "img/IMG_0526_kvadrat.jpg"
  },
  {
    "name": "Mie Madsen",
    "age": 25,
    "hairColor": "brown",
    "relation": "blonde",
    "img": "img/mie.jpg"
  },
  {
    "name": "Mads Madsen",
    "age": 18,
    "hairColor": "dark",
    "relation": "blonde",
    "img": "img/mads.jpg"
  },
  {
    "name": "Jens Madsen",
    "age": 14,
    "hairColor": "blonde",
    "relation": "uncle",
    "img": "img/jenspeter.jpg"
  }
]
```

fetch-family-members

```
/*
Fetches json data from the file persons.json
*/
fetch('json/persons.json')
  .then(function (response) {
    return response.json();
  })
  .then(function (jsonData) {
    console.log(jsonData);
    appendPersons(jsonData)
  });

/*
Appends json data to the DOM
*/
function appendPersons(persons) {
  let htmlTemplate = "";
  for (let person of persons) {
    htmlTemplate += /*html*/`
      <article>
        <img src="${person.img}">
        <h4>${person.name}</h4>
        <p>${person.age} years old</p>
        <p>Hair color: ${person.hairColor}</p>
        <p>Relation: ${person.relation}</p>
      </article>
    `;
  }
  document.querySelector("#persons").innerHTML = htmlTemplate;
}
```

```
},
{
  "name": "Rasmus Madsen",
  "age": 28,
  "hairColor": "blonde",
  "relation": "brother",
  "img": "img/IMG_0526_kvadrat.jpg"
},
{
  "name": "Mie Madsen",
  "age": 25,
  "hairColor": "brown",
  "relation": "blonde",
  "img": "img/mie.jpg"
},
{
  "name": "Mads Madsen",
  "age": 18,
  "hairColor": "dark",
  "relation": "blonde",
  "img": "img/mads.jpg"
},
{
```

fetch-family-members

request (fetch)

response (JSON)

app.js — web-diplom-frontend

app.js

fetch-persons-grid > app.js > ...

```
let persons = []; // global variable

async function getPerson() {
    const response = await fetch(
        "https://raw.githubusercontent.com/cederdorff/web-diplom-frontend/main/_data/persons.json"
    ); // fetch request - fetch data from a given url
    persons = await response.json(); // setting global variable with fetched data
    displayPersons(persons); // calling displayPersons with persons as parameter
}

function displayPersons(listOfPersons) {
    let html = ""; // variable to store html
    //loop through all persons and create an article with content for each
    for (const person of listOfPersons) {
        html += /*html*/ `
            <article>
                <img src="${person.img}">
                <h2>${person.name}</h2>
                <p>${person.title}</p>
                <a href="mailto:${person.mail}">${person.mail}</a>
            </article>
        `; // generate and save html for every person in html variable
    }
    // set grid container content with person <article> elements
    // saved in html
    document.querySelector("#content").innerHTML = html;
}

getPerson(); // execute get persons to start the fun
```

main ⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF {} JavaScript Go Live ✓ Prettier

https://raw.githubusercontent.c

raw.githubusercontent.com/cederdorff/web-diplom-frontend/mai...

Raw Parsed


```
[
    {
        "name": "Birgitte Kirk Iversen",
        "mail": "bkj@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
    },
    {
        "name": "Martin Aagaard Nøhr",
        "mail": "mnor@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/oayjq02h/martin-n%C3%B8hr.jpg?width=800&height=450"
    },
    {
        "name": "Rasmus Cederdorff",
        "mail": "race@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
    },
    {
        "name": "Dan Okkels Brendstrup",
        "mail": "dob@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/bdojel41/dan-okkels-brendstrup.jpg?width=800&height=450"
    },
    {
        "name": "Line Skjødt",
        "mail": "lskj@mail.dk",
        "title": "Senior Lecturer & Internship Coordinator",
        "img": "https://www.eaaa.dk/media/14qpfeq4/line-skj%C3%B8dt.jpg?width=800&height=450"
    },
    {
        "name": "Kasper Fischer Topp",
        "mail": "kato@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
    },
    {
        "name": "Anne Kirketerp",
        "mail": "anki@mail.dk",
```

request (fetch)

response (JSON)

app.js — web-diplom-frontend

app.js

fetch-persons-grid > app.js > ...

```
let persons = []; // global variable

async function getPerson() {
    const response = await fetch(
        "https://raw.githubusercontent.com/cederdorff/web-diplom-frontend/main/_data/persons.json"
    ); // fetch request - fetch data from a given url
    persons = await response.json(); // setting global variable with fetched data
    displayPersons(persons); // calling displayPersons with persons as parameter
}

function displayPersons(listOfPersons) {
    let html = ""; // variable to store html
    //loop through all persons and create an article with content for each
    for (const person of listOfPersons) {
        html += /*html*/ `
            <article>
                <img src="${person.img}">
                <h2>${person.name}</h2>
                <p>${person.title}</p>
                <a href="mailto:${person.mail}">${person.mail}</a>
            </article>
        `; // generate and save html for every person in html variable
    }
    // set grid container content with person <article> elements
    // saved in html
    document.querySelector("#content").innerHTML = html;
}

getPerson(); // execute get persons to start the fun
```

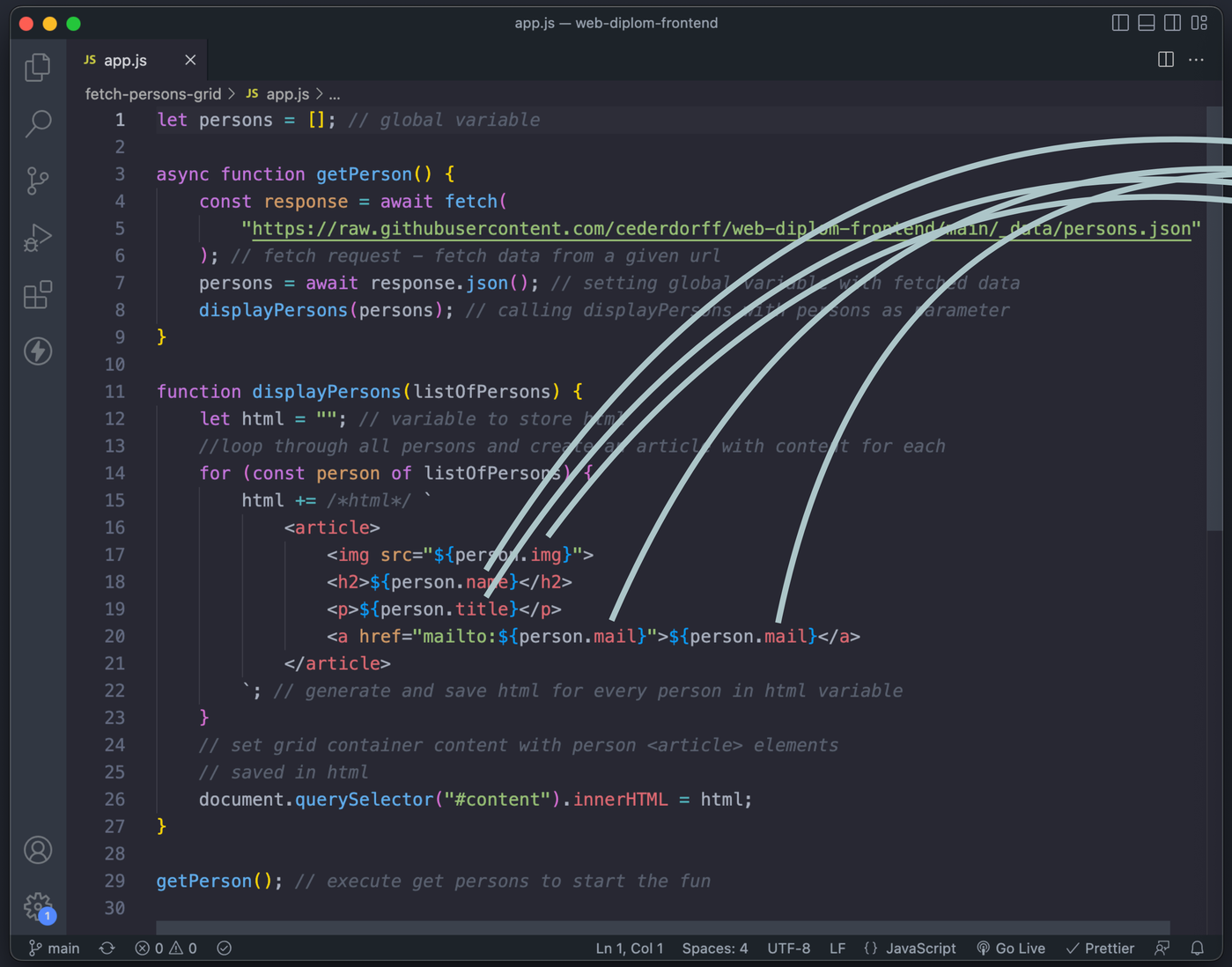
main ⊗ 0 ⚠ 0 Ln 1, Col 1 Spaces: 4 UTF-8 LF {} JavaScript Go Live ✓ Prettier

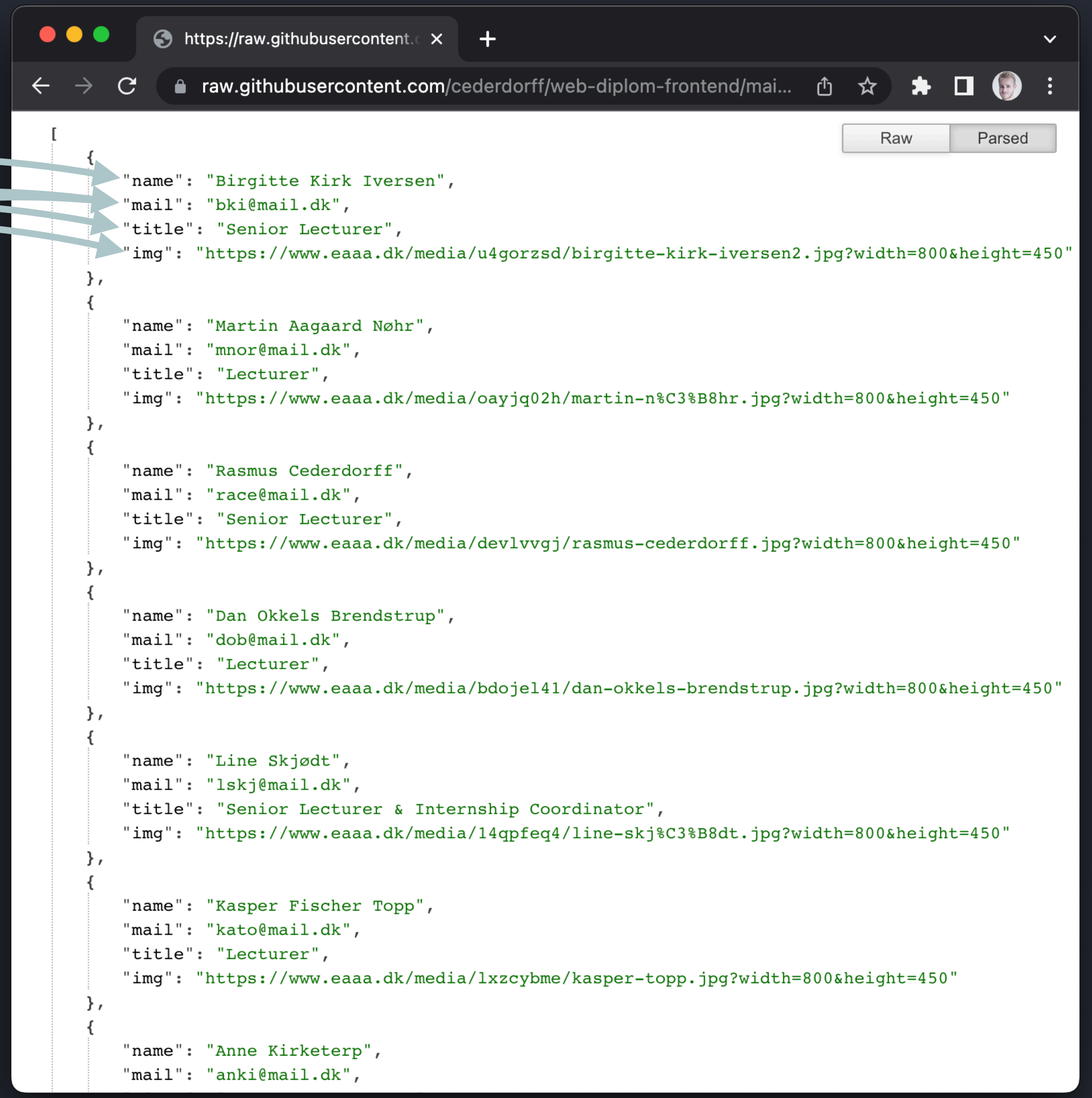
https://raw.githubusercontent.c

raw.githubusercontent.com/cederdorff/web-diplom-frontend/mai...

Raw Parsed


```
[
    {
        "name": "Birgitte Kirk Iversen",
        "mail": "bkj@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg?width=800&height=450"
    },
    {
        "name": "Martin Aagaard Nøhr",
        "mail": "mnor@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/oayjq02h/martin-n%C3%B8hr.jpg?width=800&height=450"
    },
    {
        "name": "Rasmus Cederdorff",
        "mail": "race@mail.dk",
        "title": "Senior Lecturer",
        "img": "https://www.eaaa.dk/media/devlvvgj/rasmus-cederdorff.jpg?width=800&height=450"
    },
    {
        "name": "Dan Okkels Brendstrup",
        "mail": "dob@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/bdojel41/dan-okkels-brendstrup.jpg?width=800&height=450"
    },
    {
        "name": "Line Skjødt",
        "mail": "lskj@mail.dk",
        "title": "Senior Lecturer & Internship Coordinator",
        "img": "https://www.eaaa.dk/media/14qpfeq4/line-skj%C3%B8dt.jpg?width=800&height=450"
    },
    {
        "name": "Kasper Fischer Topp",
        "mail": "kato@mail.dk",
        "title": "Lecturer",
        "img": "https://www.eaaa.dk/media/lxzcybme/kasper-topp.jpg?width=800&height=450"
    },
    {
        "name": "Anne Kirketerp",
        "mail": "anki@mail.dk",
```





fetch-persons-grid

index.html — web-diplom-frontend

app.js

```
fetch-persons-grid > app.js > ...
```

```js
let persons = []; // global variable

async function getPerson() {
    const response = await fetch(
        "https://raw.githubusercontent.com/cederdorff/web-diplom-frontend/main/_data/persons.json"
    ); // fetch request - fetch data from a given url
    persons = await response.json(); // setting global variable with fetched data
    displayPersons(persons); // calling displayPersons with persons as parameter
}

function displayPersons(listOfPersons) {
    let html = ""; // variable to store html
    //loop through all persons and create an article with content for each
    for (const person of listOfPersons) {
        html += /*html*/ `
            <article>
                <img src="${person.img}">
                <h2>${person.name}</h2>
                <p>${person.title}</p>
                <a href="mailto:${person.mail}">${person.mail}</a>
            </article>
        `; // generate and save html for every person in html variable
    }
    // set grid container content with person <article> elements
    // saved in html
    document.querySelector("#content").innerHTML = html;
}

getPerson(); // execute get persons to start the fun
```

index.html

```
fetch-persons-grid > index.html > html > body > main > section#content.grid-container
```

```html
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="app.css">
    <title>Fetch Persons</title>
</head>

<body>
    <header>
        <h1>Fetch Persons</h1>
    </header>
    <main>
        <section id="content" class="grid-container"></section>
    </main>
    <script src="app.js"></script>
</body>

</html>
```

DOM Manipulation

main ⊗ 0 ⚠ 0 — Ln 17, Col 9 Spaces: 4 UTF-8 LF HTML Go Live Prettier

fetch-persons-grid

Fetch

... get & post data from and to a data source

```
// Simple javascipt 😄

//Synchronous fetch using async/await.

// Usual way
✓ const jsonData = fetch('URL')
                    .then(response ⇒ response.json())
                    .then(json ⇒ console.log(json));

// Using await
✓ const jsonData = await fetch('URL').then(res ⇒ res.json())


// Shorter syntax 😍
✓ const jsonData = await (await fetch('URL')).json();
```

https://www.instagram.com/p/B0nxQjXj9Zi/

Async JS

JavaScript reads and runs the script from top to bottom.

JavaScript functions are executed in the sequence they are called. Not in the sequence they are defined.

... by default JavaScript is synchronous.

JS is Synchronous & Single-Threaded

```
function myFirst() {
    console.log("Hello");
}

function mySecond() {
    console.log("Goodbye");
}

mySecond();
myFirst();
```

```
/* ---------- Global Variables ---------- */
let _users = [];
let _selectedUserId;

/* ------------------------------------- */

async function fetchUsers() {…
}
function appendUsers(usersArray) {…
}

// ========= INIT APP =========

async function initApp() {
    await fetchUsers();
    appendUsers(_users);
}

initApp();
```

With callbacks, we can make JS Asynchronous

```
setTimeout(() => {
    console.log("Hey, I'm async!");
}, 3000);

btn.addEventListener('click', () => {
    alert("Hey, you clicked me!");
});
```

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/canva
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
```

Fetch is Asynchronous

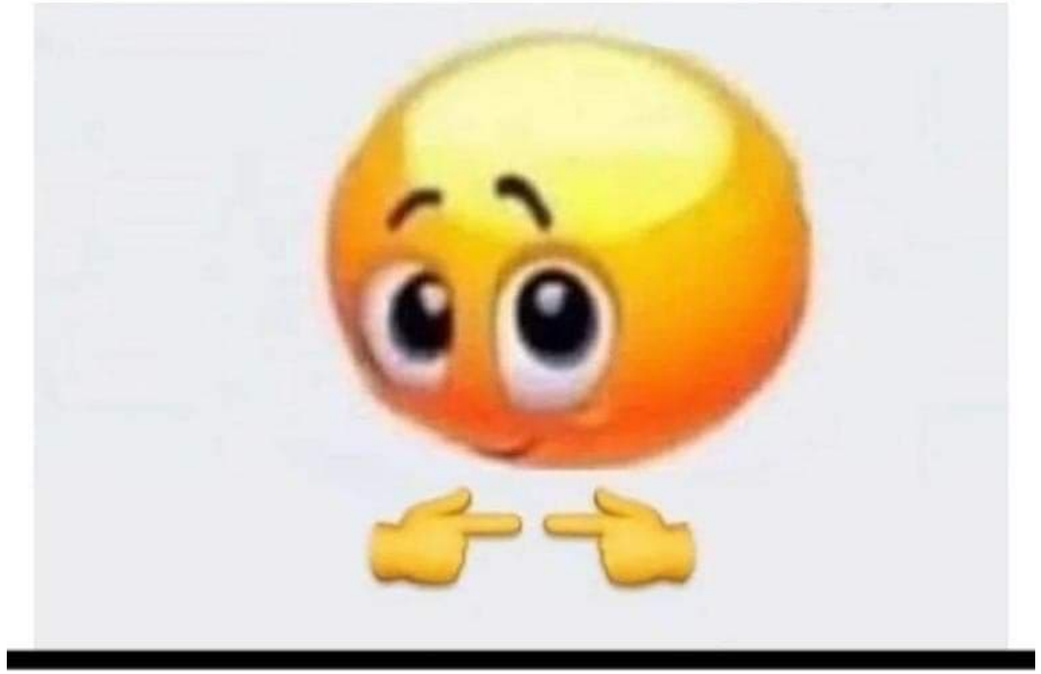
And it’s about making HTTP requests in JavaScript.
... and a way to get & post data from and to a data source.

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/canvas-users/data.json")
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
```

Fetch: callback (`then`) vs `async/await`

```
// fetch with callbacks
fetch("https://cederdorff.github.io/web-frontend/canvas-users/data.json")
    .then(function (response) {
        return response.json();
    })
    .then(function (data) {
        console.log(data);
    });
//
//
// or with async/await
const response = await fetch("https://cederdorff.github.io/web-frontend/canvas-users/data.json");
const data = await response.json();
console.log(data);
```

will you be async to my await?



async & await

- Use `await` to tell JS to wait for a fetch call to finish and to wait for JSON to parse.
- When using `await` you must tell JS that inside of the function goes some asynchronous code by wrapping it in an `async` function.

```
async function getPosts() {
    const url = "https://raw.githubuserconten
    const response = await fetch(url);
    const data = await response.json();
    setPosts(data);
}
getPosts();
```

"`async` and `await` makes promises easier to write”

`async` makes a function return a Promise

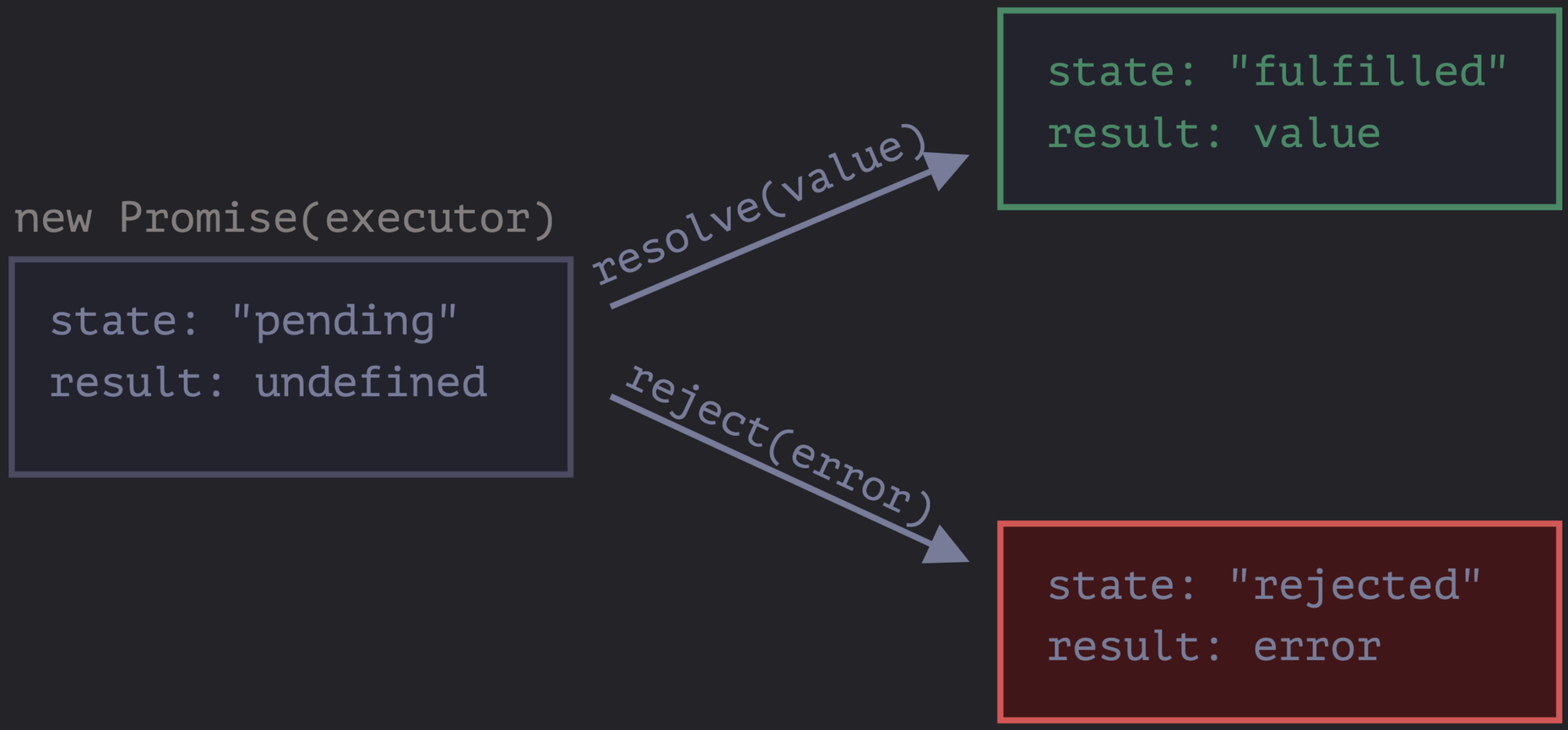
`await` makes a function wait for a Promise

https://www.w3schools.com/js/js_async.asp

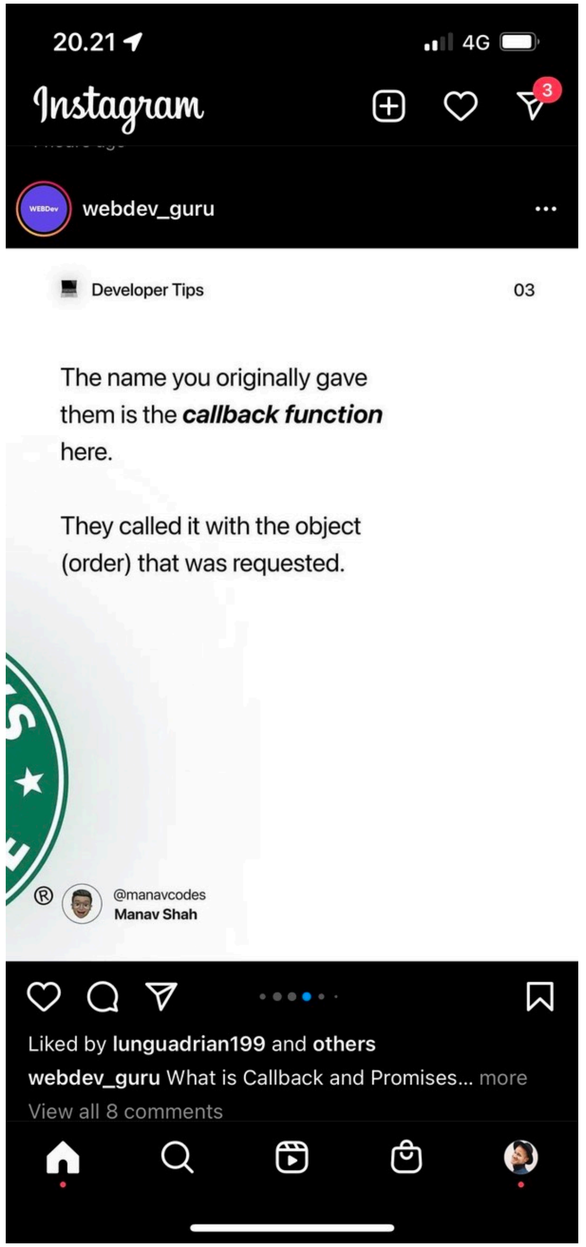
Fetch returns a promise

We can use `await` to wait for fetch to finish 🤷

https://www.w3schools.com/js/js_async.asp



https://javascript.info/promise-basics



https://www.instagram.com/p/CVQGdNIjWMx/

```
let myPromise = new Promise(function (resolve, reject) {
  // "Producing Code" (May take some time)
  // Making coffee ...
  resolve("Yaaay, here's your coffee"); // when successful
  reject("Ohh, f***. Something went wrong");  // when error
});

// "Consuming Code" (Must wait for a fulfilled Promise)
myPromise.then(
  function (value) {
    /* code if successful */
    console.log(value); // Yaaay, here's your coffee
  },
  function (error) {
    /* code if some error */
    console.log(error); // Ohh, f***. Something went wrong
  }
);
```

https://www.w3schools.com/js/js_promise.asp

All you need to know

- Use `await` to tell JS to wait for a fetch call to finish.
- When using `await` you must tell JS that here goes some asynchronous code by wrapping it in an `async` function.

```
function PostsPage() {
    const [posts, setPosts] = useState([]);

    useEffect(() => {
        async function getPosts() {
            const url = "https://raw.githubusercontent.co
            const response = await fetch(url);
            const data = await response.json();
            setPosts(data);
        }
        getPosts();
    }, []);

    return (
        <section className="page">
            <h1>Posts</h1>
            <section className="grid-container">
                {posts.map(post => (
                    <PostItem post={post} key={post.id} /
```

JSON

Java**S**cript **O**bject **N**otation

JSON

Java**S**cript **O**bject **N**otation

... a syntax for storing & exchanging data over the web

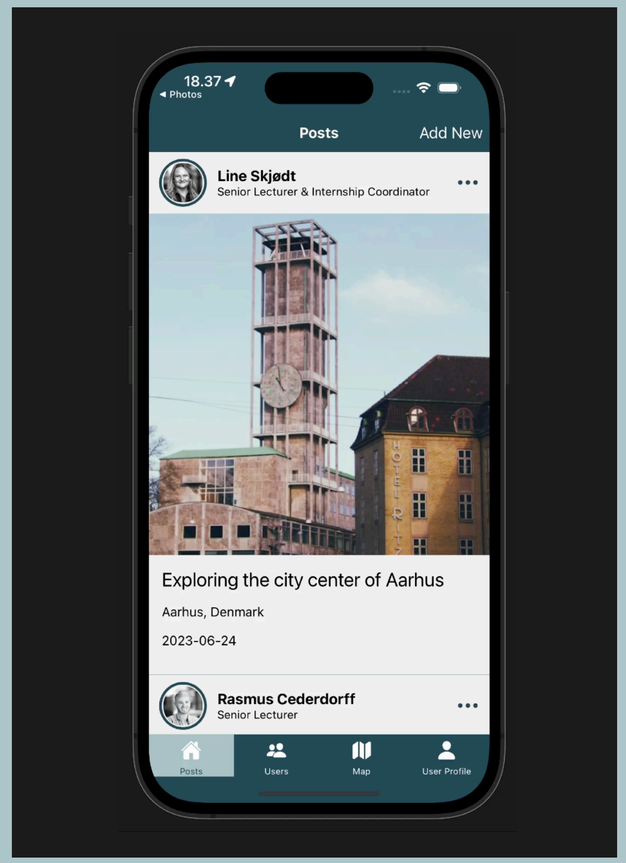
https://www.instagram.com/p/CVqbCzgsZUF/

JSON (& API) is the glue

APP

API

DATA SOURCE



FETCH(GET, POST, PUT, DELETE)

REQUEST

JSON DATA

RESPONSE

CLIENT

SERVER

JSON

... a syntax for storing and exchanging data over the web

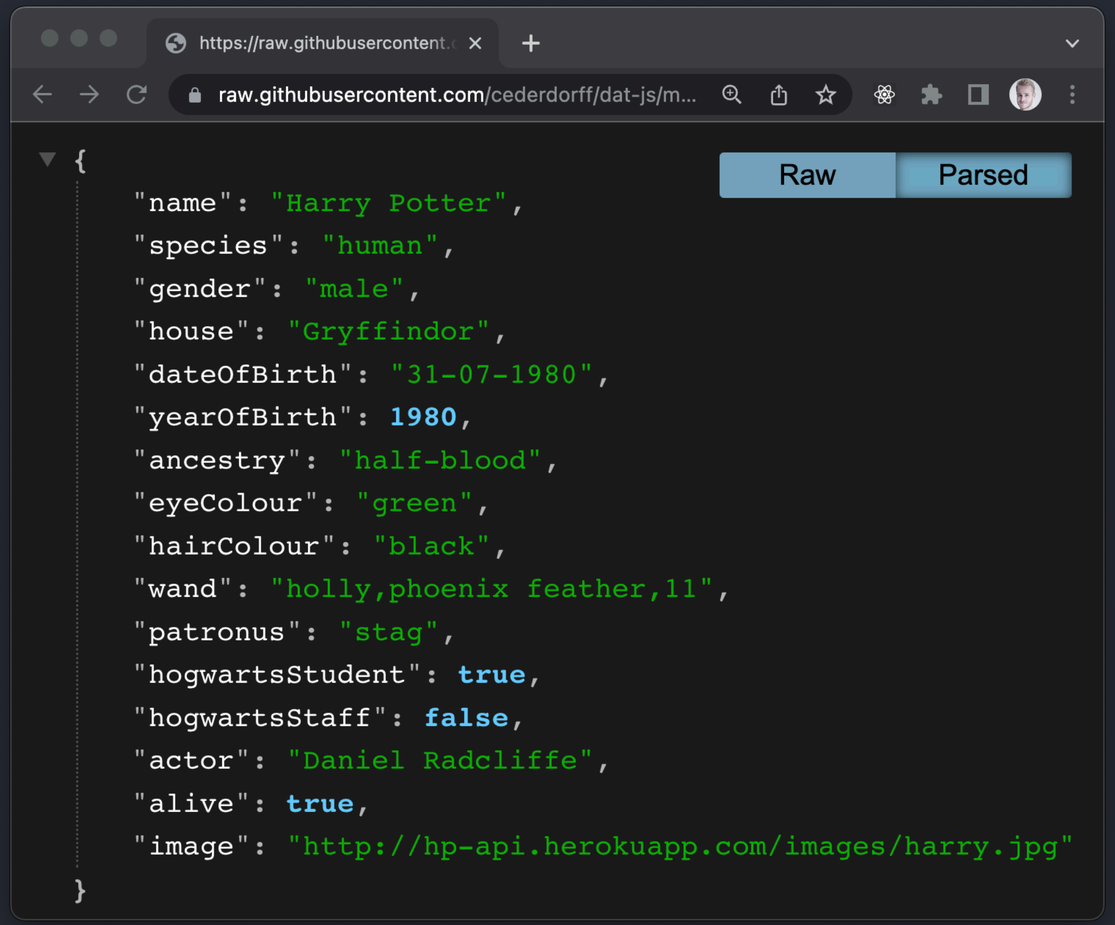
```
{
  "name": "Alicia",
  "age": 6
}
```

JSON OBJECT

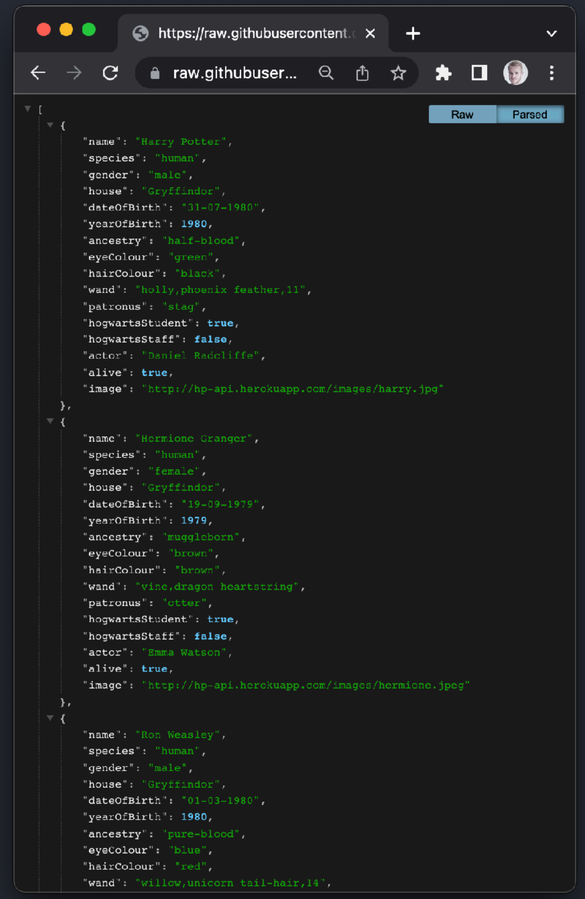
```
[{
  "name": "Alicia",
  "age": 6
}, {
  "name": "Peter",
  "age": 22
}]
```

LIST OF JSON OBJECTS

JSON

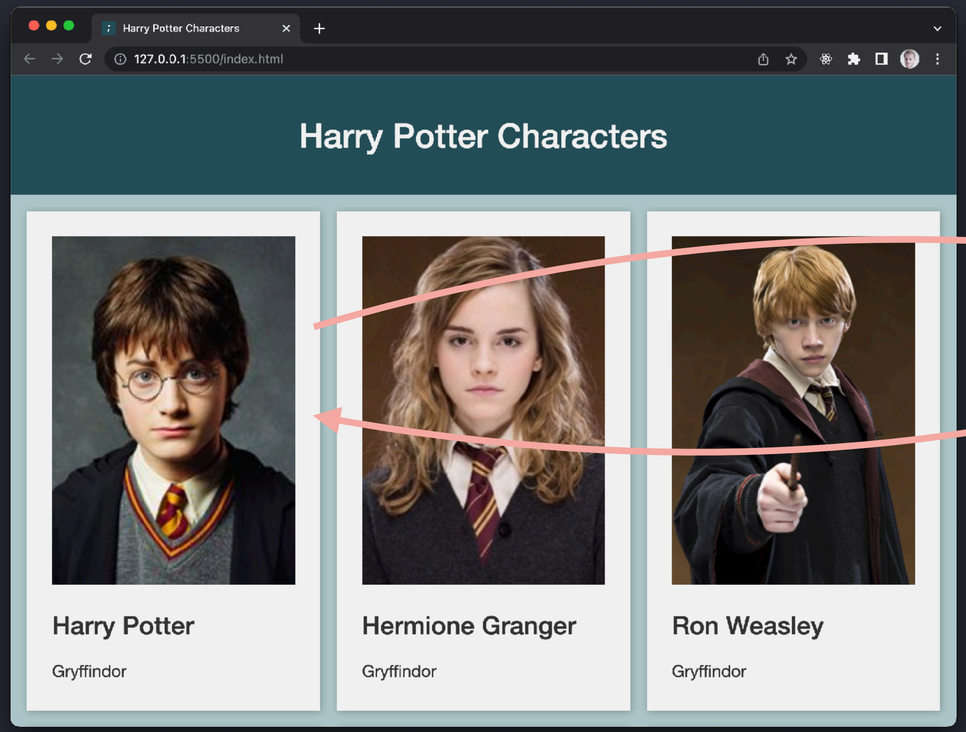


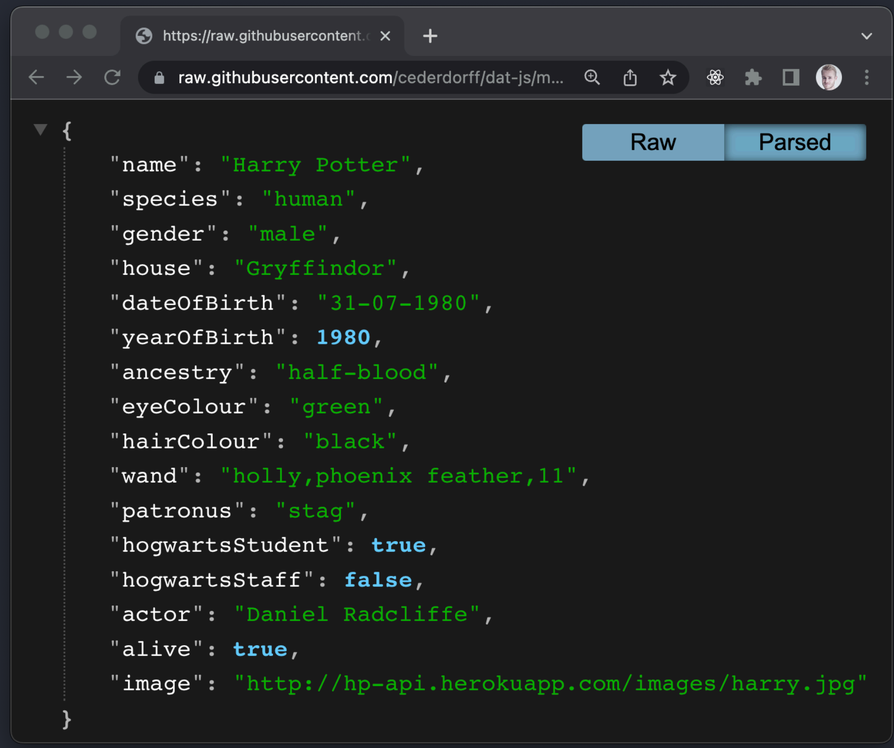
JSON OBJECT



LIST OF JSON OBJECTS

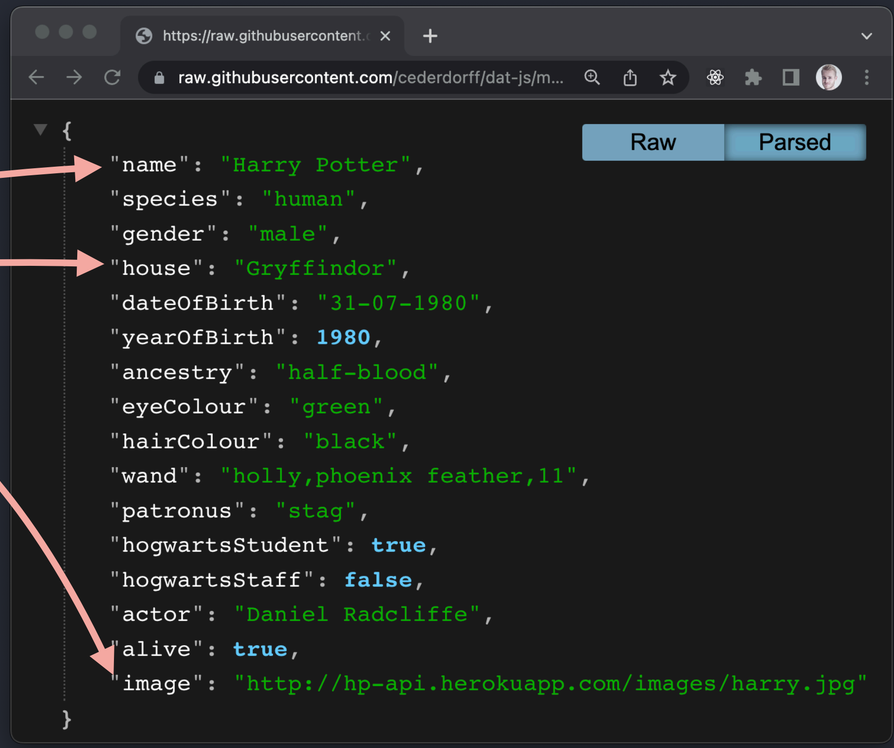
JSON Object





https://raw.githubusercontent.com/cederdorff/dat-js/main/data/harry.json

JSON Object



https://raw.githubusercontent.com/cederdorff/dat-js/main/data/harry.json

JAVASCRIPT OBJECT NOTATION

- Collection of key-value pair: “`key”: “value”`
- List of values, collections or objects
- Lightweight data-interchange format
- Syntax / text format for storing and exchanging data over the web
- Human and machine readable **text**: small, fast and simple
- Language independent
- Can be parsed directly to JavaScript Object
- JavaScript Objects can be converted directly to JSON
- The glue between programs (interface between frontend and backend)


```
[
    {
        "id": "1",
        "firstname": "Kasper",
        "lastname": "Topp",
        "age": "34",
        "haircolor": "Dark Blonde",
        "countryName": "Denmark",
        "gender": "Male",
        "lookingFor": "Female"
    },
    {
        "id": "2",
        "firstname": "Nicklas",
        "lastname": "Andersen",
        "age": "22",
        "haircolor": "Brown",
        "countryName": "Denmark",
        "gender": "Male",
        "lookingFor": "Female"
    },
    {
        "id": "3",
        "firstname": " Sarah",
        "lastname": "Dybvad ",
        "age": "34",
        "haircolor": "Blonde",
        "countryName": "Denmark",
        "gender": "Female",
        "lookingFor": "Male"
    },
    {
        "id": "4",
        "firstname": "Alex",
```

JSON METHODS

```
const user = {
  name: "John",
  age: 30,
  gender: "male",
  lookingFor: "female"
};

// === JSON.stringify === //
const jsonUser = JSON.stringify(user);
console.log(jsonUser); // {"name":"John","age":30,"gender":"male","lookingFor":"female"}

// === JSON.parse === //
const jsonString = '{"name":"John","age":30,"gender":"male","lookingFor":"female"}';
const userObject = JSON.parse(jsonString);
console.log(userObject); // logging userObject
```

json-parse-stringify

CRUD App

127.0.0.1:5500/index.html

EAAA | Canvas | KEA | BOOKS | Code & Templates | Summer School -...

CRUD App

Rasmus Cederdorff

Senior Lecturer
race@dev.dk

UPDATE DELETE

Lars Bogetoft

Head of Education
larb@eaaa.dk

UPDATE DELETE

Edith Terte

Lecturer
edan@kea.dk

UPDATE DELETE

Frederikke Bender

Head of Education
fbe@kea.dk

UPDATE DELETE

Murat Kilic

Senior Lecturer
mki@eaaa.dk

UPDATE DELETE

Anne Andersen

Head of Education
anki@mail.dk

UPDATE DELETE

Elements | Console | Sources | Network | Performance | Memory | Application

Preserve log | Disable cache | No throttling

Filter | Invert | Hide data URLs | Hide extension URLs

All | Doc | JS | Fetch/XHR | CSS | Font | Img | Media | Manifest | WS | Wasm | Other | Blocked response cookies

Blocked requests | 3rd-party requests

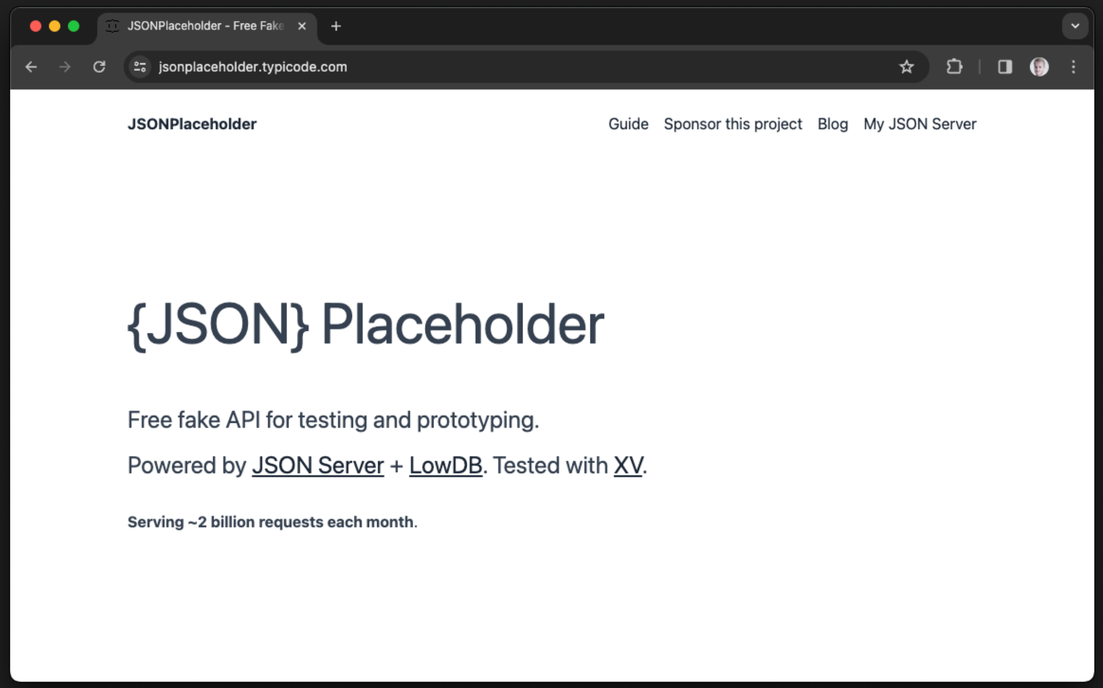
10 ms | 20 ms | 30 ms | 40 ms | 50 ms | 60 ms | 70 ms | 80 ms | 90 ms | 100 ms | 110

Name | Headers | Preview | Response | Initiator | Timing

users


```
[
    {
        "id": 2,
        "name": "Rasmus Cederdorff",
        "mail": "race@dev.dk",
        "title": "Senior Lecturer",
        "image": "https://share.cederdorff.com/images/race.jpg"
    },
    {
        "id": 3,
        "name": "Lars Bogetoft",
        "mail": "larb@eaaa.dk",
        "title": "Head of Education",
        "image": "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/larb.jpg"
    },
    {
        "id": 4,
        "name": "Edith Terte",
        "mail": "edan@kea.dk",
        "title": "Lecturer",
        "image": "https://kea.dk/slir/w180-c1x1/images/user-profile/synced/EDAN.jpg"
    },
    {
        "id": 5,
        "name": "Frederikke Bender",
        "mail": "fbe@kea.dk",
        "title": "Head of Education",
        "image": "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/fbe.jpg"
    },
    {
        "id": 6,
        "name": "Murat Kilic",
        "mail": "mki@eaaa.dk",
        "title": "Senior Lecturer",
        "image": "https://www.eaaa.dk/media/llyavasj/murat-kilic.jpg?width=800&heigh
    },
    {
        "id": 7,
        "name": "Anne Andersen",
        "mail": "anki@mail.dk",
        "title": "Head of Education",
        "image": "https://www.eaaa.dk/media/5buh1xeo/anne-kirketerp.jpg?width=800&he
    }
]
```


1 / 11 requests | 21



https://jsonplaceholder.typicode.com/

Test endpoints

- Install JSON Formatter
- Test the following endpoints:
  - https://jsonplaceholder.typicode.com/posts/
  - https://jsonplaceholder.typicode.com/posts/5
  - https://jsonplaceholder.typicode.com/users/
  - https://jsonplaceholder.typicode.com/users/1
  - https://jsonplaceholder.typicode.com/todos
  - https://jsonplaceholder.typicode.com/todos/5

```json
{
  "name": "Rasmus Cederdorff",
  "birthday": "1990-03-12",
  "title": "Experienced JavaScript Developer",
  "experienceYears": 10,
  "education": {
    "degree": "Master of Science in IT - Web Communication",
    "specialization": "Web Architecture"
  },
  "skills": ["JavaScript", "React", "Node.js", "UI/UX Design"],
  "currentPosition": "Senior Lecturer at EAAA",
  "teachingSubjects": ["Web Development", "JavaScript", "React", "BaaS & Node.js"],
  "lovesJavaScript": true
}
```

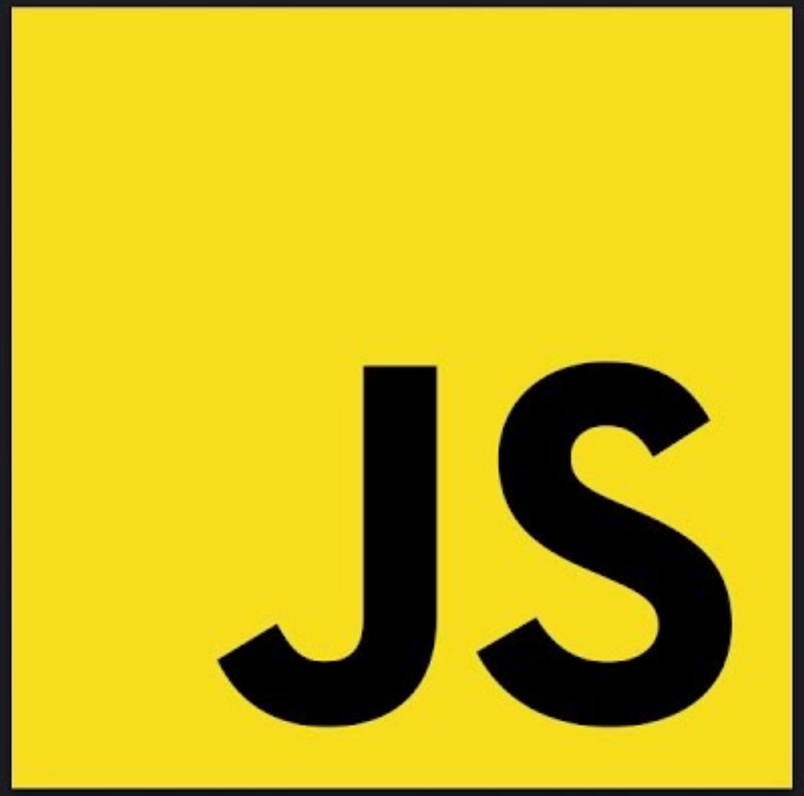
Define yourself with JSON

- https://race.notion.site/Define-yourself-with-JSON-391944b6b1a041bfa341a0d46f5c0e4a?pvs=4

DOM Manipulation

Application Programming Interface

100 SECONDS OF



https://www.youtube.com/watch?v=DHjqpvDnNGE

“The cool thing about JavaScript is that you can create stuff that will put a smile on your face, as a developer.

You can create stuff and it will visually look like something, and it'll do stuff, and it makes you feel good, **it makes you fall in love with programming.**”

Lex Fridman

Computer Scientist

https://www.youtube.com/watch?v=GLhyjVZp0cw&t=252s

index.html

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>This is a paragraph.</p>
    <button onclick="tryMe()">Try me</button>
    <script src="app.js"></script>
  </body>
</html>
```

app.js

```js
function tryMe() {
  document.body.style.backgroundColor = "red";
  document.body.style.color = "white";
}
```

What is JavaScript?

.. is the world's most popular programming language.

... is the programming language of the Web.

... is easy to learn.

... can change content of a webpage (HTML content).

... can change styling of HTML.

https://www.w3schools.com/js/default.asp

index.html

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Page Title</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1>This is a Heading</h1>
    <p>This is a paragraph.</p>
    <button onclick="tryMe()">Try me</button>
    <script src="app.js"></script>
  </body>
</html>
```

app.js

```js
function tryMe() {
  document.body.style.backgroundColor = "red";
  document.body.style.color = "white";
}
```

With JavaScript we are able to

... build dynamic web pages and web apps.

... fetch content/ data from a backend (web service, data source, etc.) through an API.

... do DOM-manipulation.

... build and develop anything 🤷

https://www.w3schools.com/js/default.asp

DOM Manipulation

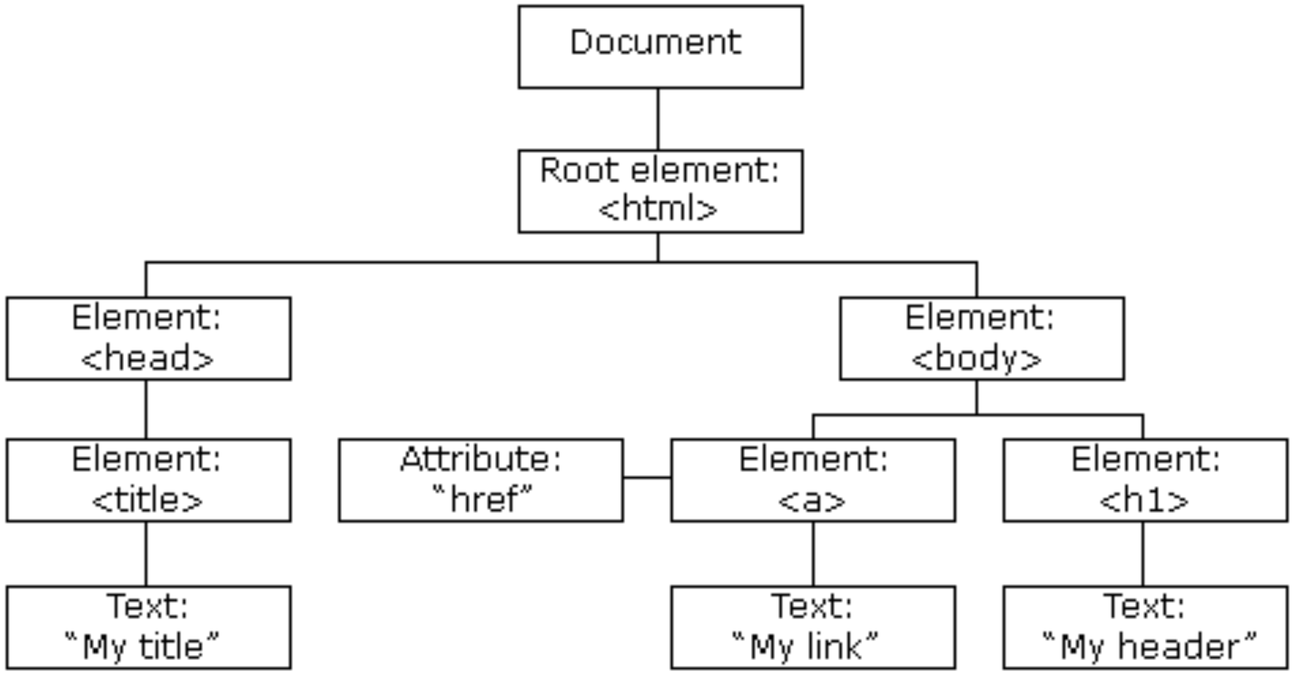
```
// declaring a variable with a value
let message = "Hi Frontenders!"

//accessing the variable and logging it to the console
console.log(message);

// appending the variable (the string) to the DOM element #content
document.querySelector("#content").innerHTML = message;
```

```
<body>
  <header>
    <h1>PROJECT TEMPLATE</h1>
  </header>
  <section id="content"></section>

  <!-- main js file -->
  <script src="js/main.js"></script>
</body>
```

JavaScript HTML DOM

index.html

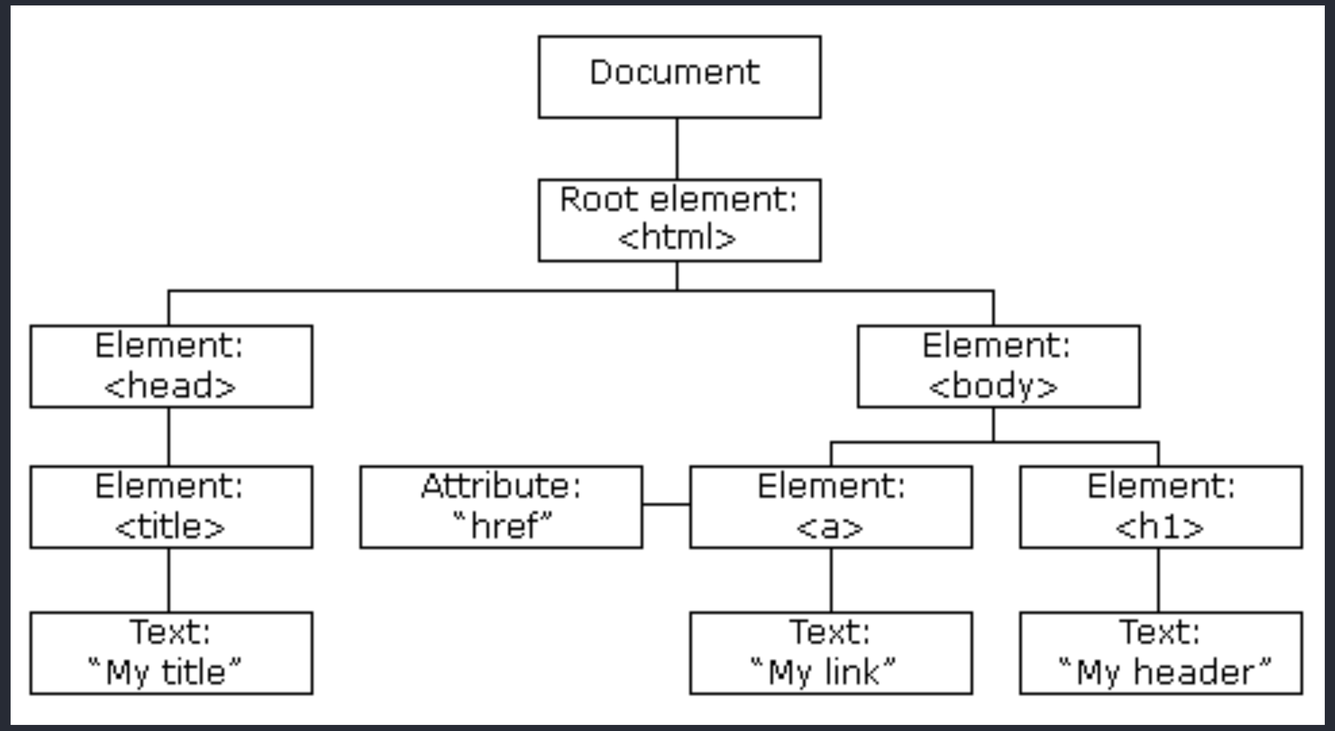
```html
<!DOCTYPE html>
<html>
  <head>
    <title>My title</title>
  </head>

  <body>
    <h1>My header</h1>
    <a href="https://cederdorff.com">My link</a>
  </body>
</html>
```



https://www.w3schools.com/js/js_htmldom.asp
https://javascript.info/dom-nodes
https://javascript.info/dom-navigation

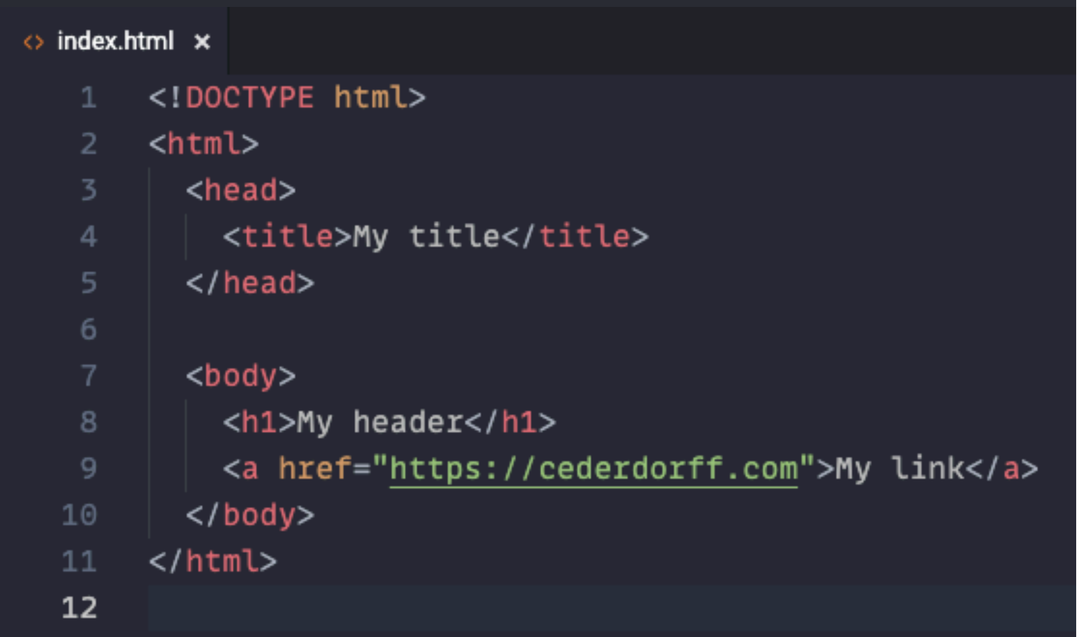
The HTML DOM (Document Object Model)

A model as a tree of Objects



- Object Model for HTML:
  - HTML elements as objects
  - Properties for all HTML elements
  - Methods for all HTML elements
  - Events for all HTML elements

https://www.w3schools.com/whatis/whatis_htmldom.asp



JavaScript HTML DOM

Document Object Model

The HTML document as an object

Gives us the power to create dynamic HTML and manipulate with the HTML (the DOM).

JavaScript can:

… change all the HTML elements in the page
… change all the HTML attributes in the page
… change all the CSS styles in the page
… remove existing HTML elements and attributes
… add new HTML elements and attributes
… react to all existing HTML events in the page
… create new HTML events in the page

https://www.w3schools.com/js/js_htmldom.asp
https://javascript.info/dom-nodes
https://javascript.info/dom-navigation

DOM

Elements

```
1  <html>
2    <head></head>
3    <body>
4      <div id="app">
5        <h1>Develop. Preview.
6  Ship. 🚀</h1>
7      </div>
8      <script type="text/
9  javascript">...</script>
10   </body>
11 </html>
```

SOURCE CODE (HTML)

index.html

```
1  <html>
2    <head></head>
3    <body>
4      <div id="app"></div>
5      <script type="text/
6  javascript">...</script>
7    </body>
8  </html>
9
10
11
```

https://nextjs.org/learn/foundations/from-javascript-to-react/updating-ui-with-javascript

Searching the DOM: `getElement*` & `querySelector*`

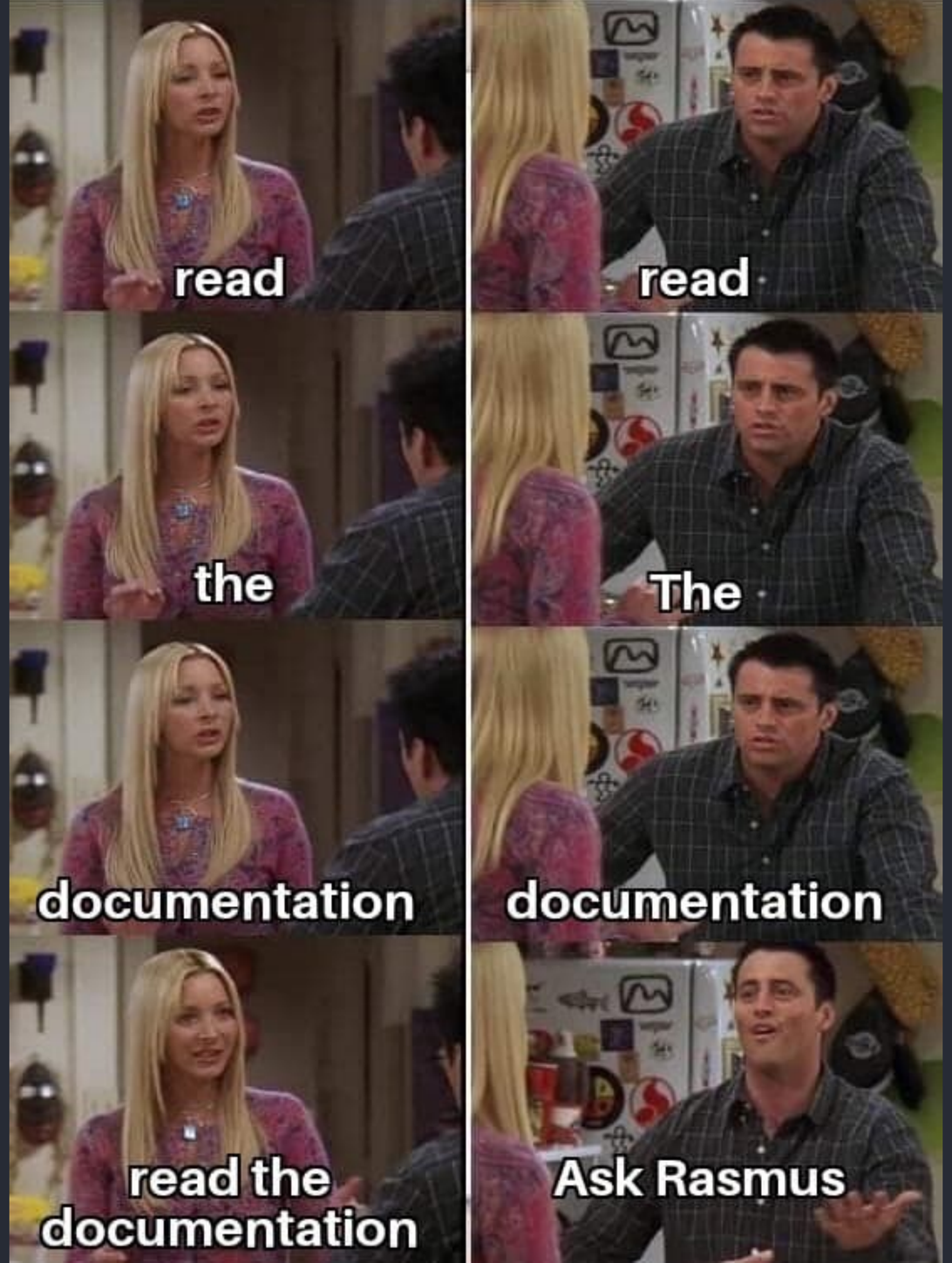
```
<section id="elem">
    <article id="elem-content">Element</article>
</section>

<script>
    // get the element
    const element = document.getElementById('elem');
    // make its background red
    element.style.background = 'red';
    // get the elementContent
    const elementContent = document.querySelector('#elem-content');
    // change inner HTML
    elementContent.innerHTML = "<h2>Hi Web Developers!</h2>"
</script>
```

https://javascript.info/searching-elements-dom

Searching the DOM: `querySelectorAll`

```
<section id="elem">
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
    <article class="elem-content">Element</article>
</section>

<script>
    // get all elements matching the selector - returns an array
    const elements = document.querySelectorAll('.elem-content');
    // loop through all elements
    for (const element of elements) {
        element.innerHTML = "<h2>Hi Web Developers!</h2>";
    }
</script>
```

https://javascript.info/searching-elements-dom


read
read
the
The
documentation
documentation
read the documentation
Ask Rasmus

Headless WordPress

Access your content through WP REST API

What’s a Headless CMS?

A CMS without a head 🤷

I’m not kidding

It’s a CMS without any frontend 💁‍♀️

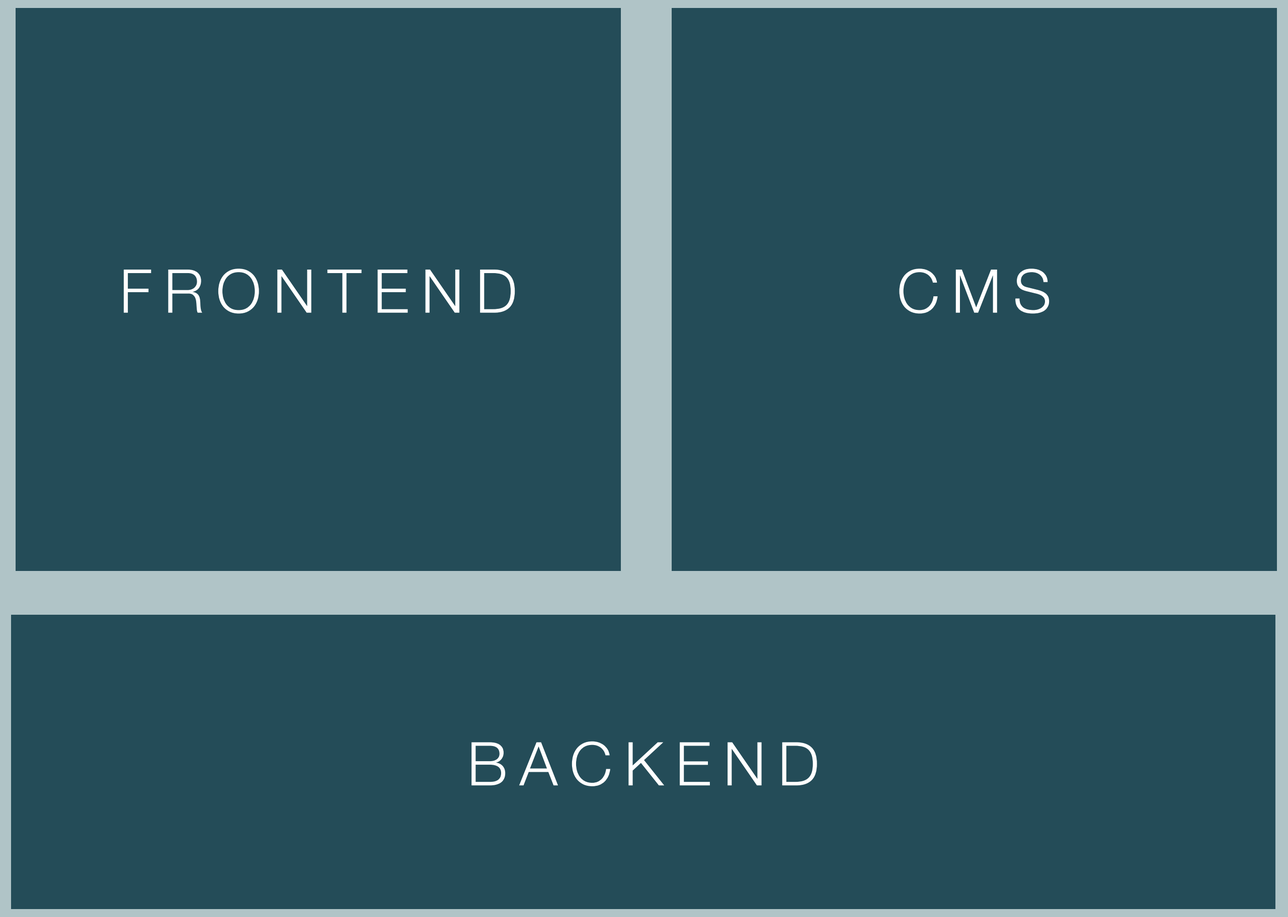
Headless CMS

...is a system to control the content that is separated from the display layer or the front-end user experience.

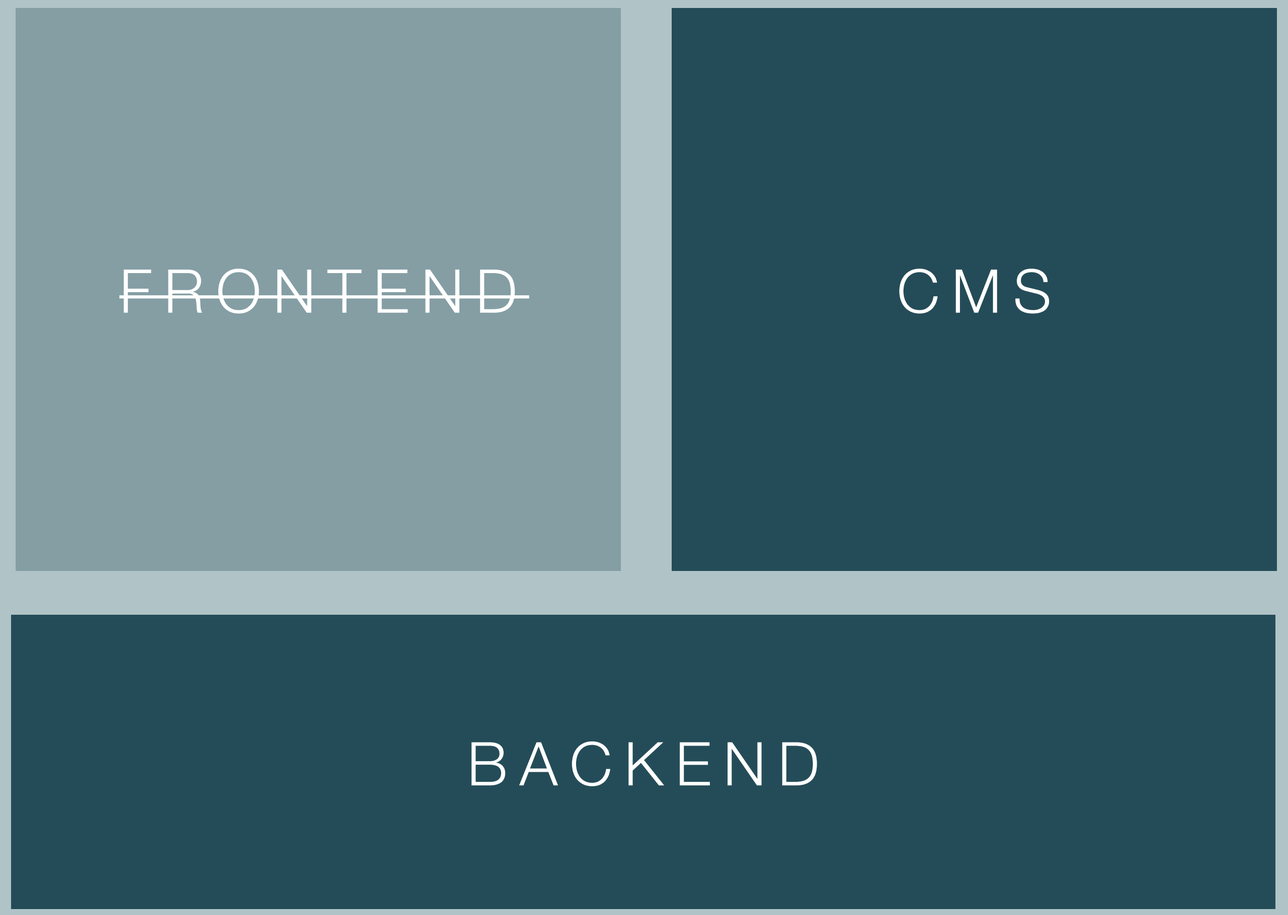
https://medium.com/tech-tajawal/why-headless-cms-is-becoming-so-popular-57d262b1e096

What Is Headless CMS?

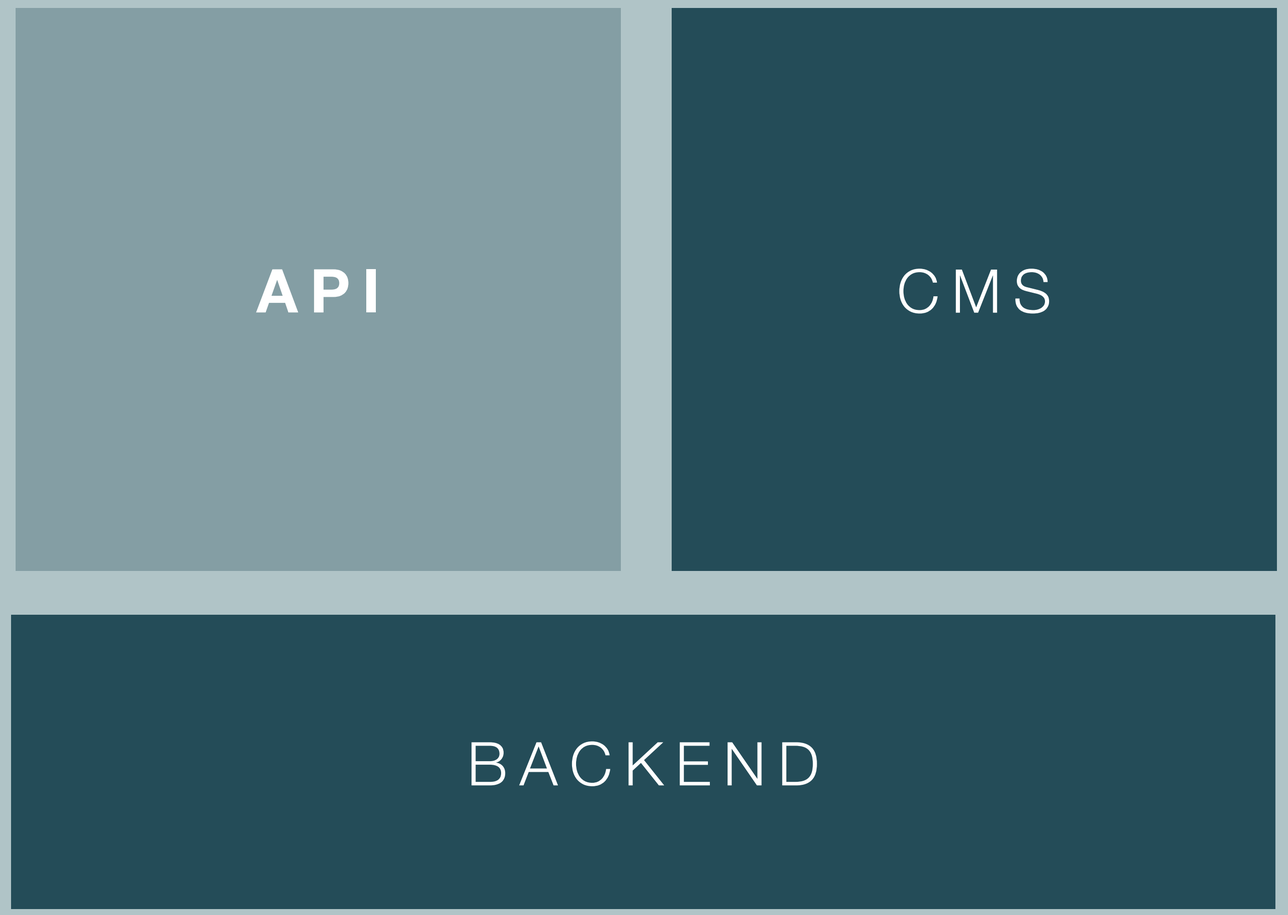
Traditional CMS



Headless CMS



Headless CMS



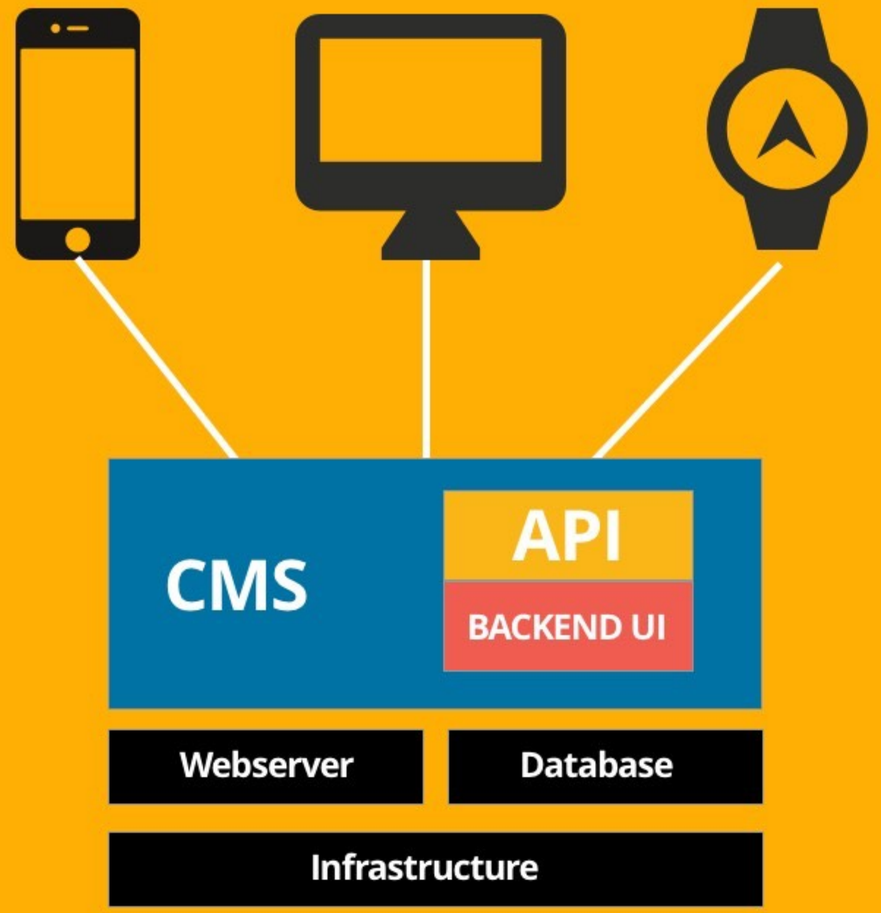
Headless CMS

API

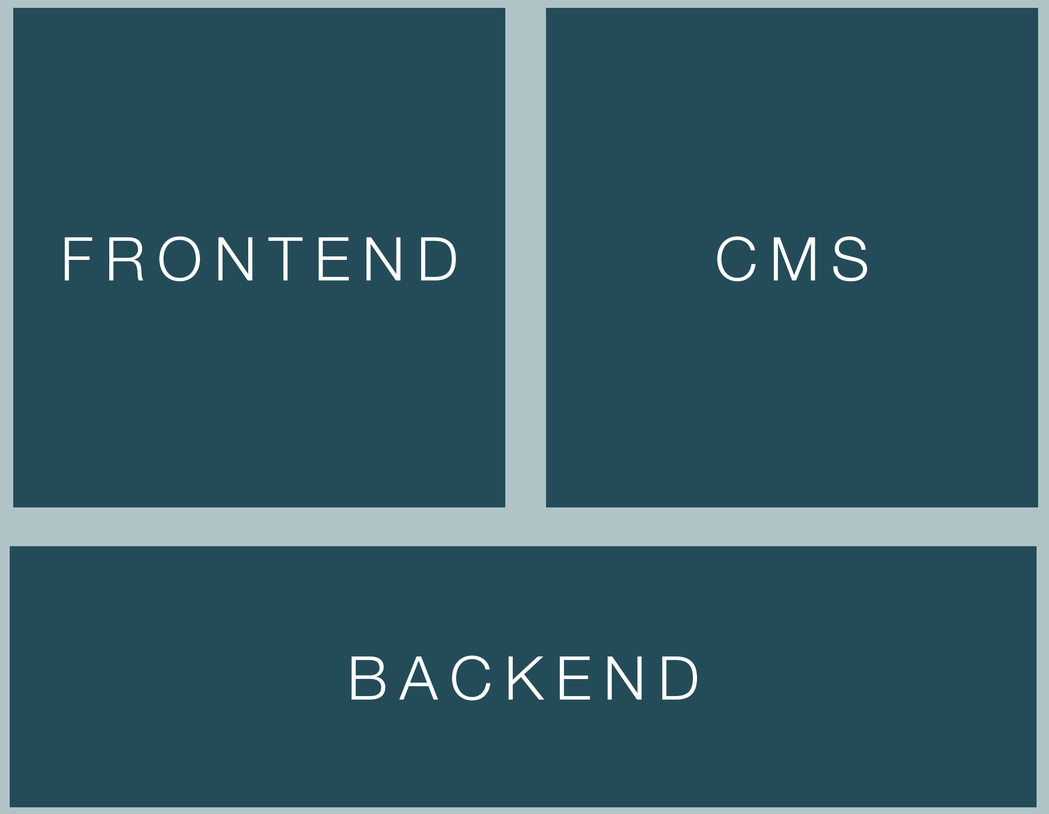
CMS

BACKEND

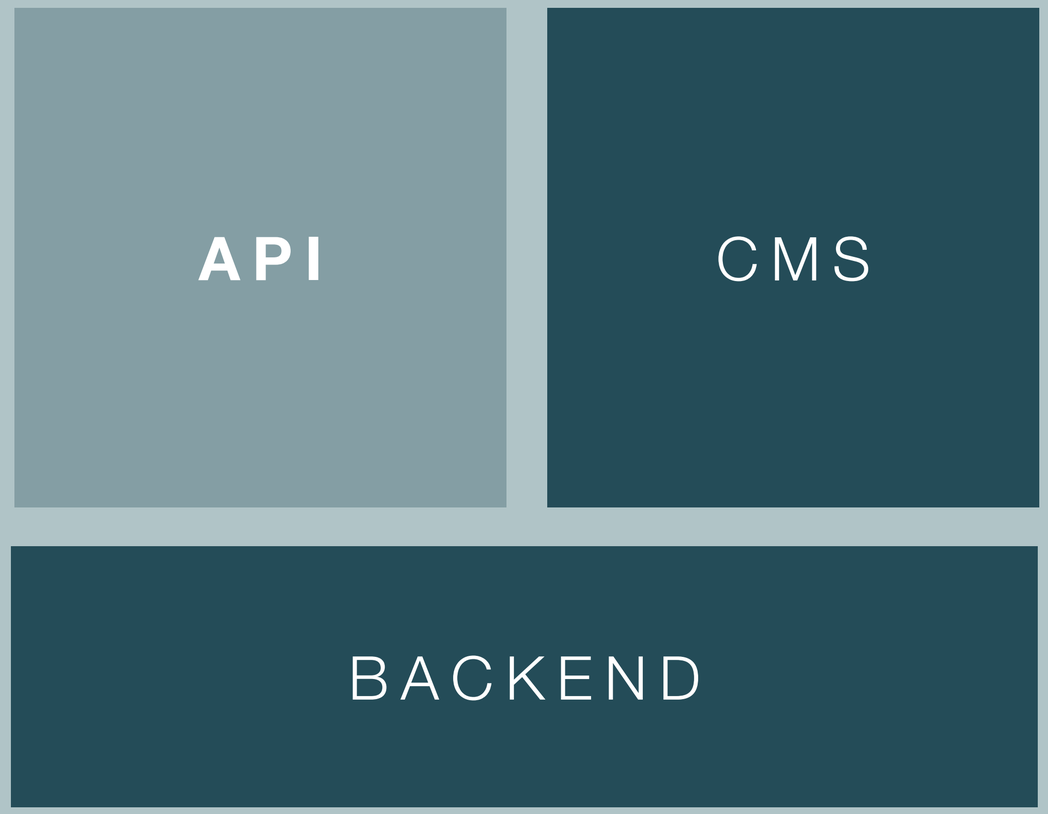
Headless CMS



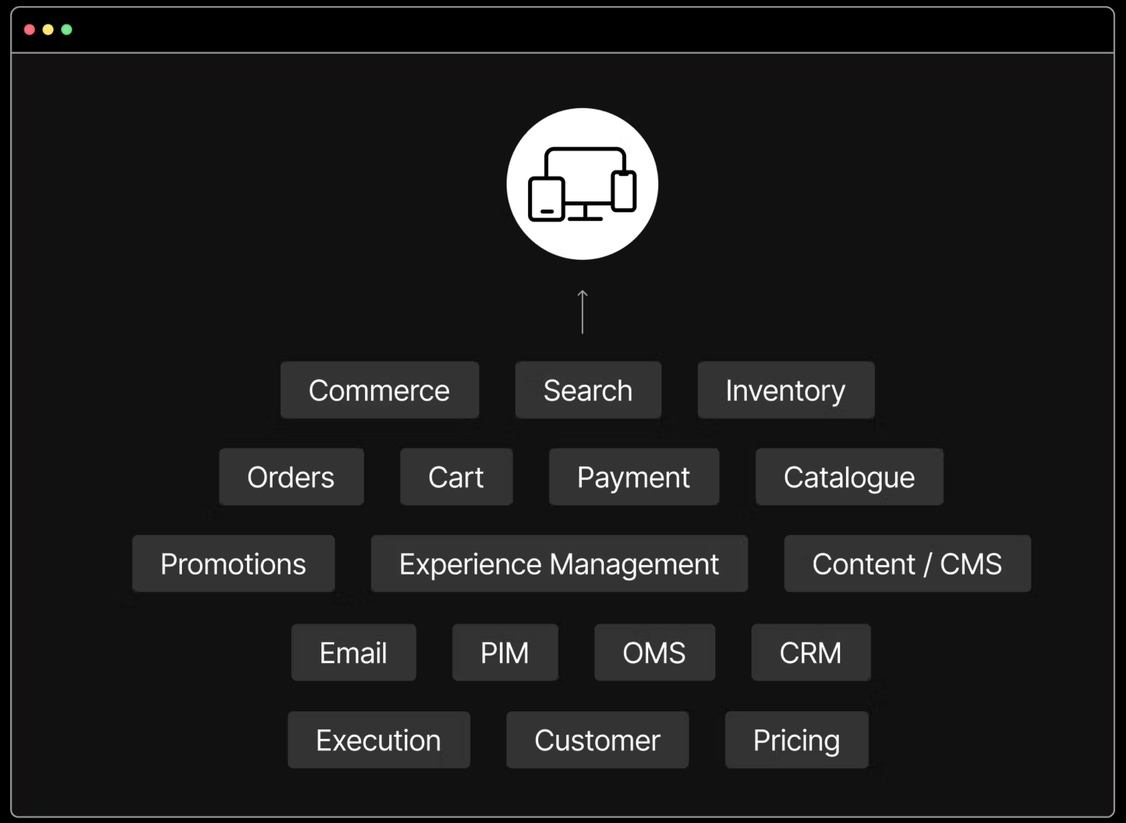
Traditional CMS

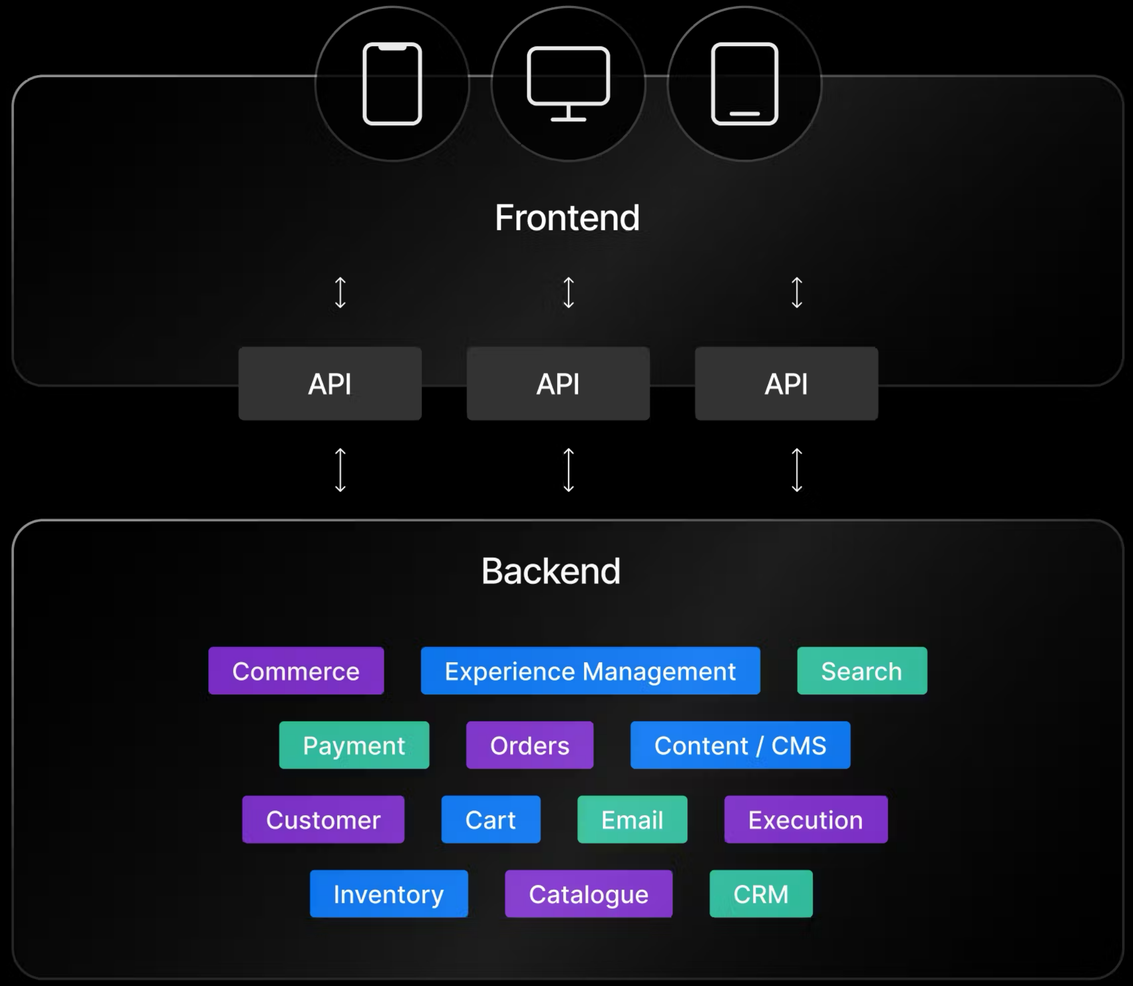


Headless CMS

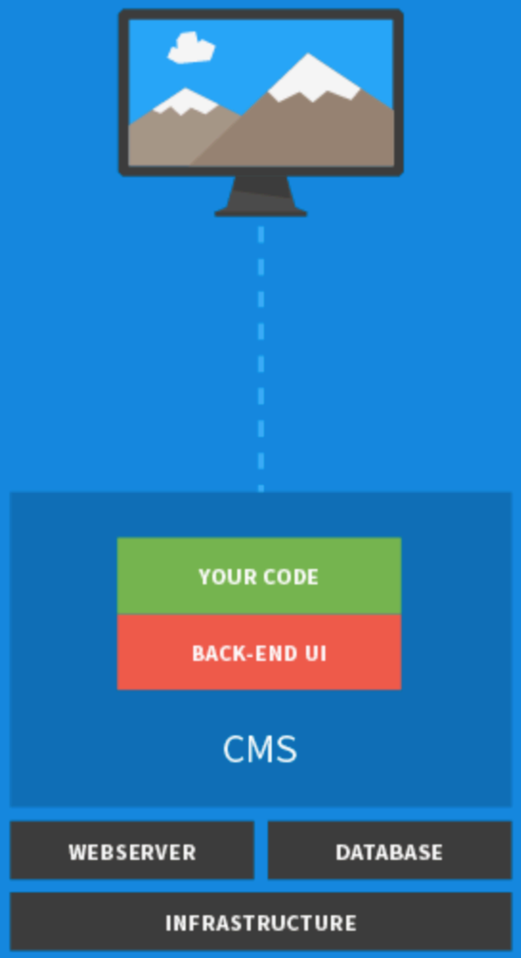


Monolith vs. Headless

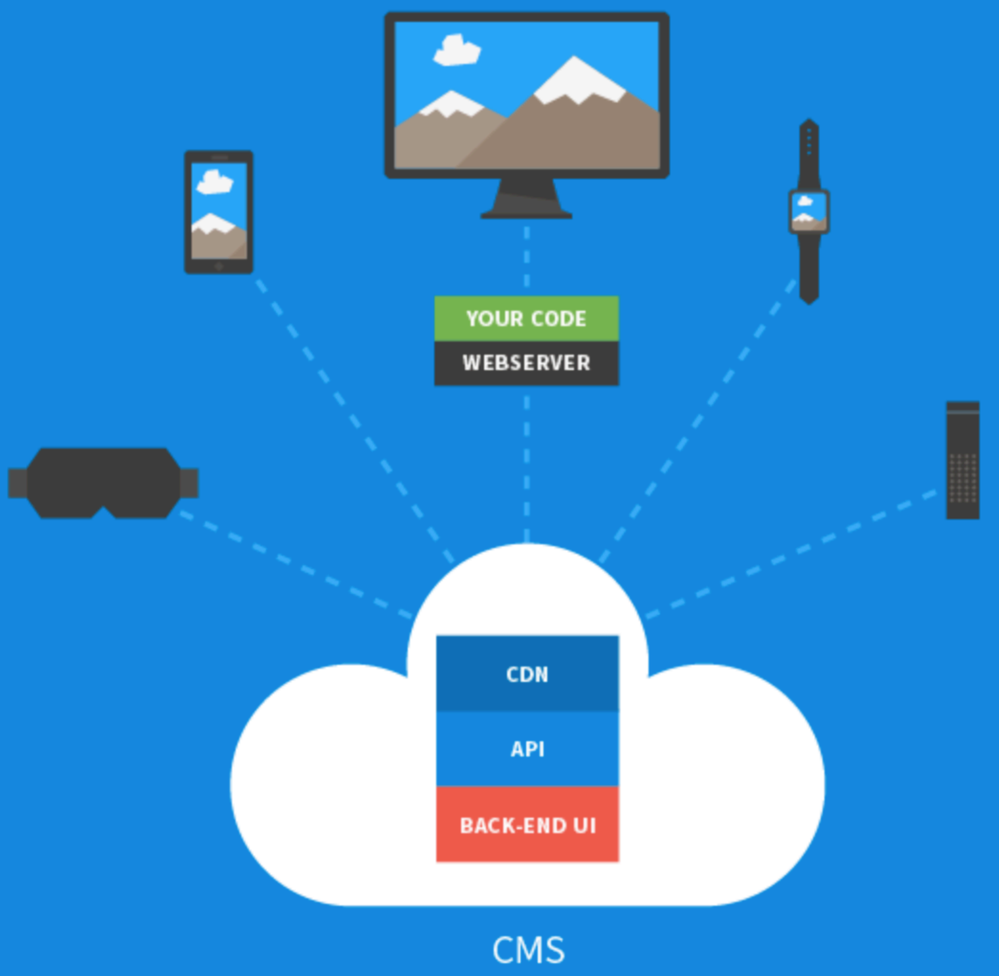




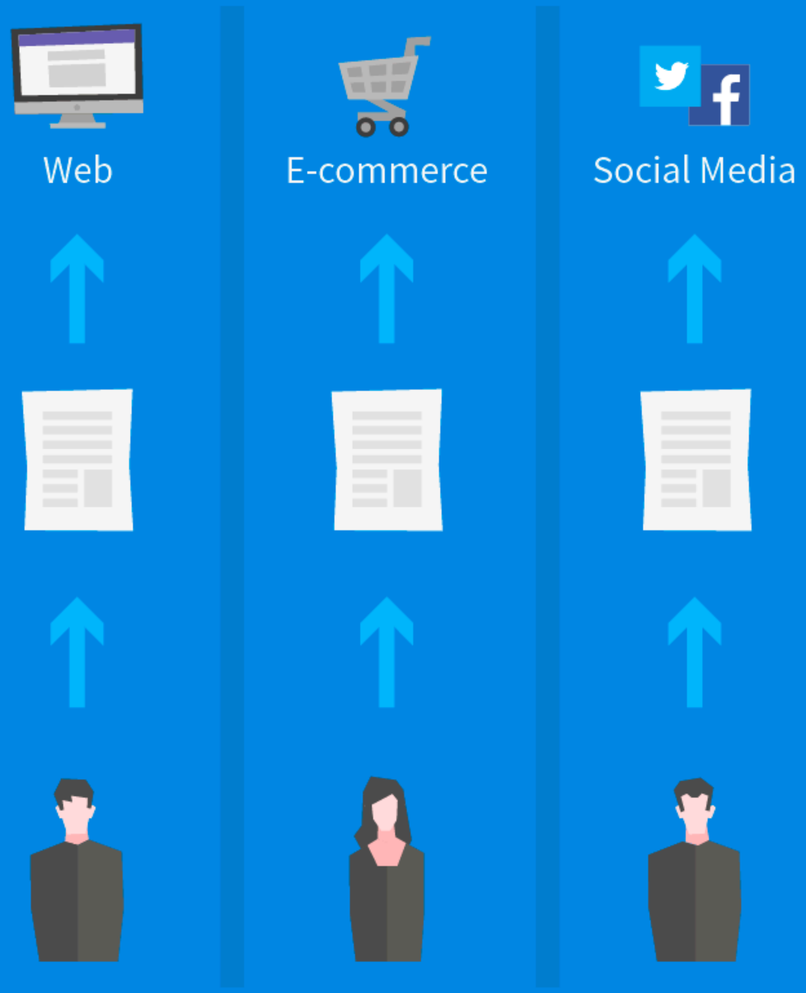
TRADITIONAL CMS



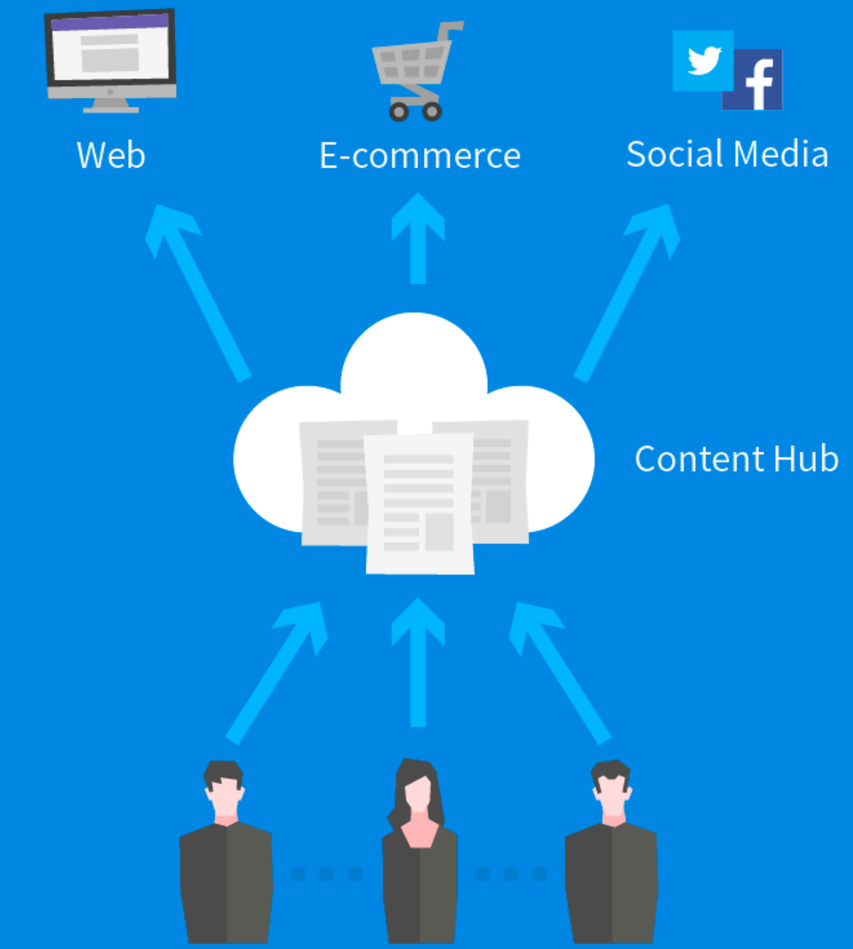
HEADLESS CMS



TRADITIONAL CMS



HEADLESS CMS



TRADITIONAL CMS

A WAY TO STORE DATA

A CRUD UI

A WAY TO DISPLAY THE DATA

HEADLESS CMS

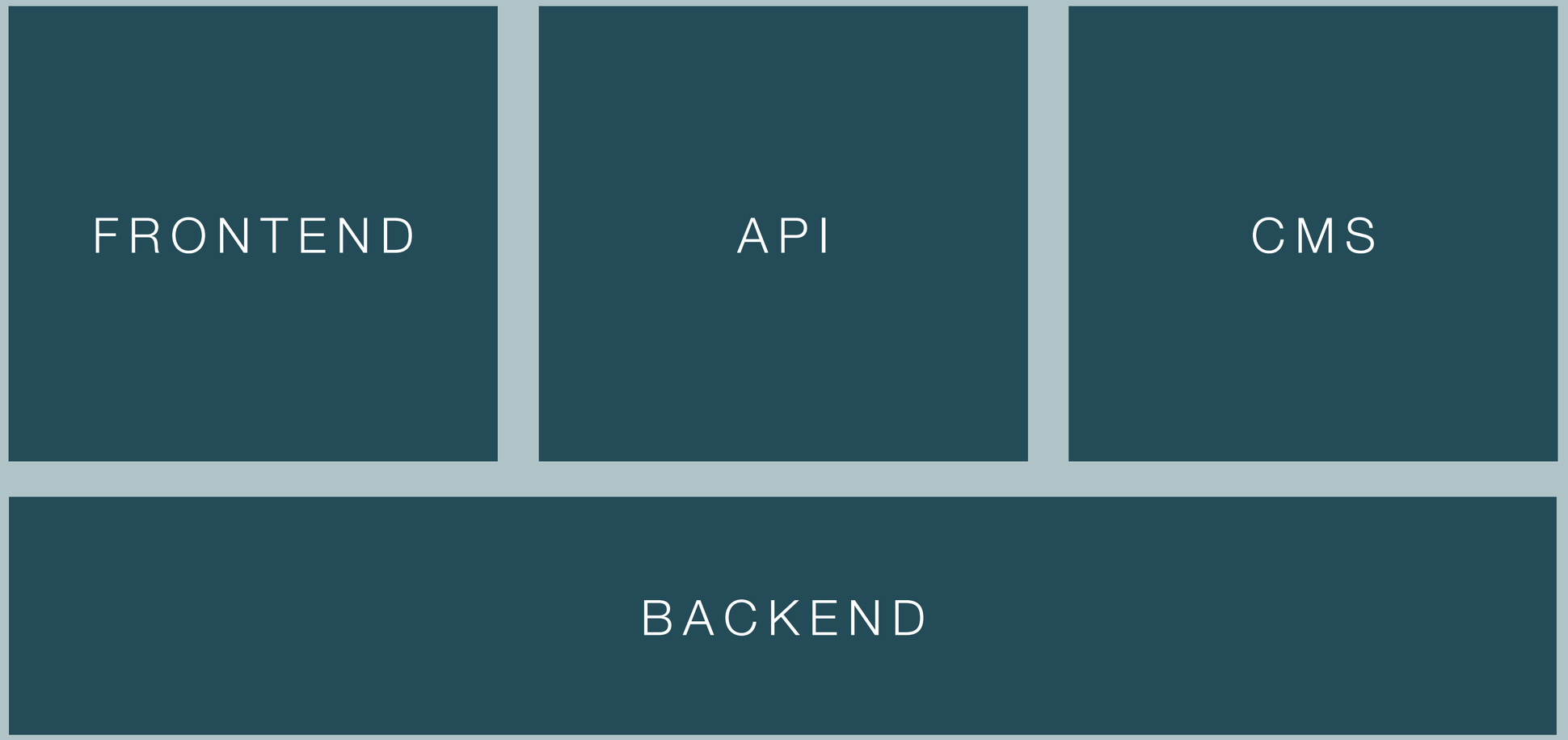
A WAY TO STORE DATA

A CRUD UI

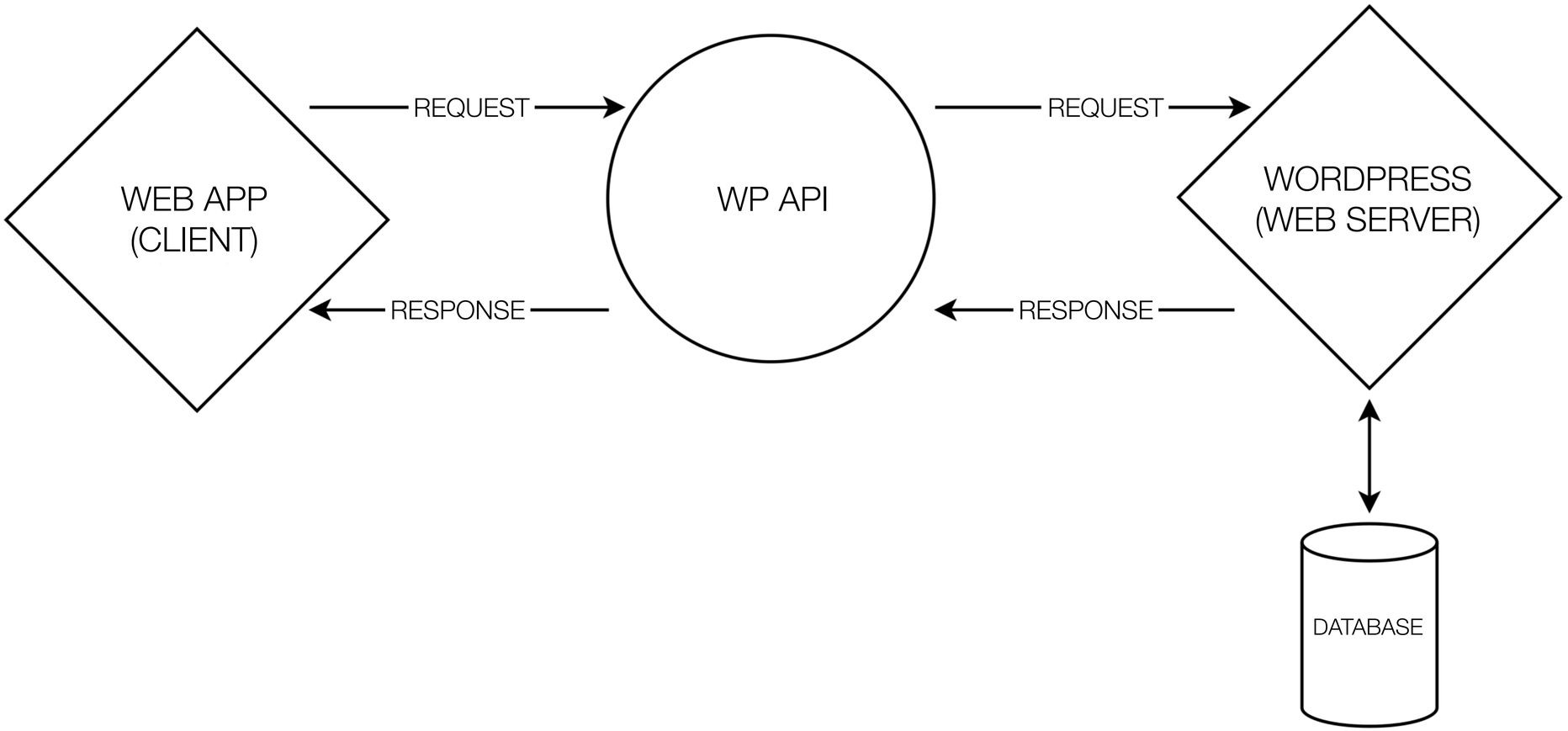
AN API TO ACCESS THE DATA

CRUD: CREATE, READ, UPDATE & DELETE

WordPress



APPLICATION PROGRAMMING INTERFACE

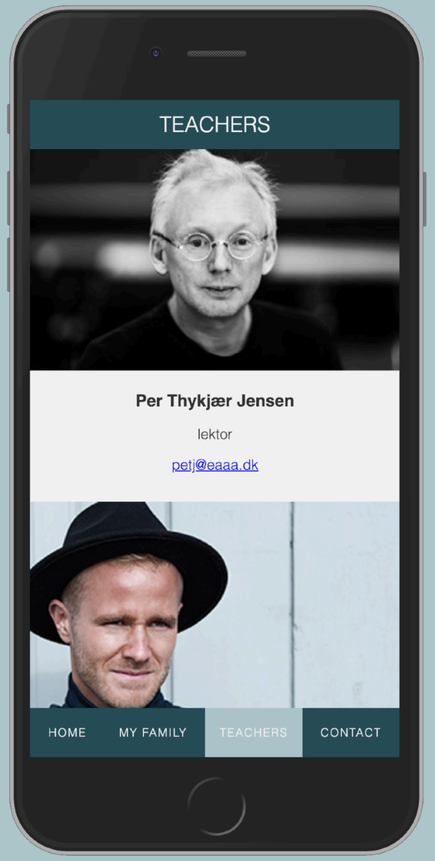


Fetch & Headless CMS

WEB APP

WP API

DATA SOURCE

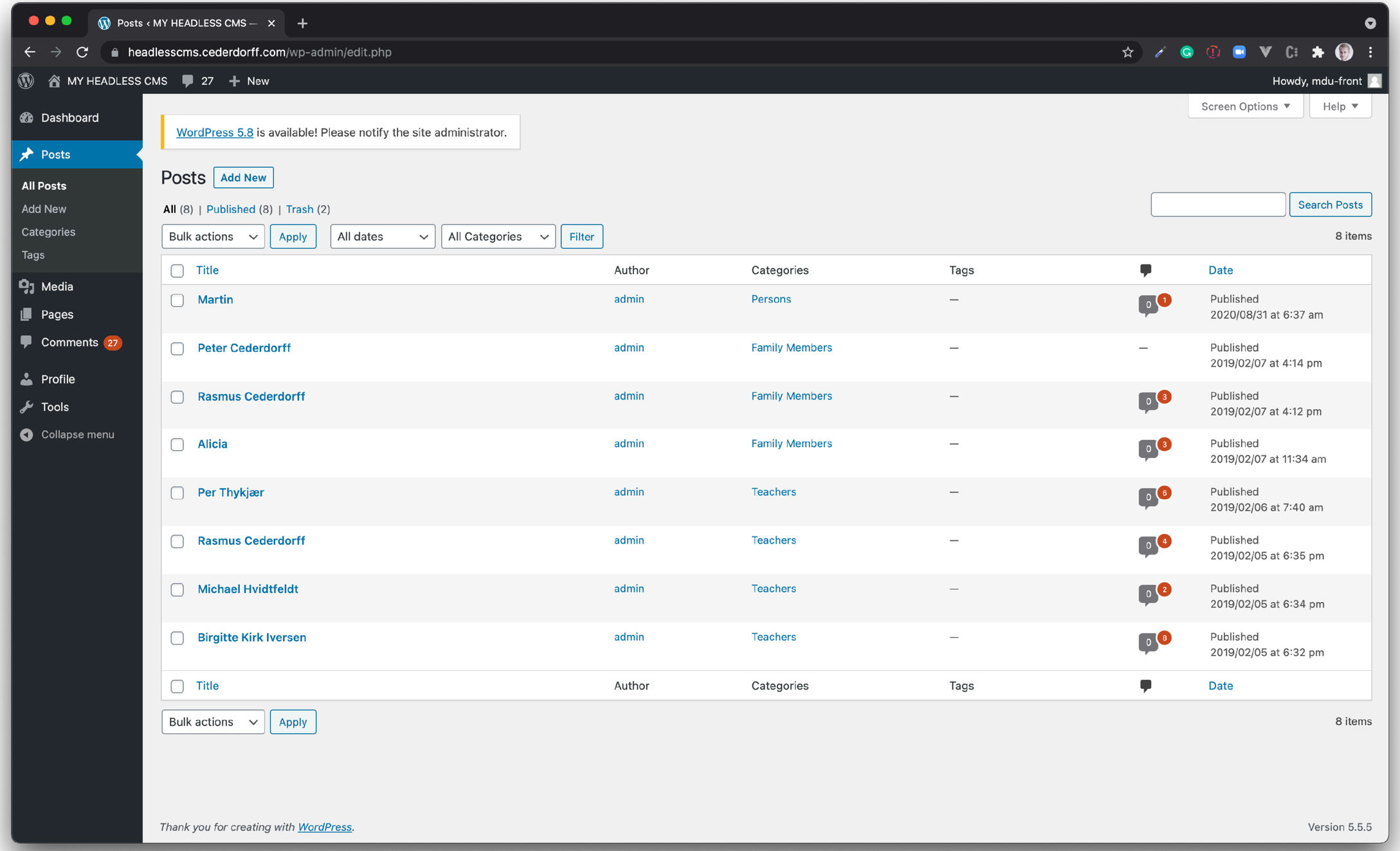


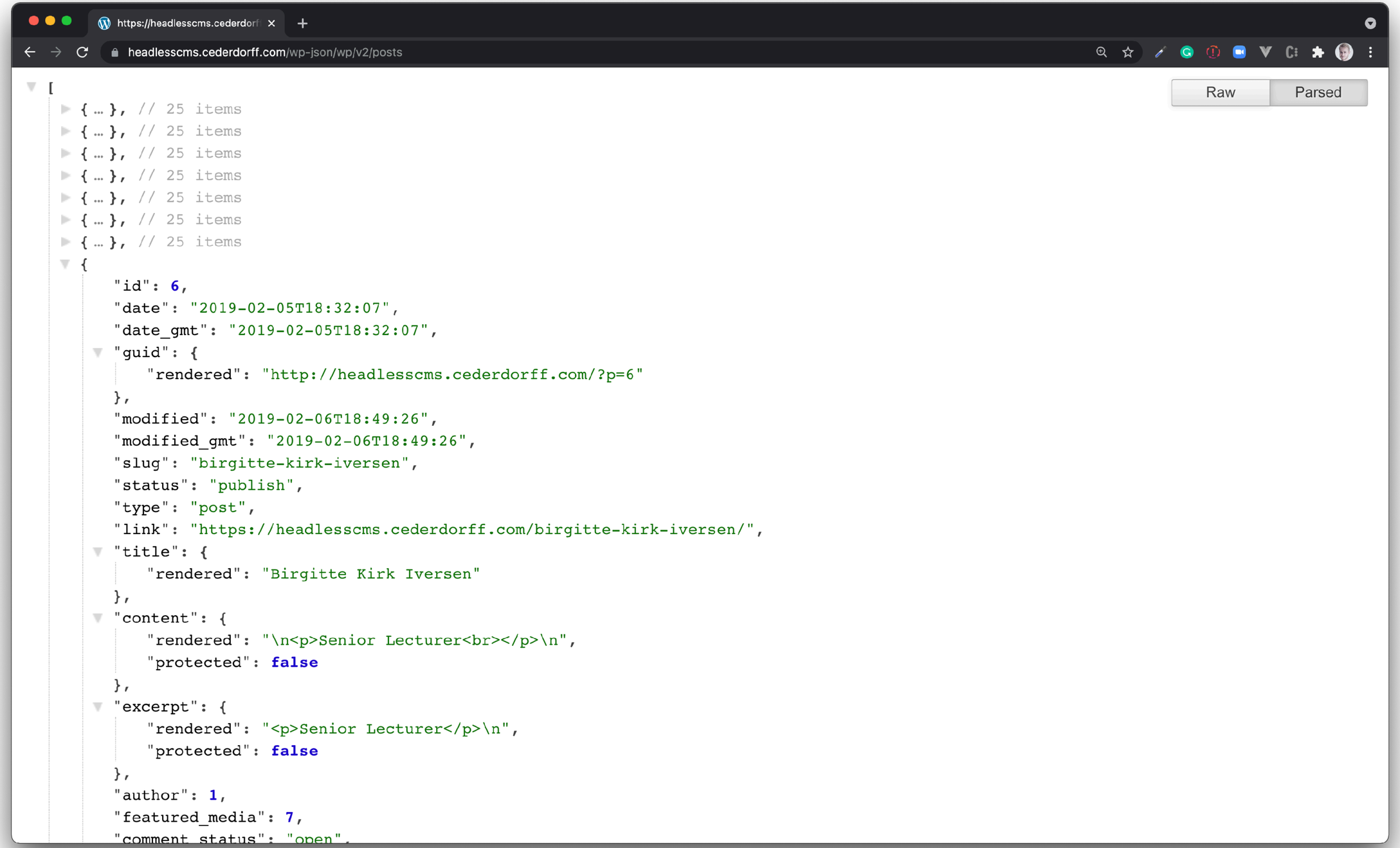
FETCH(“.../wp-json/wp/v2/posts”)

REQUEST

[... posts ...]

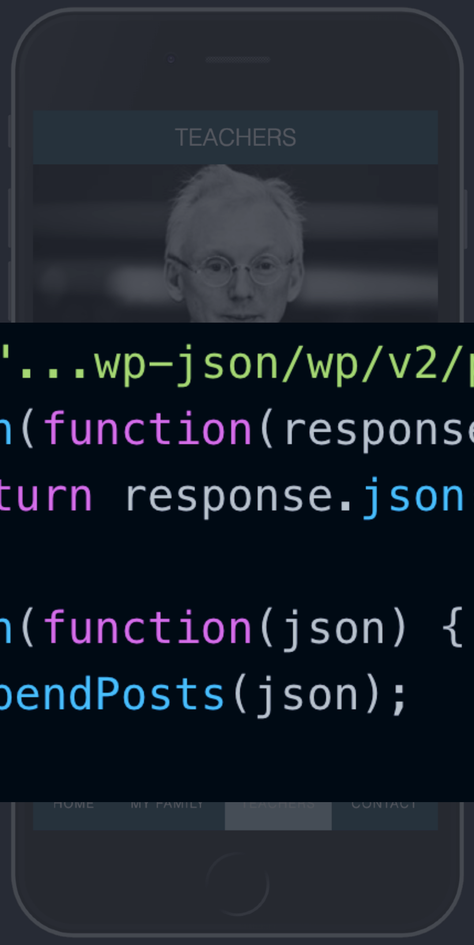
RESPONSE


Posts ‹ MY HEADLESS CMS —
headlesscms.cederdorff.com/wp-admin/edit.php
MY HEADLESS CMS
27
New
Howdy, mdu-front
Dashboard
Posts
All Posts
Add New
Categories
Tags
Media
Pages
Comments 27
Profile
Tools
Collapse menu
Screen Options
Help
WordPress 5.8 is available! Please notify the site administrator.
Posts
Add New
All (8) | Published (8) | Trash (2)
Search Posts
Bulk actions
Apply
All dates
All Categories
Filter
8 items
Title
Author
Categories
Tags
Date
Martin admin Persons — Published 2020/08/31 at 6:37 am
Peter Cederdorff admin Family Members — — Published 2019/02/07 at 4:14 pm
Rasmus Cederdorff admin Family Members — Published 2019/02/07 at 4:12 pm
Alicia admin Family Members — Published 2019/02/07 at 11:34 am
Per Thykjær admin Teachers — Published 2019/02/06 at 7:40 am
Rasmus Cederdorff admin Teachers — Published 2019/02/05 at 6:35 pm
Michael Hvidtfeldt admin Teachers — Published 2019/02/05 at 6:34 pm
Birgitte Kirk Iversen admin Teachers — Published 2019/02/05 at 6:32 pm
Title
Author
Categories
Tags
Date
Bulk actions
Apply
8 items
Thank you for creating with WordPress.
Version 5.5.5


https://headlesscms.cederdorff
headlesscms.cederdorff.com/wp-json/wp/v2/posts
Raw
Parsed
[
{ … }, // 25 items
{ … }, // 25 items
{ … }, // 25 items
{ … }, // 25 items
{ … }, // 25 items
{ … }, // 25 items
{ … }, // 25 items
{
"id": 6,
"date": "2019-02-05T18:32:07",
"date_gmt": "2019-02-05T18:32:07",
"guid": {
"rendered": "http://headlesscms.cederdorff.com/?p=6"
},
"modified": "2019-02-06T18:49:26",
"modified_gmt": "2019-02-06T18:49:26",
"slug": "birgitte-kirk-iversen",
"status": "publish",
"type": "post",
"link": "https://headlesscms.cederdorff.com/birgitte-kirk-iversen/",
"title": {
"rendered": "Birgitte Kirk Iversen"
},
"content": {
"rendered": "\n Senior Lecturer
 \n",
"protected": false
},
"excerpt": {
"rendered": " Senior Lecturer \n",
"protected": false
},
"author": 1,
"featured_media": 7,
"comment_status": "open",

Fetch & Headless CMS

WEB APP




```
fetch("...wp-json/wp/v2/posts")
  .then(function(response) {
    return response.json();
  })
  .then(function(json) {
    appendPosts(json);
  });
```

WP API

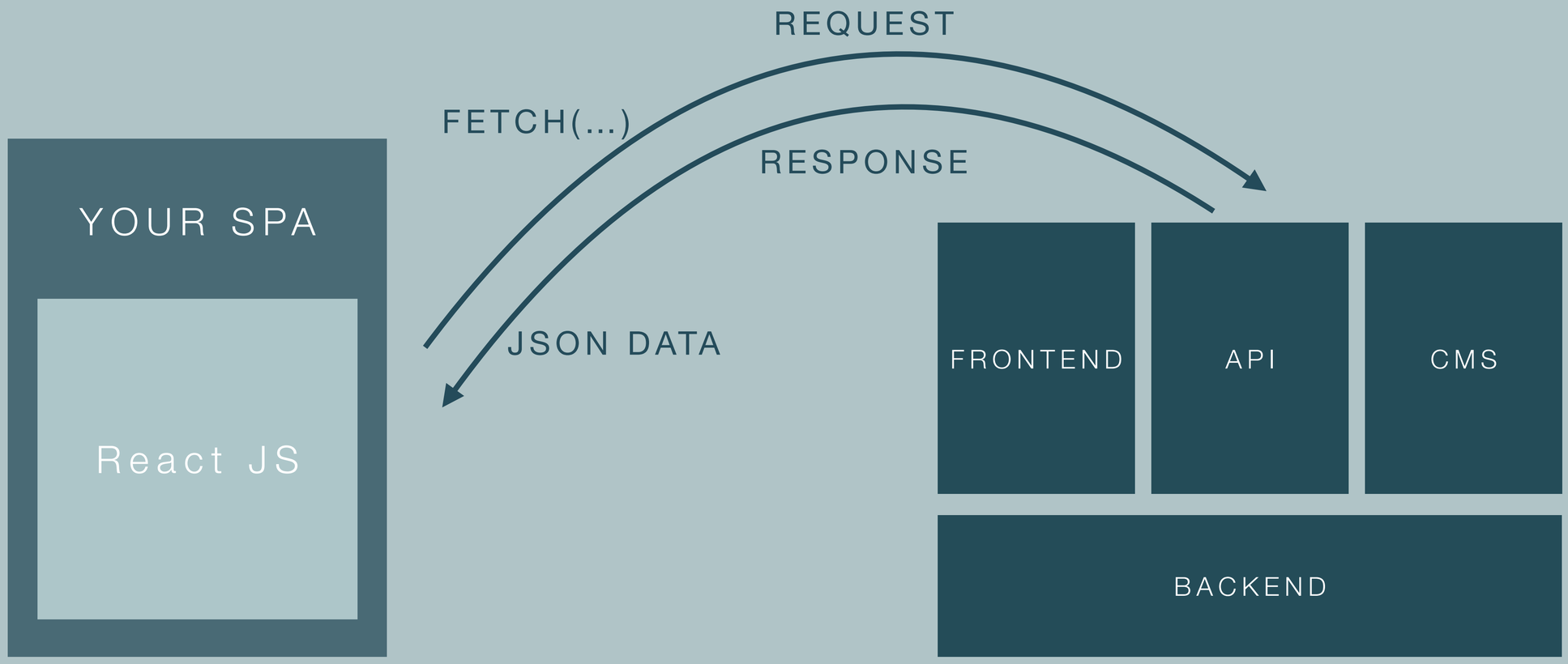
REQUEST (fetch)

RESPONSE (then)

DATA SOURCE

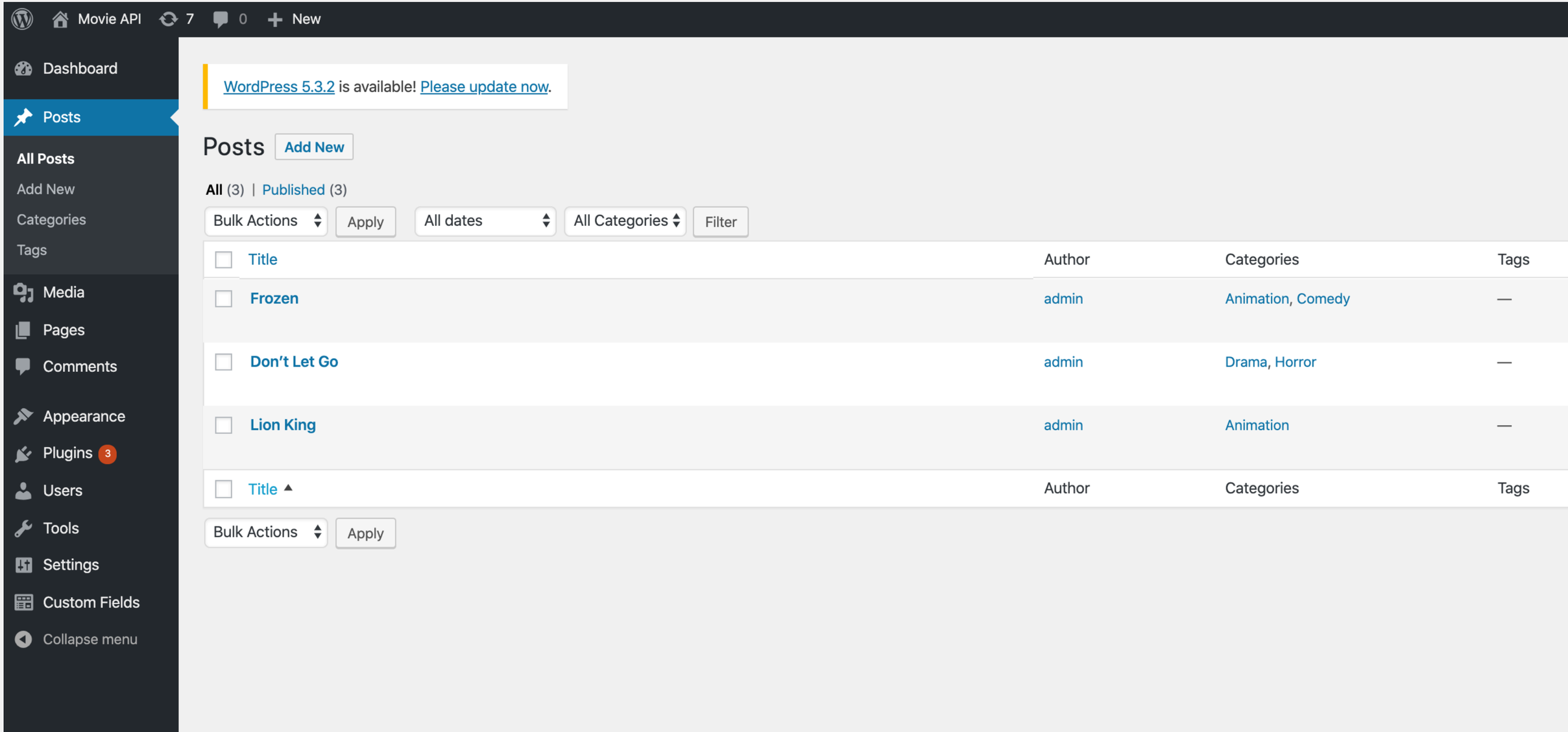
WEB APP

HEADLESS WP



movie-api.cederdorff.com

Fetch posts


Movie API
7
0
New
Dashboard
Posts
All Posts
Add New
Categories
Tags
Media
Pages
Comments
Appearance
Plugins
3
Users
Tools
Settings
Custom Fields
Collapse menu
WordPress 5.3.2 is available! Please update now.
Posts
Add New
All (3) | Published (3)
Bulk Actions
Apply
All dates
All Categories
Filter
Title
Author
Categories
Tags
Frozen
admin
Animation, Comedy
—
Don't Let Go
admin
Drama, Horror
—
Lion King
admin
Animation
—
Title
Author
Categories
Tags
Bulk Actions
Apply

Fetch posts

```
let _movies = [];

// fetch all movies from WP
async function getMovies() {
    let response = await fetch("https://movie-api.cederdorff.com/wp-json/wp/v2/posts?_embed");
    let data = await response.json();
    _movies = data;
    appendMovies(data);
    showLoader(false);
}

getMovies();
```

API Routes

Meta data: http://your-domain.com/wp-json/

Posts: http://your-domain.com/wp-json/wp/v2/posts

Pages: http://your-domain.com/wp-json/wp/v2/pages

Media: http://your-domain.com/wp-json/wp/v2/media

Types: http://your-domain.com/wp-json/wp/v2/types

Taxonomies: http://your-domain.com/wp-json/wp/v2/taxonomies

Categories: http://your-domain.com/wp-json/wp/v2/categories

Tags: http://your-domain.com/wp-json/wp/v2/tags

Users: http://your-domain.com/wp-json/wp/v2/users

Comments: http://your-domain.com/wp-json/wp/v2/comments

DOCS: https://developer.wordpress.org/rest-api/

Filter, sort & order

https://api.cederdorff.com/wp-json/wp/v2/posts?_embed

https://api.cederdorff.com/wp-json/wp/v2/posts?_embed&categories=2

https://api.cederdorff.com/wp-json/wp/v2/posts?orderby=date&order=desc&after=2019-00-10T17:00:00

https://api.cederdorff.com/wp-json/wp/v2/posts?orderby=date&order=asc

DOCS: https://developer.wordpress.org/rest-api/
ASK GOOGLE

Recap on JS

More on Fetch, JSON, Objects, Arrays, Loops, Functions, backticks and DOM Manipulation

https://cederdorff.com/race/slides/js-concepts.pdf


Code
Every
Day