Frontend Introduction

Web Development

ERHVERVSAKADEMI AARHUS
BUSINESS ACADEMY AARHUS

REMINDER 🙋🏼‍♂️

React, Angular, Vue.js, jQuery,

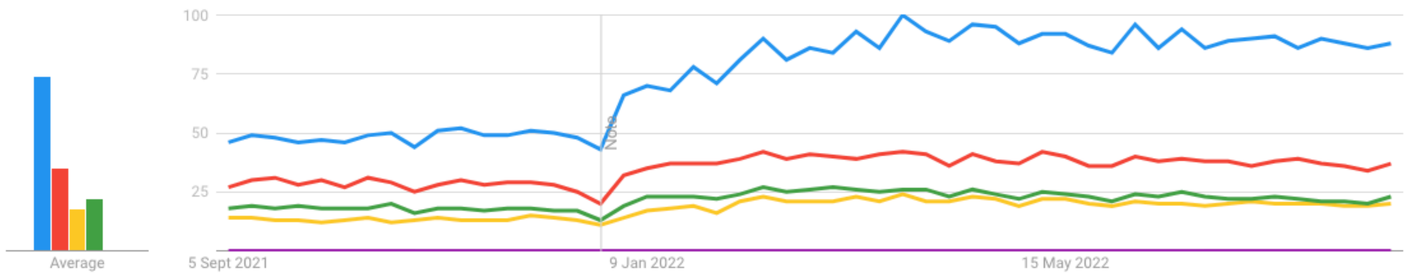
https://trends.google.com/trends/explore?q=%2Fm%2F012l1vxv,%2Fg%2F11c6w0ddw9,%2Fg%2F11c0vmgx5d,%2Fm%2F0268gyp,%2Fm%2F0s8wr70

Google Trends | Compare

React	Angular	Vue.js	jQuery	Ember.js
JavaScript library	Web framework	Topic	Software	Topic

Worldwide | Past 12 months | All categories | Web Search

Interest over time



Compared breakdown by region

Region

React Angular Vue.js jQuery Ember.js

Sort: Interest for React

1	Nigeria
2	Finland

Agenda

- Frontend Dev Stack & Tools
- Canvas Users Case
  - CSS Layout & JS Concepts
- Introduction to React
  - Setup React Dev Environment
  - React Core Concepts
  - Canvas Users Case w/ React

Lectures (RACE)

Date	Name	Notes
02/09/2022	1. Frontend: Introduction	Course Intro CSS Layouts From JS to React Pre React Intro
08/09/2022	2. Frontend: Thinking in React	Introduction to React React Core Concepts
19/09/2022	3. Frontend: React & Single Page Apps	React: Read + Create
26/09/2022	4. Frontend: React & CRUD	React: Update + Delete

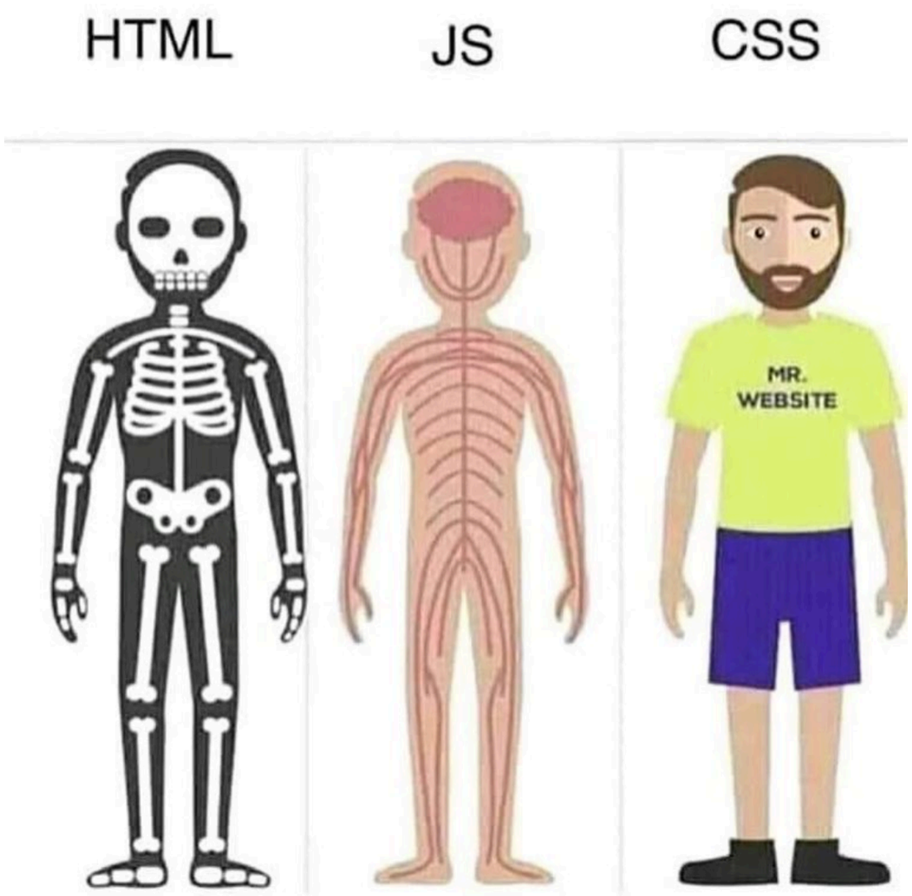
Frontend Programming - Lectures

Frontend stack

STRUCTURE
CONTENT

FUNCTIONALITY
BEHAVIOR

LAYOUT/STYLING
PRESENTATION



https://www.instagram.com/p/Bw90x2Ojo4g/

JavaScript, HTML & CSS

You can literally build anything with it!

Separation of concerns

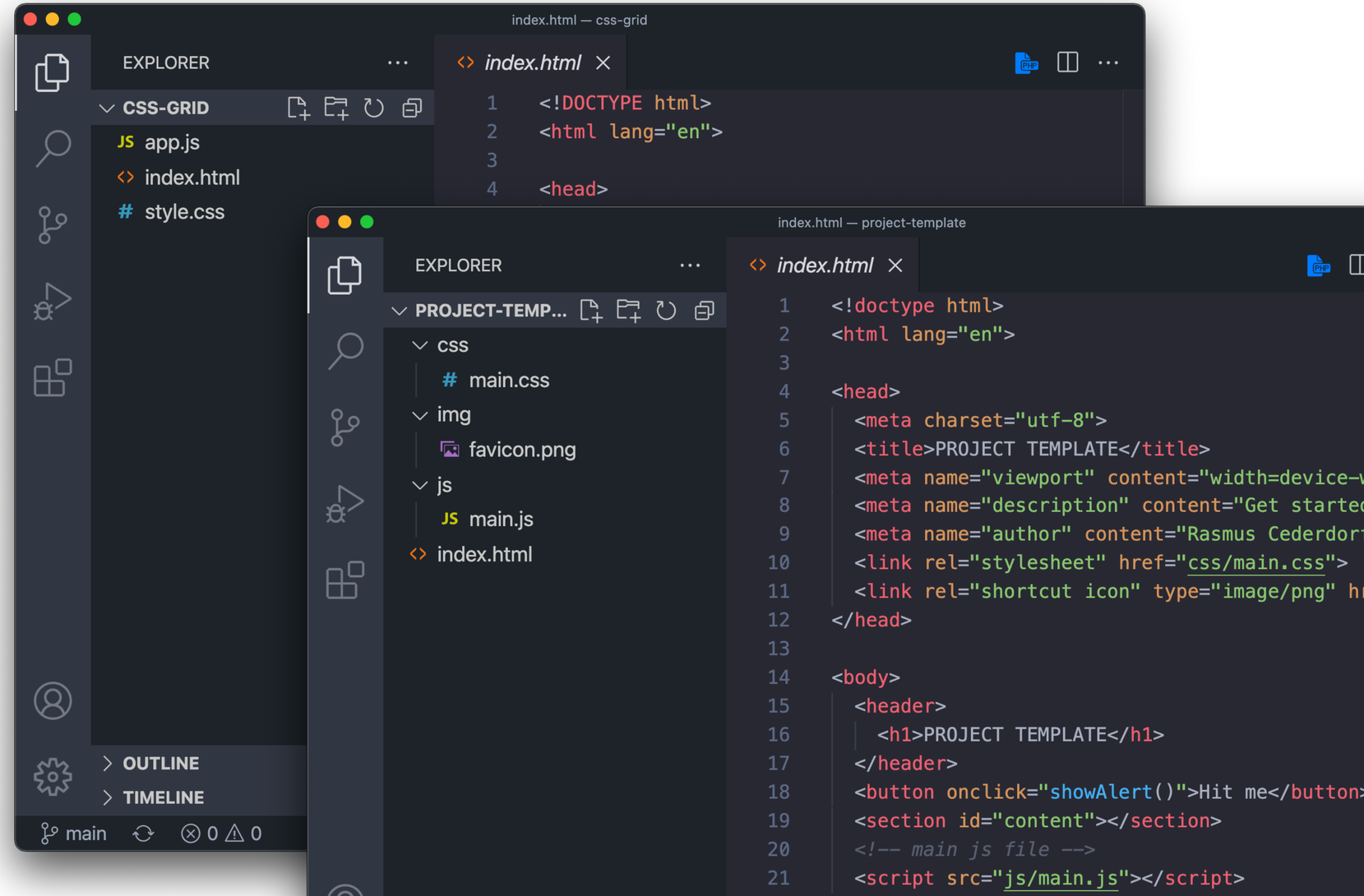
STRUCTURE
CONTENT

FUNCTIONALITY
BEHAVIOR

LAYOUT/STYLING
PRESENTATION

Frontend Project Structure

- Project structure:
  - HTML
  - CSS
  - JavaScript
- Keep a good structure & be consistent
- Separation of concerns
- Naming & conventions



Web Development

WEB APP

API

DATA SOURCE

FETCH(...)

REQUEST

JSON DATA

RESPONSE

Web Development

FRONTEND

API

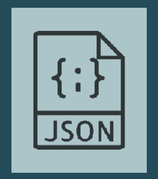
BACKEND

FETCH(...)

REQUEST

JSON DATA

RESPONSE



Tools

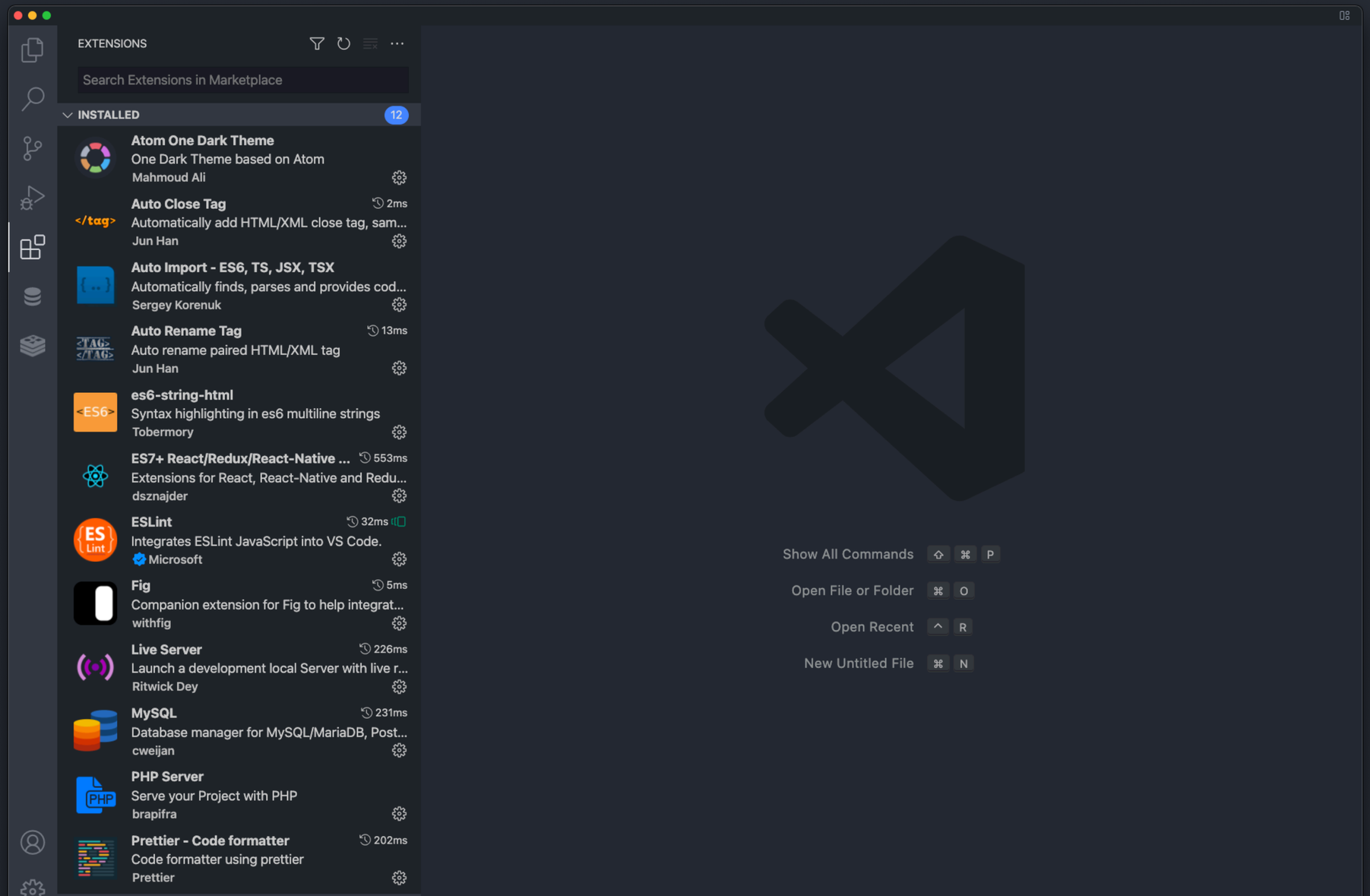
Visual Studio Code

Extensions: ESLint & Live Server

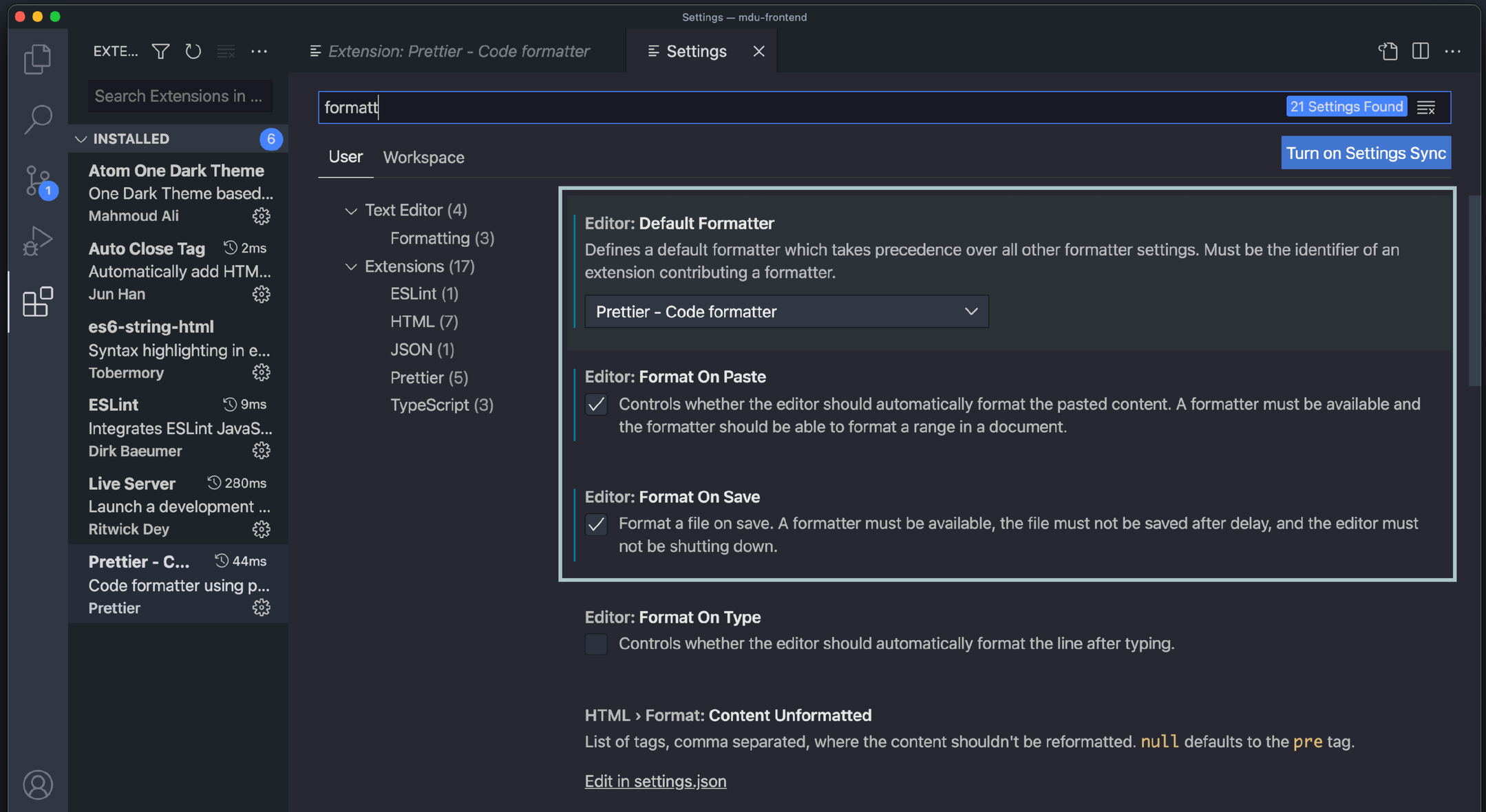
GitHub Desktop

Optional: FTP Client

Extensions


EXTENSIONS
Search Extensions in Marketplace
INSTALLED 12
Atom One Dark Theme
One Dark Theme based on Atom
Mahmoud Ali
Auto Close Tag 2ms
Automatically add HTML/XML close tag, sam...
Jun Han
Auto Import - ES6, TS, JSX, TSX
Automatically finds, parses and provides cod...
Sergey Korenuk
Auto Rename Tag 13ms
Auto rename paired HTML/XML tag
Jun Han
es6-string-html
Syntax highlighting in es6 multiline strings
Tobermory
ES7+ React/Redux/React-Native ... 553ms
Extensions for React, React-Native and Redu...
dsznajder
ESLint 32ms
Integrates ESLint JavaScript into VS Code.
Microsoft
Fig 5ms
Companion extension for Fig to help integrat...
withfig
Live Server 226ms
Launch a development local Server with live r...
Ritwick Dey
MySQL 231ms
Database manager for MySQL/MariaDB, Post...
cweijan
PHP Server
Serve your Project with PHP
brapifra
Prettier - Code formatter 202ms
Code formatter using prettier
Prettier
Show All Commands ⇧ ⌘ P
Open File or Folder ⌘ O
Open Recent ^ R
New Untitled File ⌘ N

Choose default formmatter in settings



Workflow

1. FETCH & PULL THE LATEST FROM GITHUB REPO WEB-FRONTEND (REACT-CDN-STARTERS OR REACT-STARTERS)

2. CREATE YOUR OWN LOCAL DEV FOLDER

3. COPY PROJECTS FROM WEB-FRONTEND INTO YOUR OWN DEV FOLDER.

4. OPEN YOUR OWN LOCAL DEV FOLDER IN VS CODE.

5. USE LIVE SERVER OR PHP SERVER EXTENSION TO RUN PROJECTS (PHP SERVER WHEN RUNNING PHP PROJECTS 🤷).

6. START WORKING IN YOUR OWN LOCAL DEV FOLDER.

OPTIONAL: SETUP GIT FOR YOUR OWN DEV FOLDER

Setup dev environment

1. install extensions.

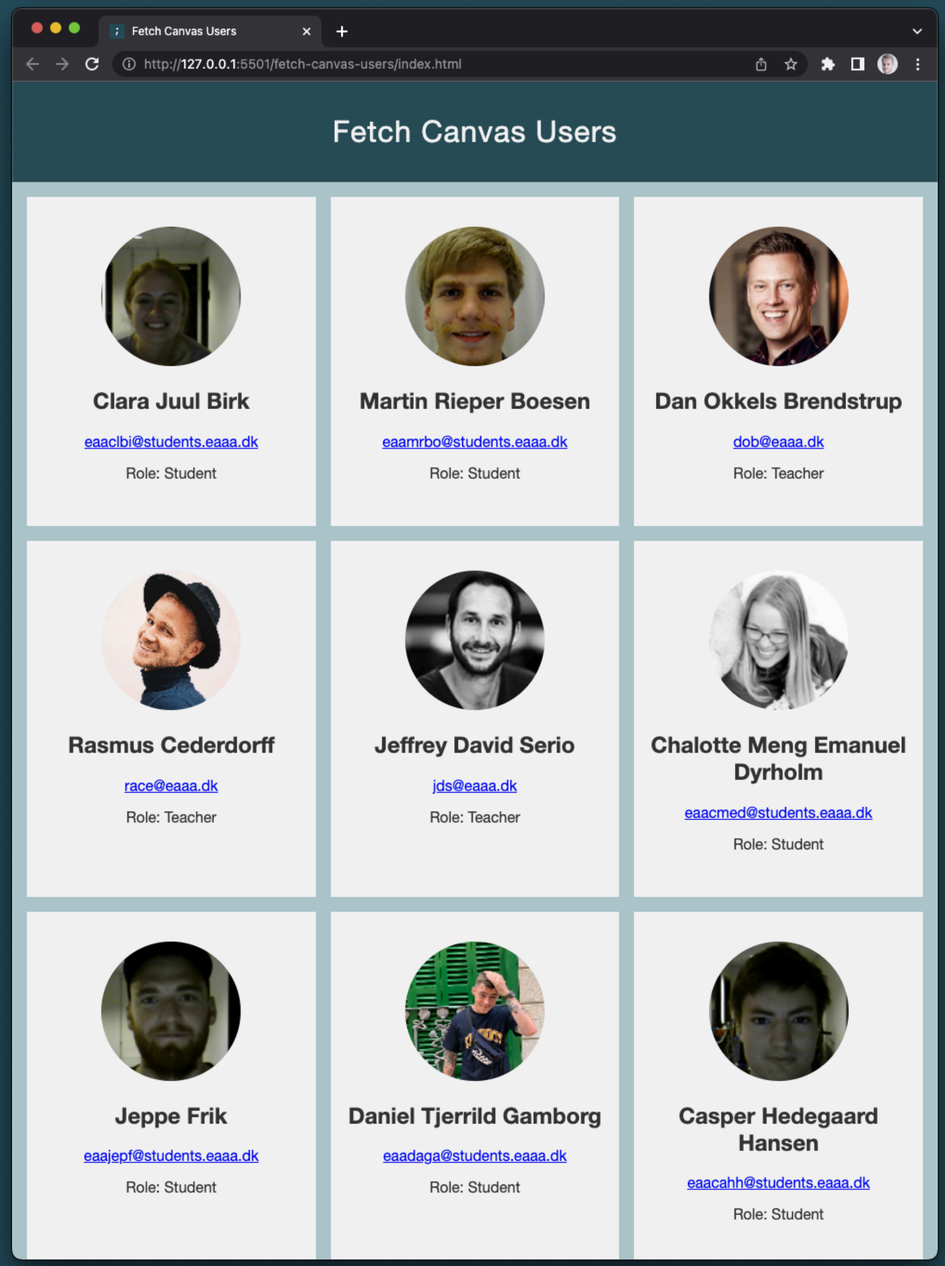
2. Choose default formatter.

3. Follow bullets on the workflow slide.

Canvas Users

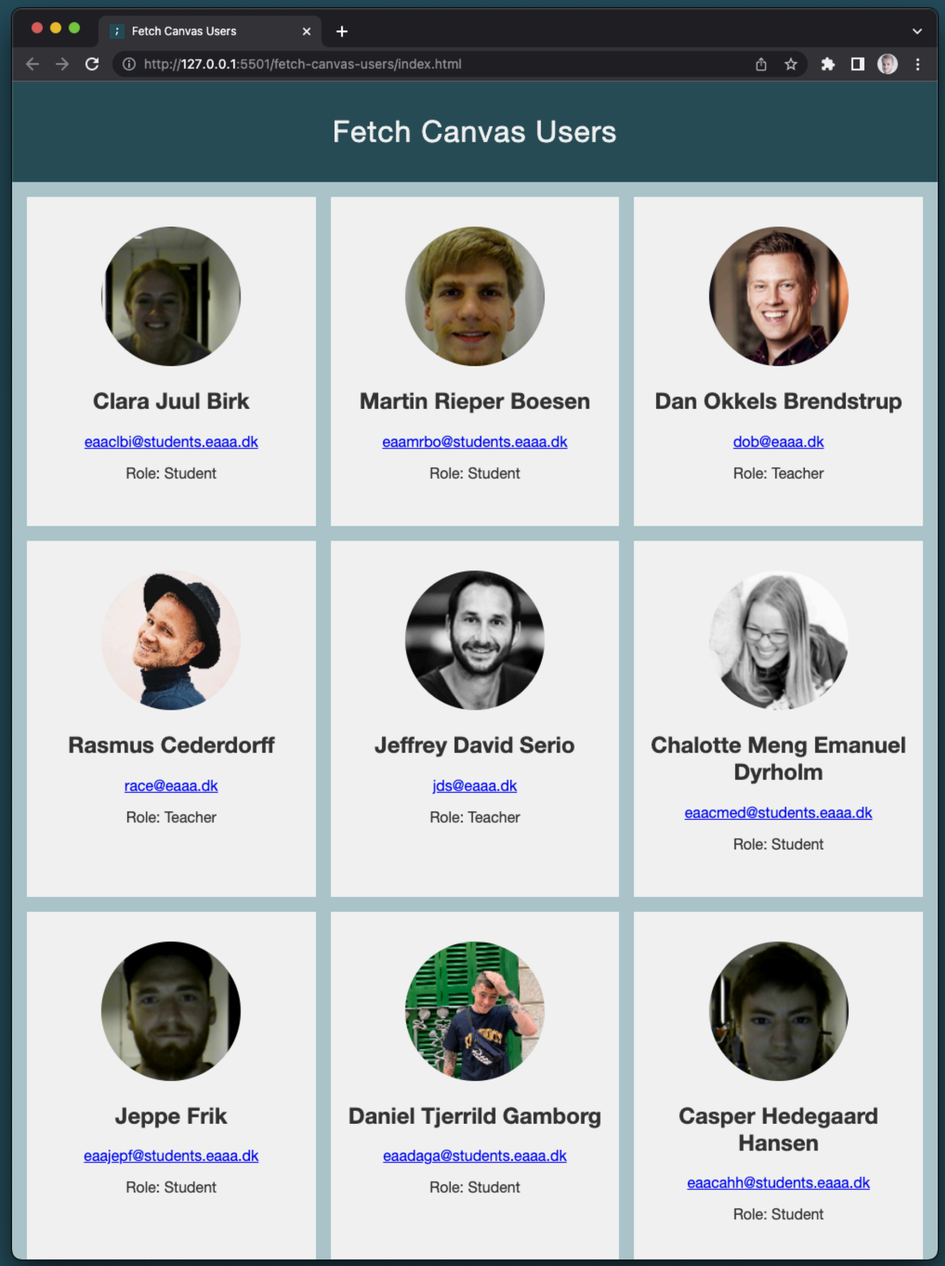
Fetch and display users from Canvas (data source) in a grid or flexbox.

1. Use `project-template` from GitHub.
2. In your `app.js` define a function named `getUsers`, fetching data from raw.githubusercontent.com/cederdorff/web-frontend/main/data/wu-e22a.json
3. Use `console.log()` to debug and test the data.
4. Create a function called `appendUsers(…)`. The function must loop through the user data and append each user to a grid container in the DOM.
    4.1. Use a `for of` loop. Inside of the loop you have to create a template for each user using a `` `template string` ``. The template is similar to how static-defined HTML would be.
    4.2. Make sure to execute the function(s).
    4.3. Add and/or customise CSS.



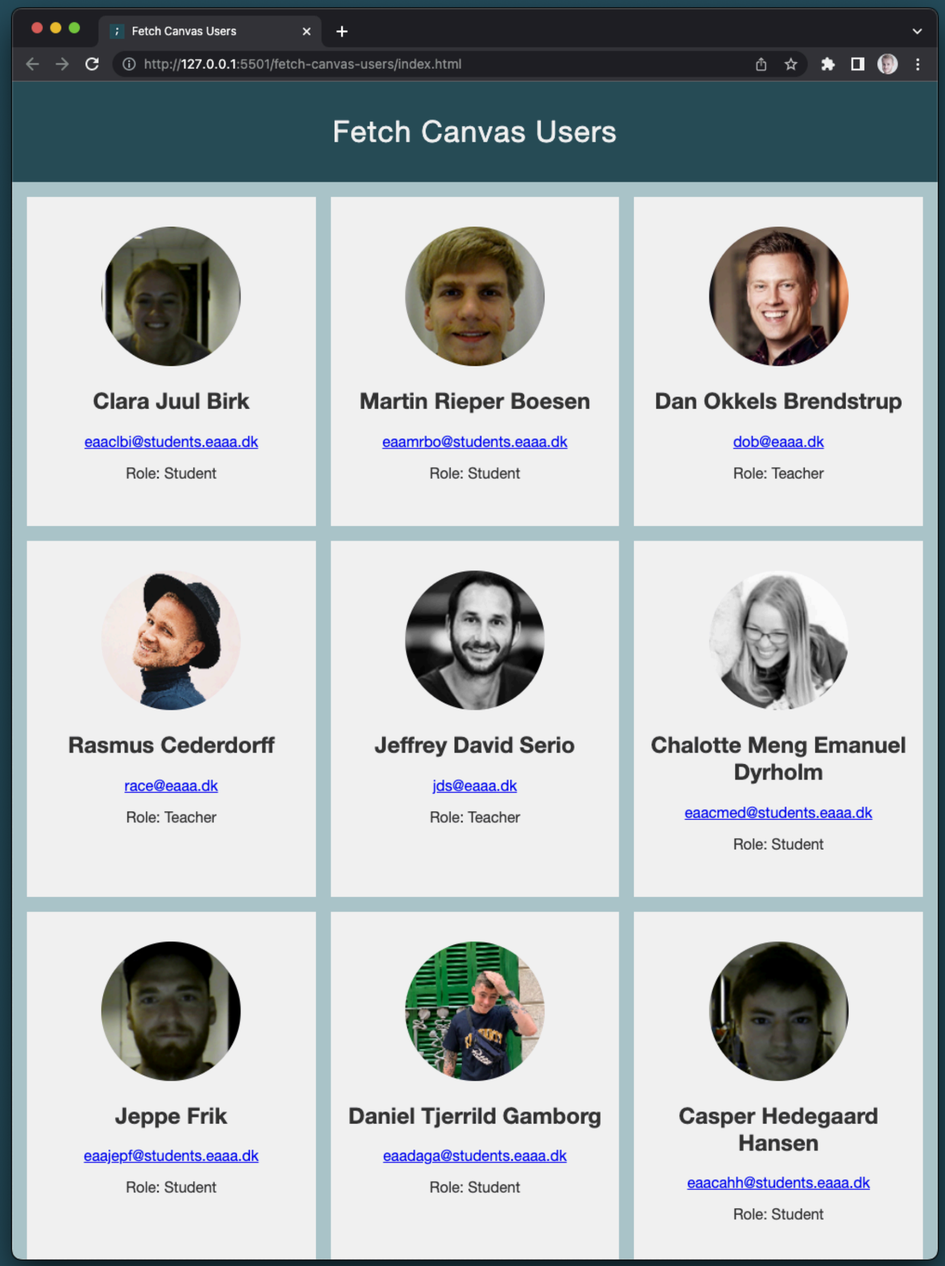
Canvas Users Cards

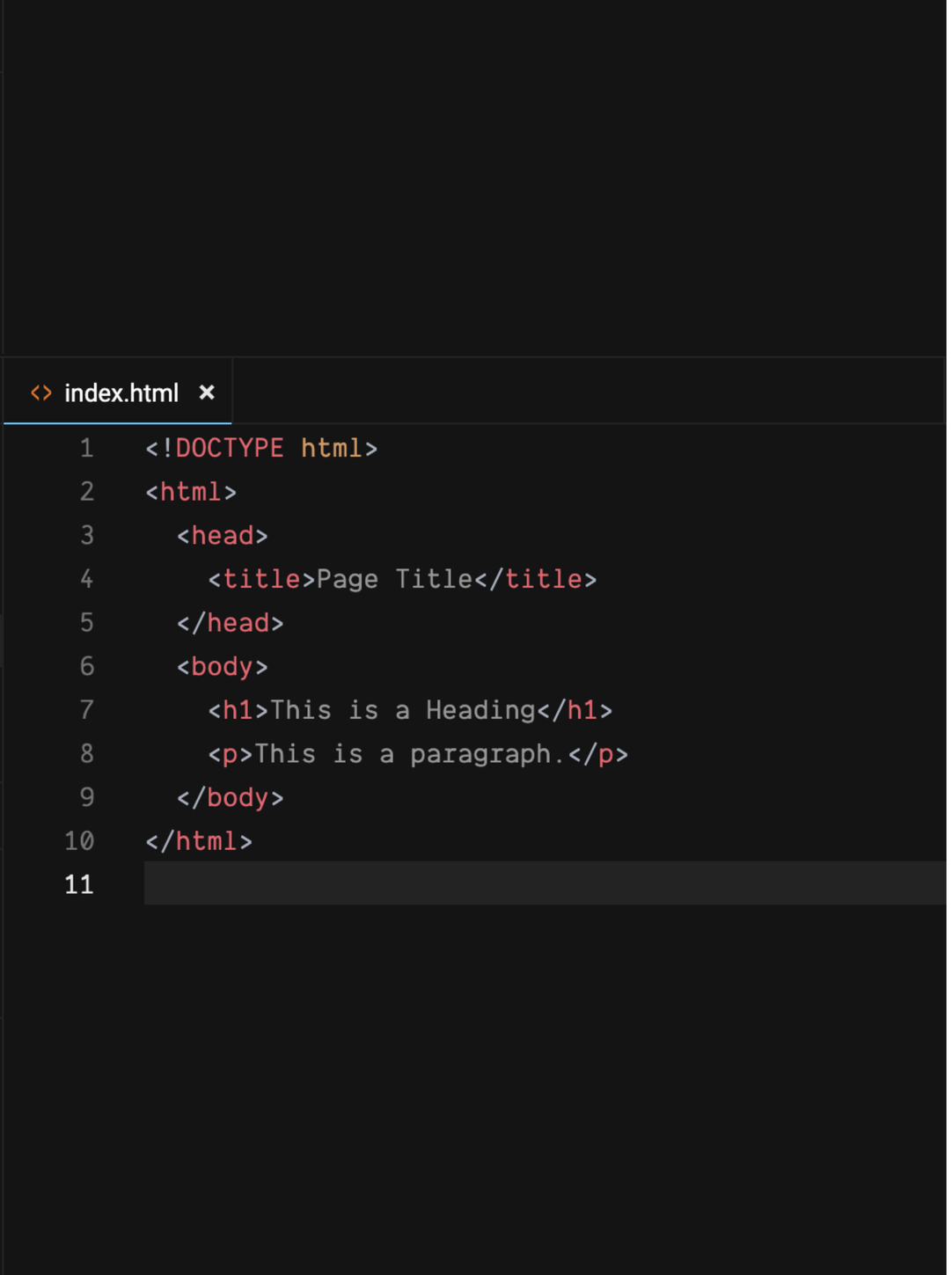
1. Explore and use `canvas-users-no-styling` from GitHub.
2. Make sure you understand `index.html` and how the CSS and JS are loaded.
3. Go through `app.js` and try to explain line by line.
    1. Identify and discuss the following: variables, functions, arguments/ parameters, template strings, DOM manipulation, loops, and the use of objects and arrays. Mark the different terms and concepts with comments in the js file.
    Use the linked slides to gain a deeper understanding of the terms and concepts.
    2. Note questions and comments.
4. When you have a solid knowledge of the project structure, start implementing CSS.
    1. Use CSS grid or flexbox to display all users nicely in a grid. Make sure to add a gap between the items (cards). Get inspired by the slides (Flexbox & Grid).
    2. Style the users as cards. Get inspired by this resource.



Canvas Users Extra

1. Implement search functionality searching on `name.includes(searchValue)`.
2. Implement sort functionality in order to sort by `name`, `sortable_name`, etc.
3. Make it possible to filter users on `enrollment_type` (students and teachers)
4. Implement create functionality using `array.push()`. Remember to generate a dummy `id` when adding a new user.
5. Create and implement an update-view using the `id` of the user to find and update properties of the user in the `users` array.
6. Implement delete functionality using the `id` to filter the `users` array.
7. Create a detail-view displaying all properties of the selected user.





WHAT IS HTML?

Hyper Text Markup Language

Standard markup language for creating Web pages

Describes the structure of a Web page

Consists of a series of elements

HTML elements tell the browser how to display the content

https://www.w3schools.com/html/html_intro.asp

HTML SEMANTIC ELEMENTS

... clearly describes its meaning to both the browser and the developer.

https://www.w3schools.com/html/html5_semantic_elements.asp

WHY USE SEMANTIC ELEMENTS?

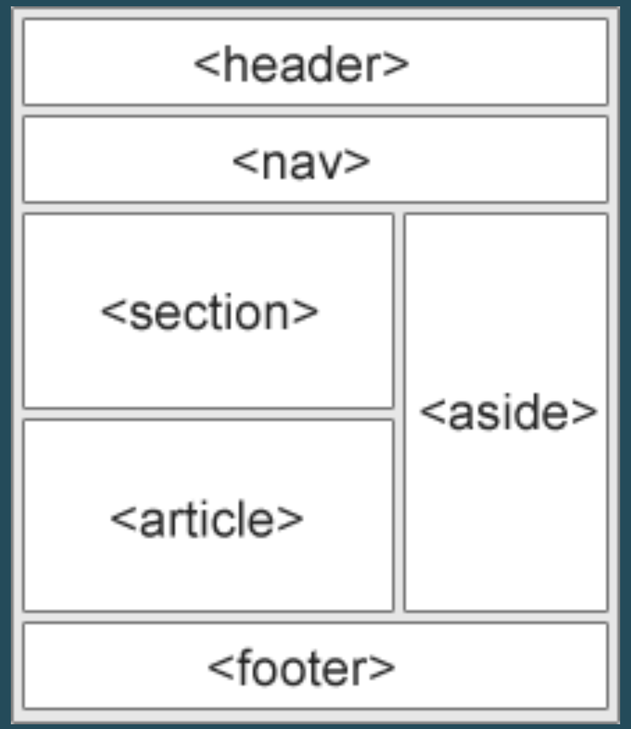
```
<body>
  <div id="header">
    <h1>PROJECT TEMPLATE</h1>
  </div>
  <div class="sections">
    <div class="article">
      <div class="figure">
        <img>
        <div class="figcaption"></div>
      </div>
    </div>
  </div>
  <div id="footer"></div>
  <!-- main js file -->
  <script src="js/main.js"></script>
</body>
```

```
<body>
  <header>
    <h1>PROJECT TEMPLATE</h1>
  </header>
  <section>
    <article>
      <figure>
        <img>
        <figcaption></figcaption>
      </figure>
    </article>
  </section>
  <footer></footer>
  <!-- main js file -->
  <script src="js/main.js"></script>
</body>
```

HTML SEMANTIC ELEMENTS

Semantic elements that can be used to define different parts of a website

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>



https://www.w3schools.com/html/html5_semantic_elements.asp

DIV TAGS?

Many web sites contain HTML code like:

```
<div id=“nav">
<div class=“header">
<div id="footer">
```

to indicate navigation, header, and footer.

Can we still use `div` tags?

https://www.w3schools.com/html/html5_semantic_elements.asp

Apple

apple.com

Incognito

div 971 × 64

Store Mac iPad iPhone Watch TV Music Support

We look forward to welcoming you to our stores. Whether you shop in a store or shop online, our Specialists can help you buy the products you love.

iPhone 13 Pro

Oh. So. Pro.

Learn more › Buy ›

iPhone 13

Your new superpower.

Learn more › Buy ›

Elements Console Sources Network 1

```
<aside id="ac-localeswitcher" data-analytics-region="locale switcher" data-analytics-activitymap-region-id="locale switcher" lang="da-DK" dir="ltr">…</aside>
<h1 class="visuallyhidden">Apple</h1>
<meta name="ac-gn-store-key" content="SFX9YPYY9PPXCU9KH">
<aside id="ac-gn-segmentbar" class="ac-gn-segmentbar" lang="en-US" dir="ltr" data-strings="{ 'exit': 'Exit', 'view': '{%STOREFRONT%} Store Home', 'segments': { 'smb': 'Business Store Home', 'eduInd': 'Education Store Home', 'other': 'Store Home' } }"></aside>
<input type="checkbox" id="ac-gn-menustate" class="ac-gn-menustate">
<nav id="ac-globalnav" class="js no-touch no-windows" role="navigation" aria-label="Global" data-hires="false" data-analytics-region="global nav" lang="en-US" dir="ltr" data-www-domain="www.apple.com" data-store-locale="us" data-store-root-path="/us" data-store-api="/[storefront]/shop/bag/status" data-search-locale="en_US" data-search-suggestions-api="/search-services/suggestions/" data-search-defaultlinks-api="/search-services/suggestions/defaultlinks/">…</nav>
<div class="ac-gn-blur"></div> == $0
<div id="ac-gn-curtain" class="ac-gn-curtain"></div>
<div id="ac-gn-placeholder" class="ac-nav-placeholder"></div>
<script type="text/javascript" src="/ac/globalnav/6/en_US/scripts/ac-globalnav.built.js"></script>
<div id="ac-gn-viewport-emitter">…</div>
<script src="/metrics/ac-analytics/2.13.1/scripts/ac-analytics.js" type="text/javascript" charset="utf-8"></script>
<main class="main" role="main">
<section class="homepage-section collection-module" data-module-template="ribbon"> grid
<div data-unit-id="shop-online" data-analytics-activitymap-region-id="ribbon-shop-online">…</div>
</section>
<section class="homepage-section collection-module" data-module-template="heroes" data-analytics-region="hero"> grid
<div data-unit-id="iphone-13-pro" data-analytics-section-engagement="name:hero-iphone-13-pro">…</div>
<div data-unit-id="iphone-13" data-analytics-section-engagement="nam
```

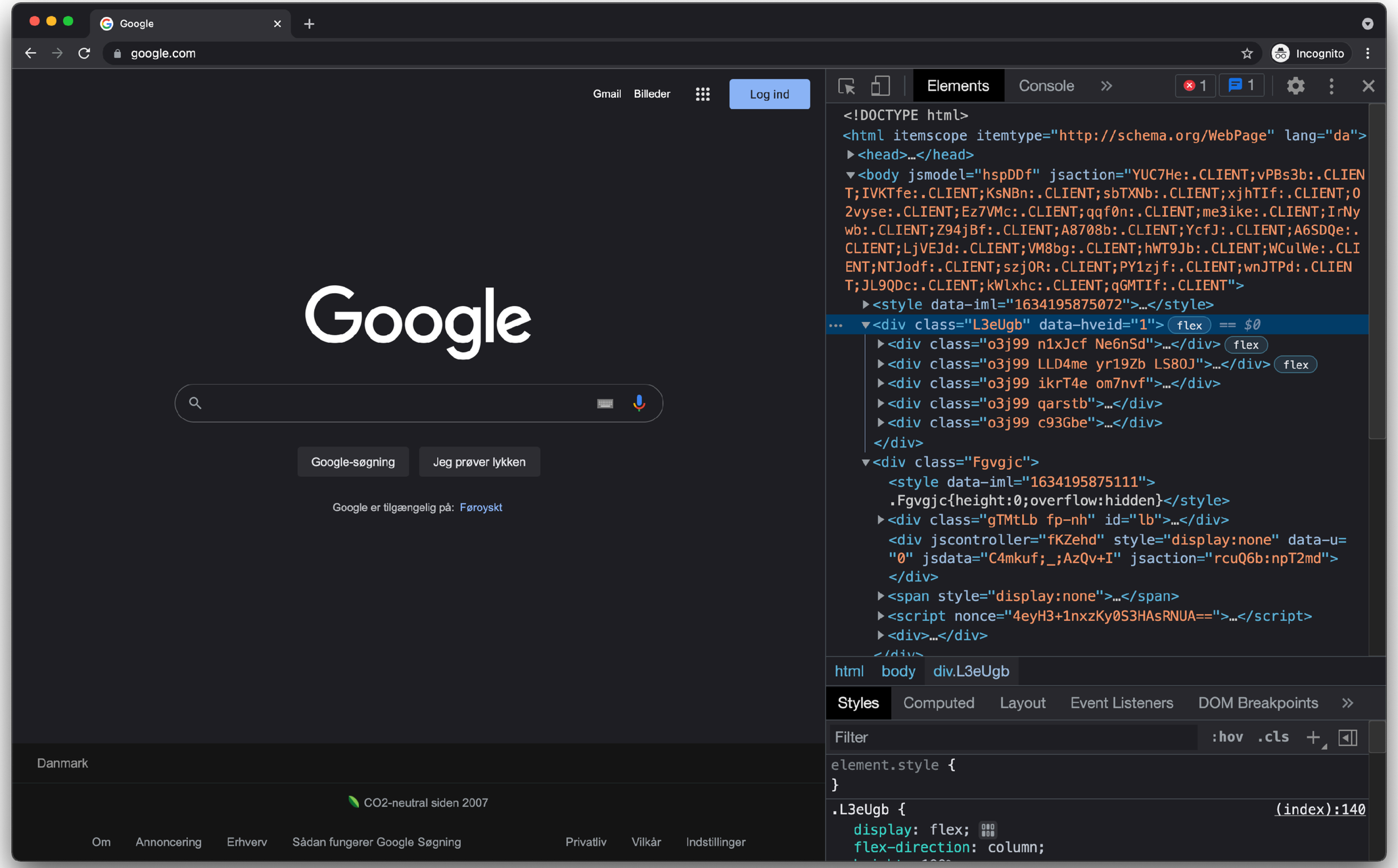
… .enhanced-layout body.page-home.ac-nav-overlap.body-with-ribbon div.ac-gn-blur …

Styles Computed Layout Event Listeners DOM Breakpoints Properties

Filter :hov .cls

```
element.style {
}

html #ac-globalnav, html #ac-globalnav ~ .ac-gn-blur, html #ac-gn-segmentbar, html #ac-localeswitcher, html div.adv-wrapper {
  position: fixed;
}
```

main.built.css:1


Google
google.com
Incognito
Gmail
Billeder
Log ind
Google
Google-søgning
Jeg prøver lykken
Google er tilgængelig på: Føroyskt
Danmark
CO2-neutral siden 2007
Om
Annoncering
Erhverv
Sådan fungerer Google Søgning
Privatliv
Vilkår
Indstillinger
Elements
Console
<!DOCTYPE html>
<html itemscope itemtype="http://schema.org/WebPage" lang="da">
<head>…</head>
<body jsmodel="hspDDf" jsaction="YUC7He:.CLIENT;vPBs3b:.CLIENT;IVKTfe:.CLIENT;KsNBn:.CLIENT;sbTXNb:.CLIENT;xjhTIf:.CLIENT;O2vyse:.CLIENT;Ez7VMc:.CLIENT;qqf0n:.CLIENT;me3ike:.CLIENT;IrNywb:.CLIENT;Z94jBf:.CLIENT;A8708b:.CLIENT;YcfJ:.CLIENT;A6SDQe:.CLIENT;LjVEJd:.CLIENT;VM8bg:.CLIENT;hWT9Jb:.CLIENT;WCulWe:.CLIENT;NTJodf:.CLIENT;szjOR:.CLIENT;PY1zjf:.CLIENT;wnJTPd:.CLIENT;JL9QDc:.CLIENT;kWlxhc:.CLIENT;qGMTIf:.CLIENT">
<style data-iml="1634195875072">…</style>

…
…
…
…
…

<style data-iml="1634195875111">
.Fgvgjc{height:0;overflow:hidden}</style>
…

…
<script nonce="4eyH3+1nxzKy0S3HAsRNUA==">…</script>
…
html
body
div.L3eUgb
Styles
Computed
Layout
Event Listeners
DOM Breakpoints
Filter
:hov
.cls
element.style {
}
.L3eUgb {
(index):140
display: flex;
flex-direction: column;

HTML Semantic Elements

w3schools.com/html/html5_semantic_elements.asp

Incognito

Tutorials References Exercises Website Paid Courses Log in

HTML CSS JAVASCRIPT SQL PYTHON PHP BOOTSTRAP

HTML Semantic Elements

❮ Previous Next ❯

Semantic elements = elements with a meaning.

What are Semantic Elements?

A semantic element clearly describes its meaning to both the browser and the developer.

Examples of **non-semantic** elements: `<div>` and `<span>` - Tells nothing about its content.

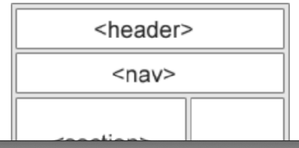
Examples of **semantic** elements: `<form>`, `<table>`, and `<article>` - Clearly defines its content.

Semantic Elements in HTML

Many web sites contain HTML code like: to indicate navigation, header, and footer.

In HTML there are some semantic elements that can be used to define different parts of a web page:

- <article>
- <aside>
- <details>
- <figcaption>



Elements Console 2 2 28

```
<body> == $0
  <div class="w3-bar w3-card-2 notranslate" style="position: relative;z-index: 4;font-size: 18px;background-color: white;color:#282A35;padding-left:12px;padding-right:16px;font-family: 'Source Sans Pro', sans-serif;">…</div>
  <div style="display:none;position:absolute;z-index:4;right:52px;height:44px;background-color:#282A35;letter-spacing:normal;" id="googleSearch">…</div>
  <div style="display:none;position:absolute;z-index:3;right:111px;height:44px;background-color:#282A35;text-align:right;padding-top:9px;" id="google_translate_element"></div>
  <div class="w3-card-2 topnav notranslate" id="topnav" style="position: relative;">…</div>
  <div id="myAccordion" class="w3-card-2 w3-center w3-hide-large w3-hide-medium" style="width: 100%; position: absolute; display: none; background-color: rgb(231, 233, 235); padding-top: 0px;">…</div>
  <script> </script>
  <div class="w3-sidebar w3-collapse" id="sidenav" style="top: 118px; display: none;">…</div>
  <div class="w3-main w3-light-grey" id="belowtopnav" style="margin-left: 220px; padding-top: 0px;">
    <div class="w3-row w3-white">
      ::before
      <div class="w3-col l10 m12" id="main">
        <div id="mainLeaderboard" style="overflow:hidden;">…</div>
        <h1>…</h1>
        <div class="w3-clear nextprev"> </div>
```

html body

Styles Computed Layout Event Listeners DOM Breakpoints

Filter :hov .cls

```
element.style {
}
html, body {                                    w3schools28.css:27
  font-family: Verdana,sans-serif;
  font-size: 15px;
```

DIV TAGS?

Use semantic elements as much as possible (!)

... but don’t let it stop you from building something amazing 🎉

- RACE.js

https://www.w3schools.com/html/html5_semantic_elements.asp

WHAT IS CSS?

CSS stands for **C**ascading **S**tyle **S**heets

CSS describes how HTML elements are to be displayed on screen, paper, or in other media

CSS **saves a lot of work**. It can control the layout of multiple web pages all at once

External stylesheets are stored in **CSS files**

https://www.w3schools.com/css/css_intro.asp

styles.css

```
1   body {
2     font-family: Helvetica, Arial, sans-serif;
3     text-align: center;
4   }
5
6   h1 {
7     font-weight: lighter;
8   }
9
10  #my-paragraph {
11    color: blue;
12  }
13
14  .big {
15    font-size: 25px;
16  }
17
```

CSS SELECTORS

CSS selectors are used to "find" (or select) the HTML elements you want to style.

Simple selectors (name, id, class)

Combinator selectors (a specific relationship between them)

Pseudo-class selectors (a certain state)

Pseudo-elements selectors (a part of an element)

Attribute selectors (attribute or attribute value)

https://www.w3schools.com/css/css_selectors.asp

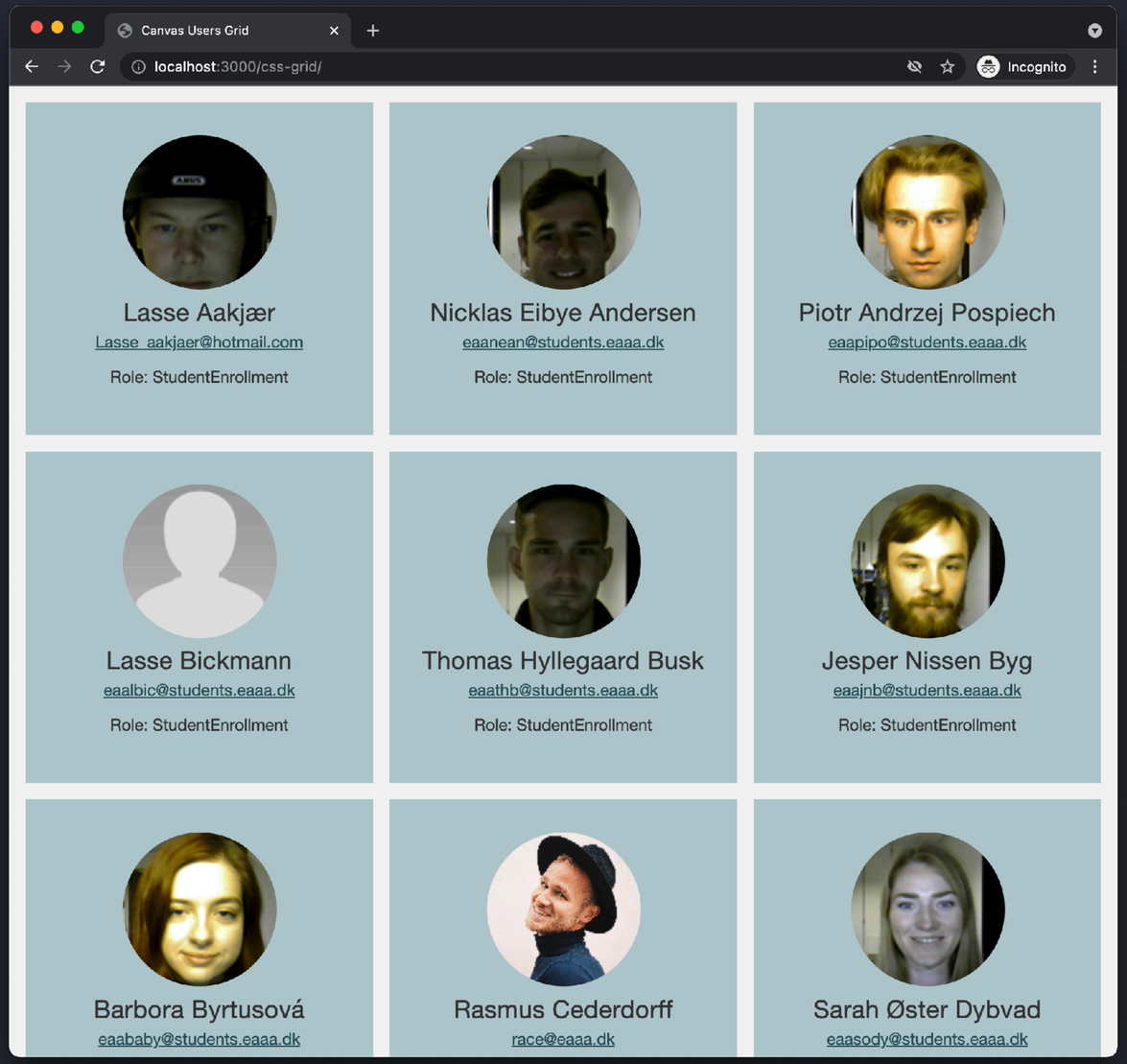
CSS LAYOUTS

GRID & FLEXBOX

... two CSS layout models that can be used to create layouts with just a couple of CSS rules.

... responsive and flexible layouts.

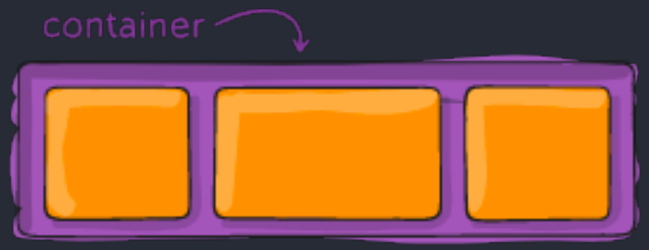
CSS GRID

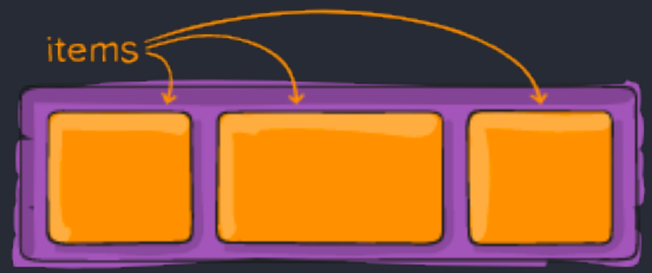



```html
<section id="users" class="grid"></section>
```

```css
/* ---------- grid styling ---------- */

.grid {
  display: grid;
  grid-template-columns: 1fr;
  padding: 1em;
  gap: 1em;
}

@media (min-width: 600px) {
  .grid {
    grid-template-columns: 1fr 1fr;
  }
}

@media (min-width: 992px) {
  .grid {
    grid-template-columns: 1fr 1fr 1fr;
  }
}
```

CSS GRID



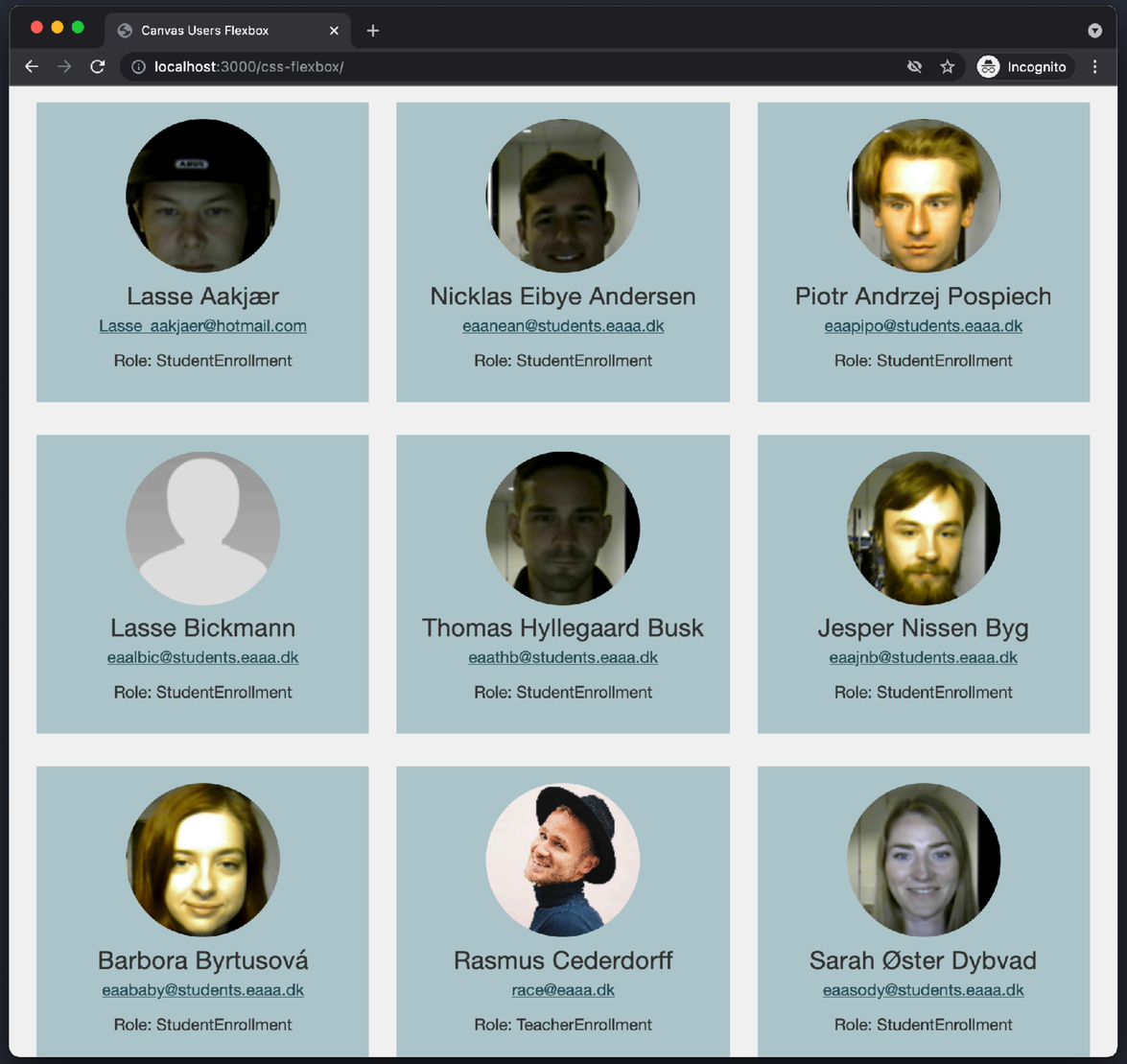



```css
.grid {
  display: grid;
  grid-template-columns: 1fr;
  padding: 1em;
  gap: 1em;
}

@media (min-width: 600px) {
  .grid {
    grid-template-columns: 1fr 1fr;
  }
}

@media (min-width: 992px) {
  .grid {
    grid-template-columns: 1fr 1fr 1fr;
  }
}
```

```css
.grid > article {
  text-align: center;
  padding: 2em 1em;
  background-color: var(--light-green);
  transition: 0.3s;
}

.grid > article:hover {
  box-shadow: 0 8px 16px 0 var(--green);
}

.grid > article img {
  width: 100%;
  max-width: 150px;
  height: auto;
  border-radius: 50%;
}
```

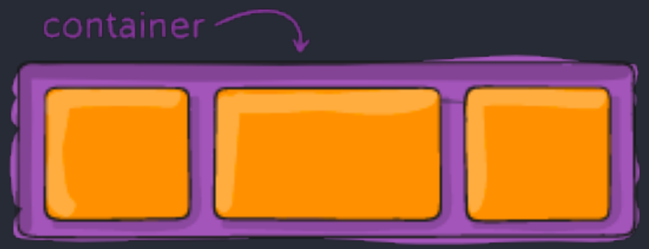
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

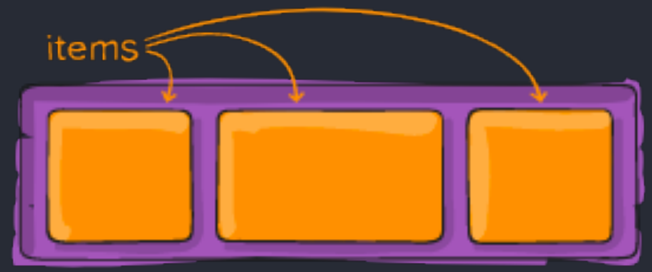
CSS FLEXBOX




```
<section id="users" class="flexbox">…
</section>

/* ---------- flexbox styling ---------- */

.flexbox {
  display: flex;
  flex-wrap: wrap;
  justify-content: space-evenly;
}

.flexbox article {
  margin: 1em;
  padding: 1em;
  text-align: center;
  background-color: var(--light-green);
  transition: 0.3s;
  flex-basis: 100%; /*default size of an element before the remaining spa
  flex-grow: 1; /*ability for a flex item to grow. 1 = all items equally*
}

@media (min-width: 600px) {
  .flexbox article {
    flex-basis: 40%;
  }
}

@media (min-width: 992px) {
  .flexbox article {
    flex-basis: 27%;
  }
}
```

CSS FLEXBOX

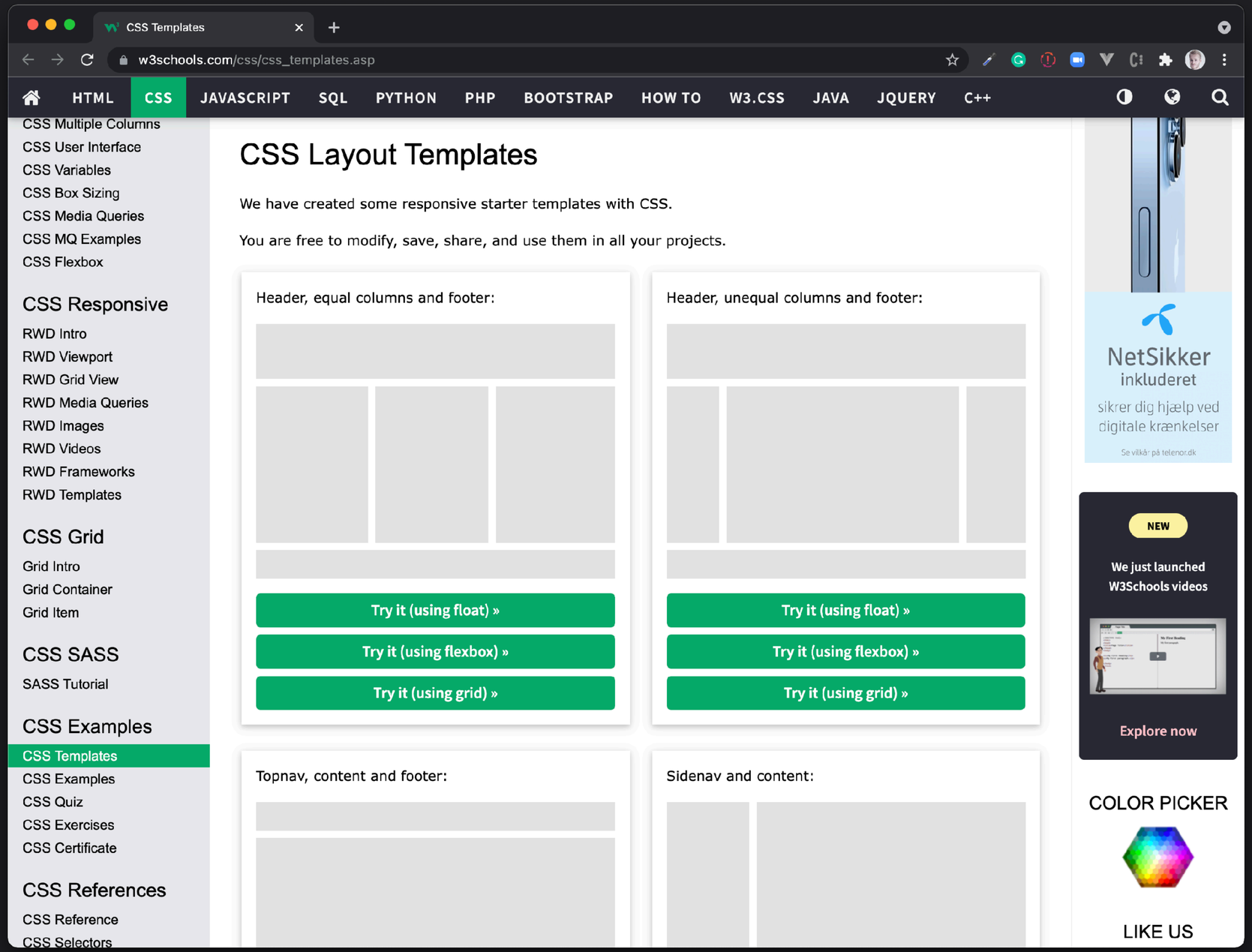




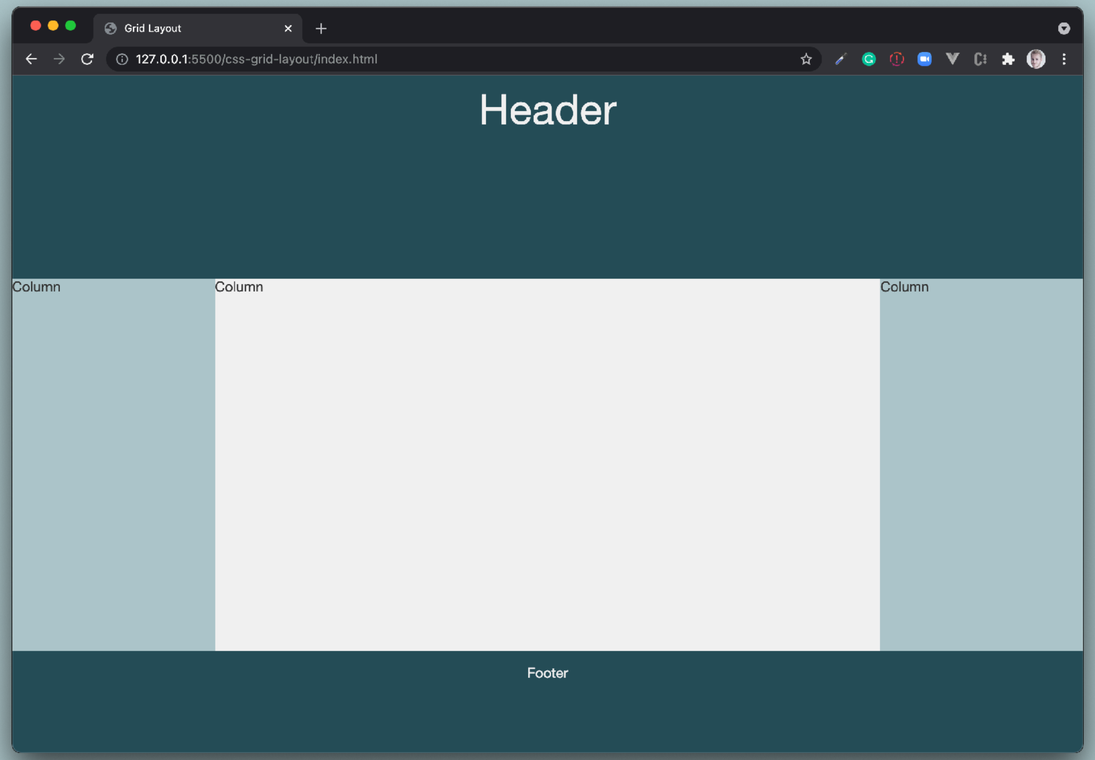

```css
.flexbox {
  display: flex;
  flex-wrap: wrap;
  justify-content: space-evenly;
}
```

```css
.flexbox article {
  margin: 1em;
  padding: 1em;
  text-align: center;
  background-color: var(--light-green);
  transition: 0.3s;
  flex-basis: 100%; /*default size of an element before the remaining space is distrib
  flex-grow: 1; /*ability for a flex item to grow. 1 = all items equally*/
}

@media (min-width: 600px) {
  .flexbox article {
    flex-basis: 40%;
  }
}

@media (min-width: 992px) {
  .flexbox article {
    flex-basis: 27%;
  }
}
```

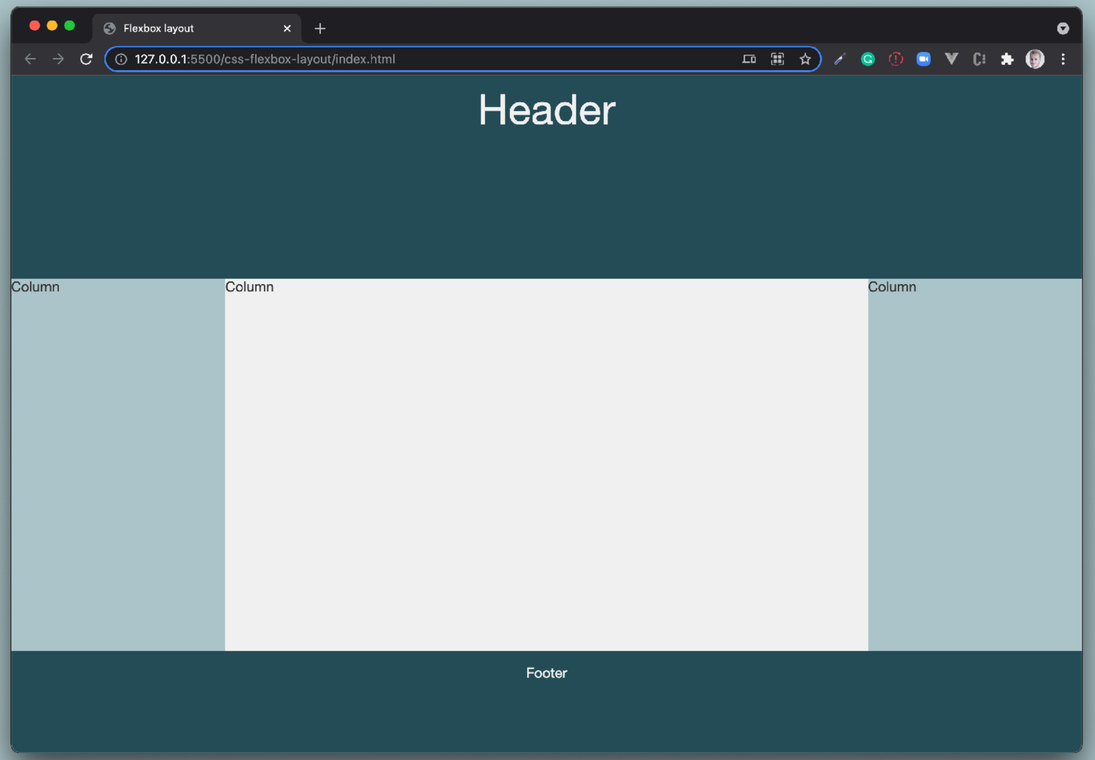
https://css-tricks.com/snippets/css/a-guide-to-flexbox/



https://www.w3schools.com/css/css_templates.asp



css-grid-layout



css-flexbox-layout

CSSLAYOUT.IO

LAYOUTS AND PATTERNS MADE WITH CSS

CSS TEMPLATES

https://www.w3schools.com/css/css_templates.asp

CSS VARIABLES

`var()` - refer to a defined CSS variable.

CSS variables are declared inside the `:root` selector

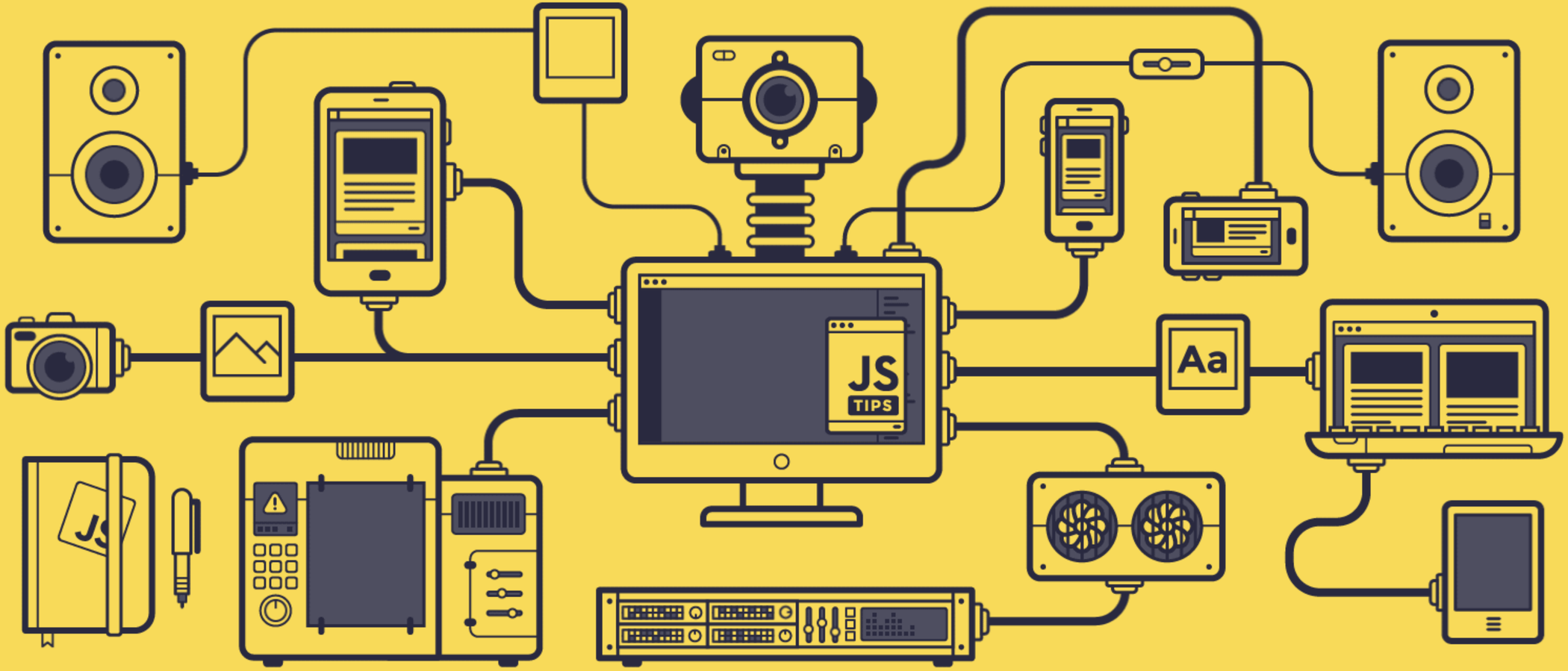
CSS variables can be accessed through the entire CSS document.

Makes the code easier to read (more understandable)

Makes it much easier to change the color values

https://www.w3schools.com/css/css3_variables.asp

```css
/* ---------- root variables ---------- */
:root {
  --green: rgb(38, 76, 89);
  --green-opacity: rgba(38, 76, 89, 0.2);
  --light-green: rgb(172, 198, 201);
  --light-grey: #f1f1f4;
  --text-color-light: #f1f1f1;
  --text-color-dark: #333;
  --white: #fff;
  --font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;
}

/* ---------- styling ---------- */
html,
body {
  color: var(--text-color-dark);
  font-family: var(--font-family);
  margin: 0;
  padding: 0;
  border: 0;
  outline: 0;
  background-color: var(--light-grey);
}

a {
  color: var(--green);
}

.grid > article {
  text-align: center;
  padding: 2em 1em;
  background-color: var(--light-green);
  transition: 0.3s;
}
```


JS
TIPS
Aa

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

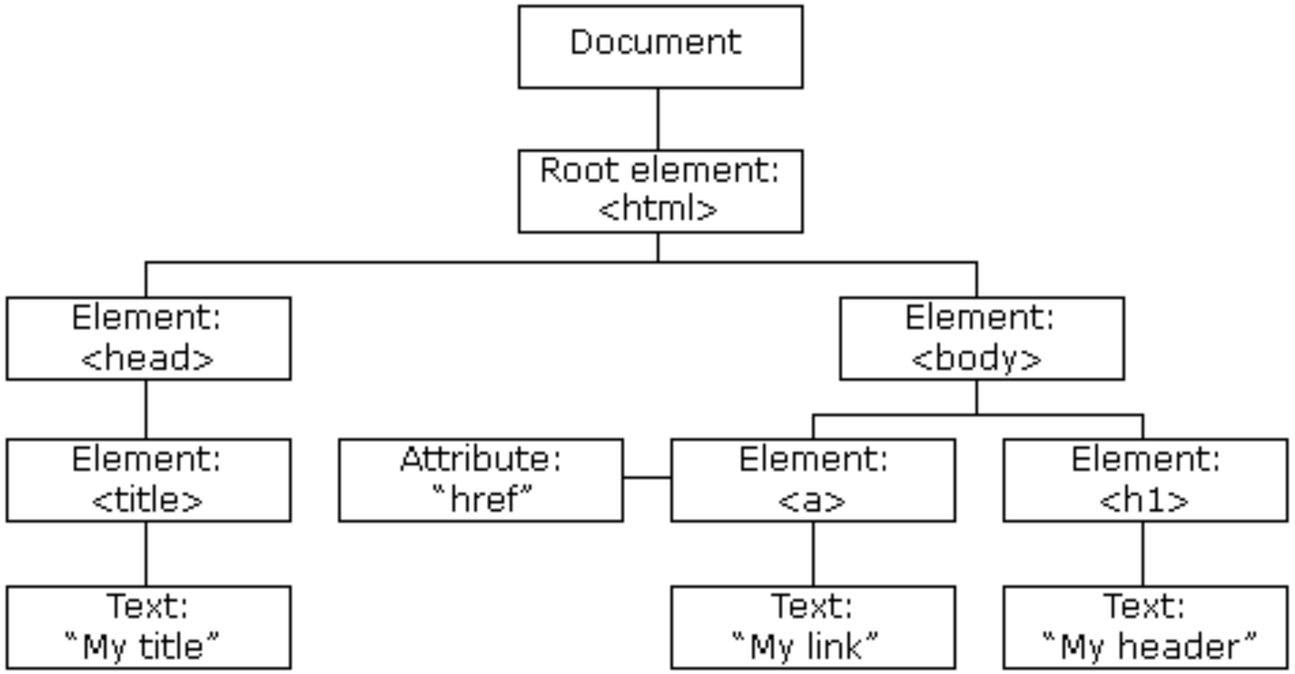
```
// declaring a variable with a value
let message = "Hi Frontenders!"

//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>
```

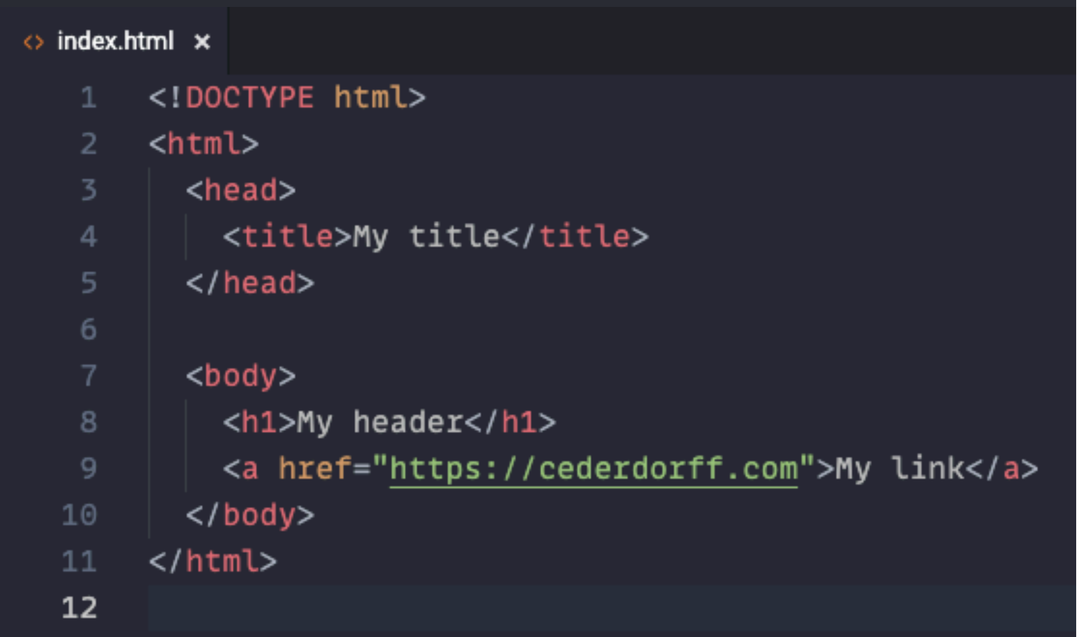
JAVASCRIPT HTML DOM

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



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: getElementsBy*

```
<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.getElementsByClassName('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

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

JS HTML DOM

`getElement*` or `querySelector*`?

ES6+

MODERN JAVASCRIPT

MODERN JAVASCRIPT

LET & CONST
TEMPLATE STRING
ARROW FUNCTIONS
FETCH
PROMISES
ASYNC & AWAIT
FOR OF LOOP
ARRAY.FIND()
ARRAY.MAP()
ARRAY.REDUCE()
ARRAY.FILTER()
ARRAY.SORT()
ARRAY.CONCAT()
DEFAULT PARAMS
DESTRUCTURING OBJECTS
DESTRUCTURING ARRAYS
OBJECT LITERAL
SPREAD OPERATOR
CLASSES
MODULES
IMPORT & EXPORT

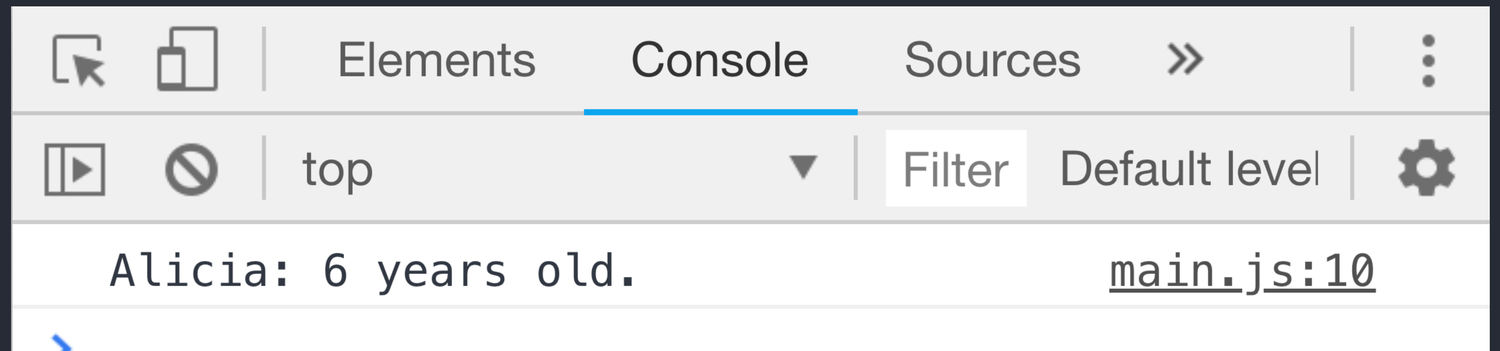
Variables

... are used to store data (values, objects, collections) in the memory

Variables

Store data in the memory

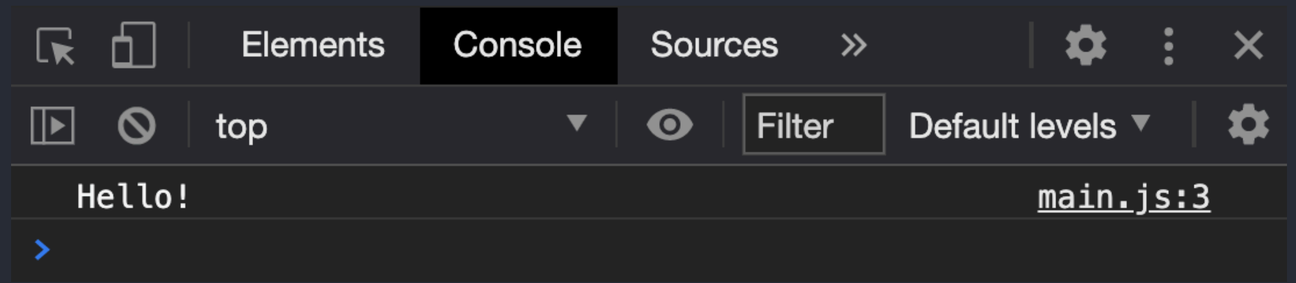
```
let name = "Alicia";
let age = 6;

console.log(name + ": " + age + " years old.");
```



Variables

Store data in the memory

```
let message = "Hello!";

console.log(message);
```



Variables

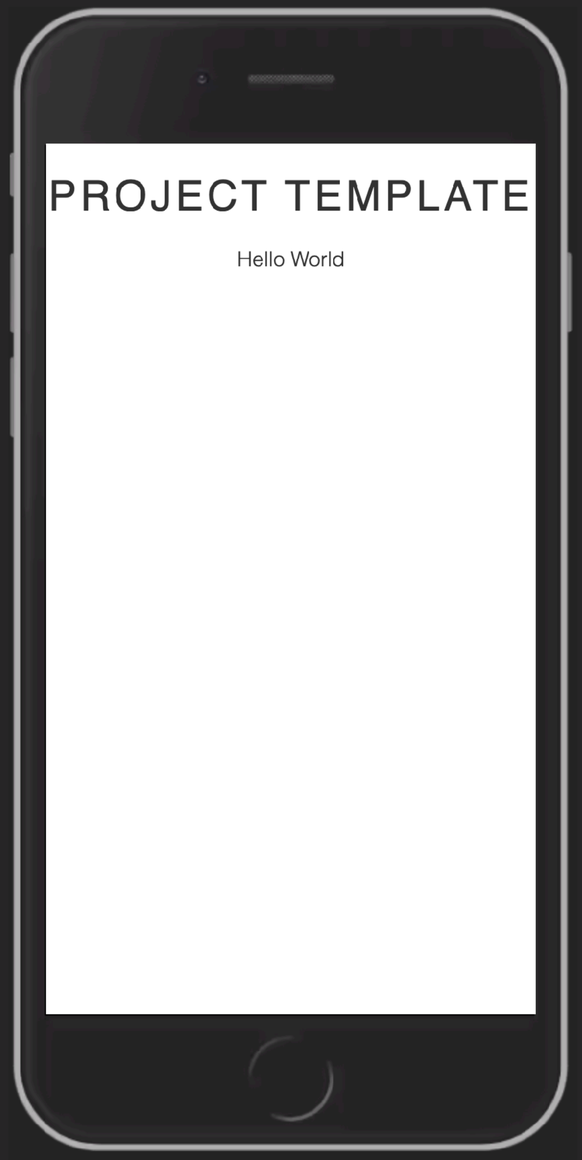
Store data in the memory

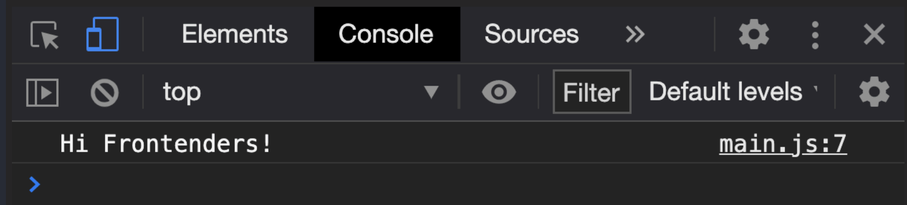
```
let message = "Hello!";
console.log(message);

message = "World!";
console.log(message);
```

```js
// 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;
```

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

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

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

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

// changing the value of the variable
message = "Hello World";

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





var **vs** let

THE DIFFERENCE IS THE SCOPING

VAR IS FUNCTION-WIDE OR GLOBAL SCOPE

LET IS BLOCK SCOPED

VAR TOLERATES REDECLARATION

https://javascript.info/variables
https://javascript.info/var

```
// Example 1
// “var” has no block scope
if (true) {
  var test1 = true; // use "var" instead of "let"
}
console.log(test1); // true, the variable lives after if

// Example 2
if (true) {
  let test2 = true; // use "let"
}
console.log(test2); // Error: test is not defined

// Example 3
for (var i = 0; i < 10; i++) {
  // ...
}
console.log(i); // 10, "i" is visible after loop, it's a global variable
```

var-vs-let

```
// “var” tolerates redeclarations
var user1 = "Pete";
var user1 = "John"; // this "var" does nothing (already declared)
// ...it doesn't trigger an error
console.log(user1); // John

let user2;
let user2; // SyntaxError: 'user' has already been declared
```

var-vs-let

CONST

Const is an unchanging variable.

```
const myBirthday = "12-03-1990";
myBirthday = "12-03-1989";
// Uncaught TypeError: can't reassign the constant!
```

`const` cannot be reassigned.

If you try to, an `error` will be thrown.

Const can’t be reassigned

```
const myBirthday = "12-03-1990";
myBirthday = "12-03-1989"; // Uncaught TypeError: can't reassign the constant!

const person = {
    name: "Kasper",
    mail: "kato@eaaa.dk",
    age: 32
};

person.age = 33; // no error

person = {
    name: "Rasmus",
    mail: "race@eaaa.dk",
    age: 31
}; // Uncaught TypeError: can't reassign the constant!
```

Use `let` & `const` instead of `var`

https://javascript.info/variables
https://javascript.info/var

Name things right

Talking about variables, there's one more extremely important thing.

A variable name should have a clean, obvious meaning, describing the data that it stores.

Variable naming is one of the most important and complex skills in programming. A quick glance at variable names can reveal which code was written by a beginner versus an experienced developer.

In a real project, most of the time is spent modifying and extending an existing code base rather than writing something completely separate from scratch. When we return to some code after doing something else for a while, it's much easier to find information that is well-labeled. Or, in other words, when the variables have good names.

Please spend time thinking about the right name for a variable before declaring it. Doing so will repay you handsomely.

Some good-to-follow rules are:

- Use human-readable names like `userName` or `shoppingCart`.
- Stay away from abbreviations or short names like `a`, `b`, `c`, unless you really know what you're doing.
- Make names maximally descriptive and concise. Examples of bad names are `data` and `value`. Such names say nothing. It's only okay to use them if the context of the code makes it exceptionally obvious which data or value the variable is referencing.
- Agree on terms within your team and in your own mind. If a site visitor is called a "user" then we should name related variables `currentUser` or `newUser` instead of `currentVisitor` or `newManInTown`.

https://javascript.info/variables#name-things-right

Objects

A set of named values

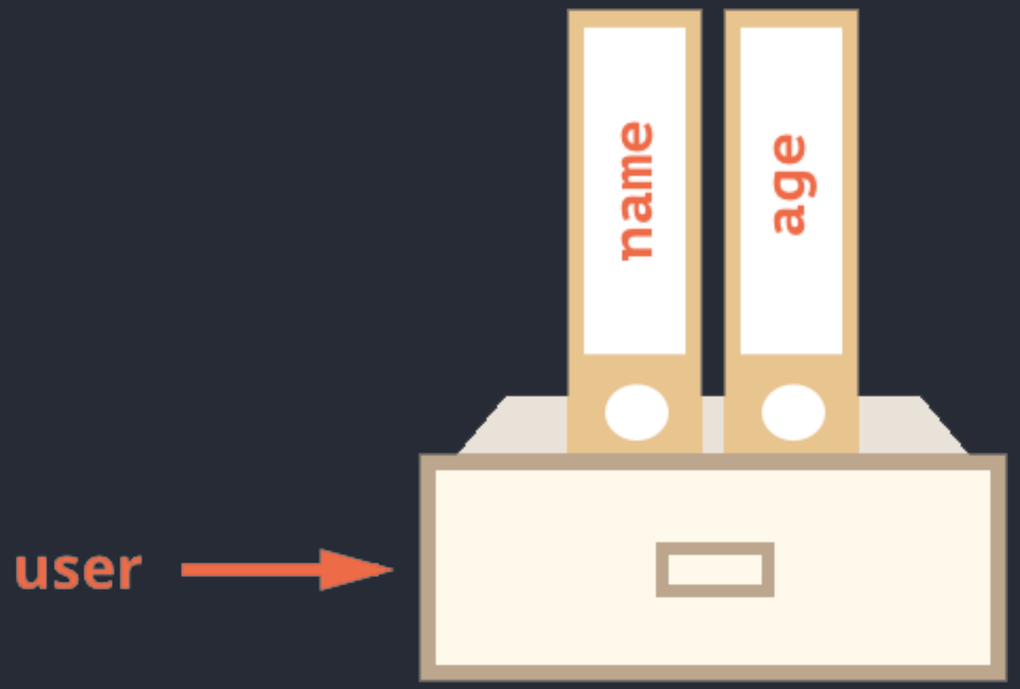
Objects are used to store keyed collections of various data

Containers for named values called properties. A property is a “`key: value`” pair

Objects

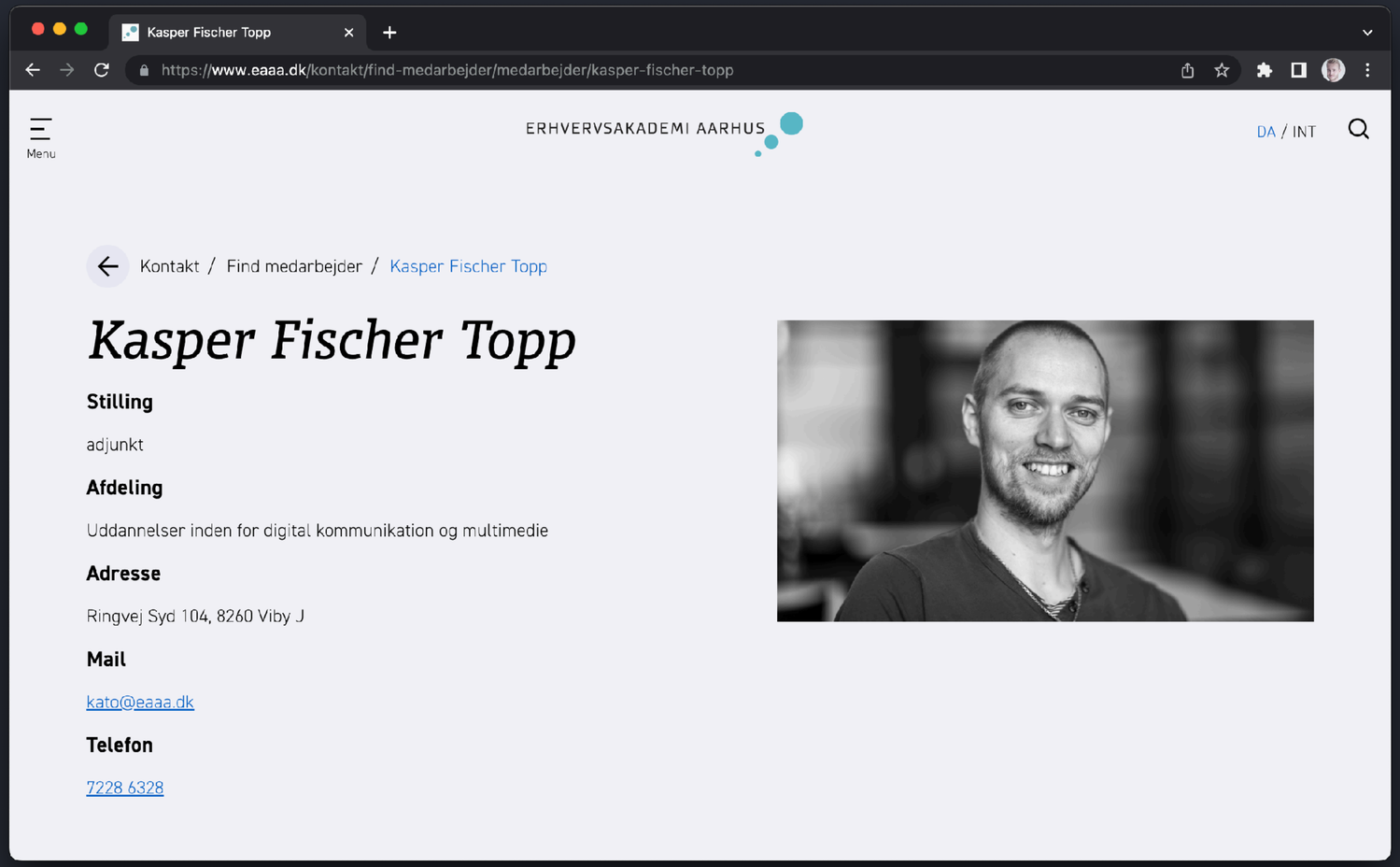
A set of named values

```
let user = {
  name: 'Alicia',
  age: 6
};

console.log(user.name +
  " is " + user.age +
  " years old.");
```




```
Alicia is 6 years old.          main.js:11
>
```

Objects



https://www.eaaa.dk/kontakt/find-medarbejder/medarbejder/kasper-fischer-topp

Objects

A set of named values

property

value

```
const mrBackend = {
  name: "Kasper Fischer Topp",
  mail: "kato@eaaa.dk",
  phone: "72286328",
  position: "Lecturer",
  favTechnologies: ["PHP", "SQL"]
};
```

netflix.com

NETFLIX

Hvem ser?

Personen der rent faktisk betaler for profilen

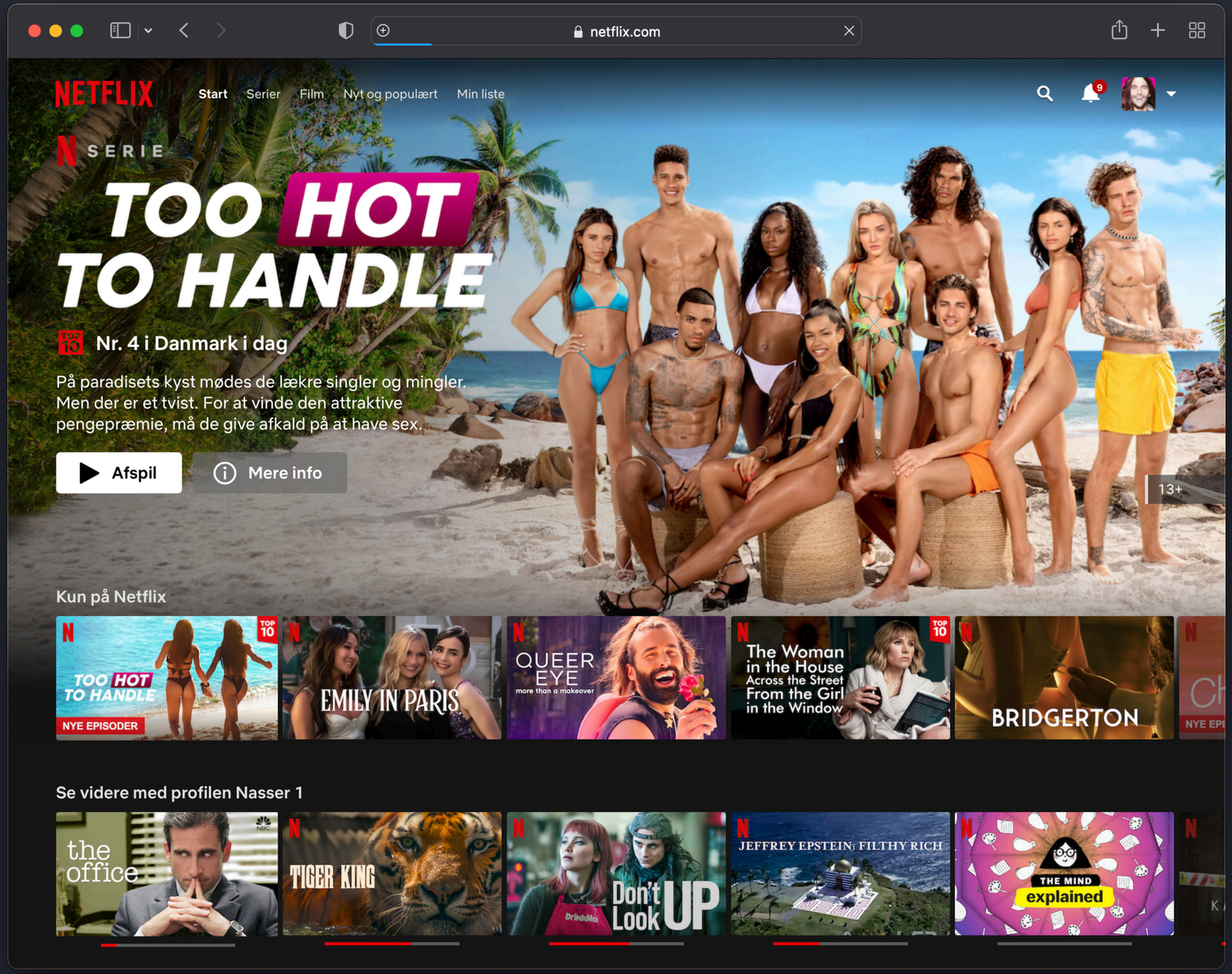
Nasser 1

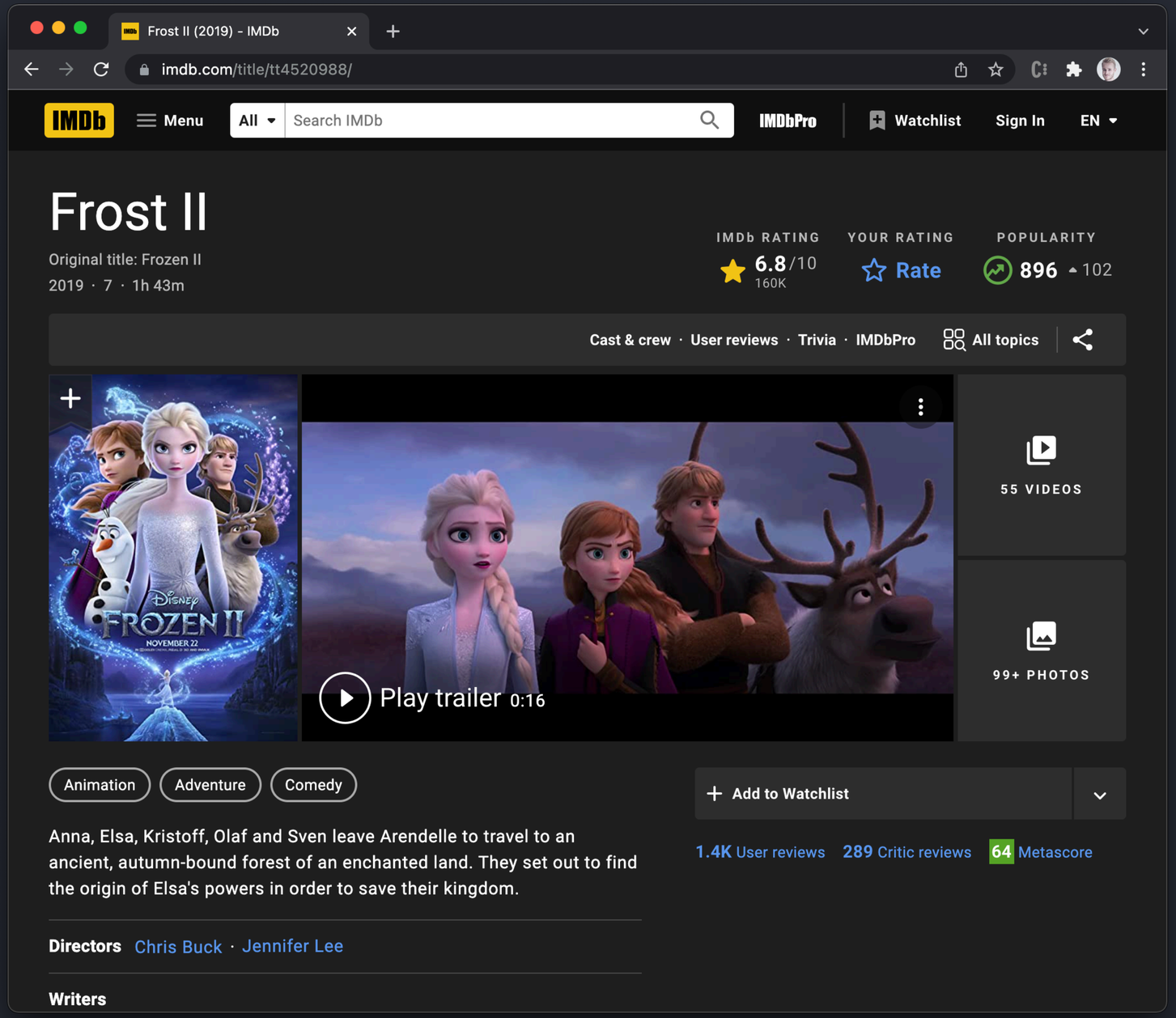
Nasser 2

Nasser 3

Nasser 4 Khader

Administrer profiler


netflix.com
NETFLIX
Start
Serier
Film
Nyt og populært
Min liste
SERIE
TOO HOT TO HANDLE
Nr. 4 i Danmark i dag
På paradisets kyst mødes de lækre singler og mingler. Men der er et tvist. For at vinde den attraktive pengepræmie, må de give afkald på at have sex.
Afspil
Mere info
13+
Kun på Netflix
NYE EPISODER
EMILY IN PARIS
QUEER EYE
more than a makeover
The Woman in the House Across the Street From the Girl in the Window
BRIDGERTON
Se videre med profilen Nasser 1
the office
TIGER KING
Don't Look UP
JEFFREY EPSTEIN: FILTHY RICH
THE MIND
explained

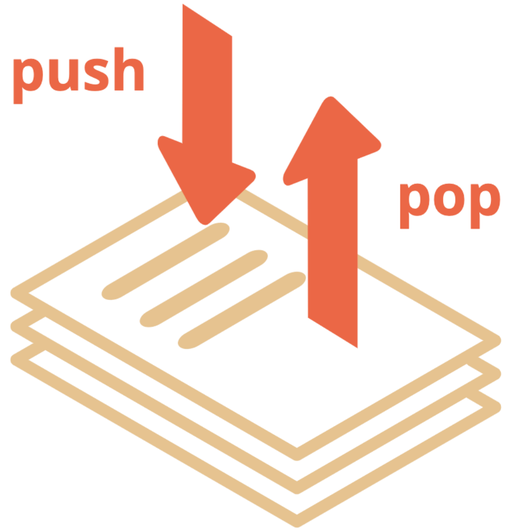



```
let movie = {
  title: "Frozen 2",
  description: "Elsa the Snow Queen has a
  trailer: "https://www.youtube.com/embed
  length: "1h 43m",
  year: "2019"
}
```

Arrays

Collections

Ordered collection of values or objects



An array is a way to hold more than one value at a time we have a 1st, a 2nd, a 3rd, a 4th element and so on.

```
let todaysLecturers = [
  {
    name: "Kasper Fischer Topp",
    mail: "kato@eaaa.dk",
    phone: "72286328",
    position: "Lecturer",
    favTechnologies: ["PHP", "SQL"],
    nickname: "Mr. Backend"
  },
  {
    name: "Rasmus Cederdorff",
    mail: "race@eaaa.dk",
    phone: "72286318",
    position: "Lecturer",
    favTechnologies: ["JavaScript"],
    nickname: "Mr. Frontend"
  }
];
```

First element

Second element

Arrays

Rasmus Cederdorff

Position:
Lecturer

Department/p
Multimedia De
Digital Concep

Address:
Ringvej Syd 1(

Mail:
race@baaa.dk

Phone:
7228 6318

Michael Hvidtfeldt

Position:
Lecturer

Department/p
Multimedia De

Address:
Ringvej Syd 1(

Mail:
mhv@baaa.dk

Phone:
7228 6328

Birgitte Kirk Iversen

Position:
Senior Lecturer

Department/programme:
Multimedia Design

Address:
Sønderhøj 30, 8260 Viby J

Mail:
bki@baaa.dk

Phone:
7228 6316

```
main.js:21
▼ (3) [{…}, {…}, {…}] i
  ▶ 0: {name: "Birgitte Kirk Iversen", mail: "bki@baaa…
  ▶ 1: {name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk…
  ▶ 2: {name: "Rasmus Cederdorff", mail: "race@baaa.dk…
    length: 3
  ▶ __proto__: Array(0)
main.js:22
▶ {name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk"}
3
main.js:23
```

```
let teachers = [{
    name: "Birgitte Kirk Iversen",
    mail: "bki@baaa.dk"
  },
  {
    name: "Michael Hvidtfeldt",
    mail: "mhv@baaa.dk"
  },
  {
    name: "Rasmus Cederdorff",
    mail: "race@baaa.dk"
  }
];

console.log(teachers);
console.log(teachers[1]);
console.log(teachers.length);
```

Teachers

http://127.0.0.1:5501/array-teachers/index.html

Teachers

Birgitte Kirk Iversen

Senior Lecturer
bki@baaa.dk

Maria Louise Bendixen

Senior Lecturer
mlbe@baaa.dk

Kim Elkjær Marcher-Jepsen

Lecturer
kije@baaa.dk

Rasmus Cederdorff

Lecturer
race@baaa.dk

Console

top

Filter

Default levels

1 Issue: 1

main.js:46

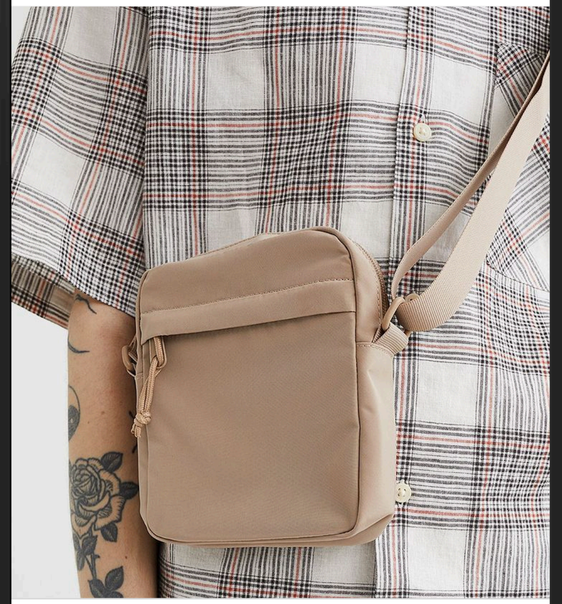
```
(4) [{…}, {…}, {…}, {…}]
  0:
    address: "Sønderhøj 30, 8260 Viby J"
    department: "Multimedia Design"
    img: "https://www.eaaa.dk/media/u4gorzs
    initials: "bki"
    mail: "bki@baaa.dk"
    name: "Birgitte Kirk Iversen"
    phone: "72286316"
    position: "Senior Lecturer"
    [[Prototype]]: Object
  1: {name: 'Maria Louise Bendixen', initia
  2: {name: 'Kim Elkjær Marcher-Jepsen', in
  3: {name: 'Rasmus Cederdorff', initials:
  length: 4
  [[Prototype]]: Array(0)
```

array-teachers

Instagram

Your story | fhergram | vladaafa... | kristybra... | gabriel

hm_man



Instagram

Liked by **praaajyot** and **others**

hm_man Shades, style and summer vibes, they all fit inside. #HMMan... more

View all 5 comments

44 MINUTES AGO

programmerhumor_io

I Am Devloper
@iamdevloper

Elon Musk: I'm putting people on Mars!
Developers: Fantastic, more timezones to support

cederdorff

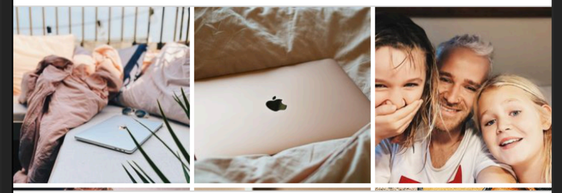
cederdorff

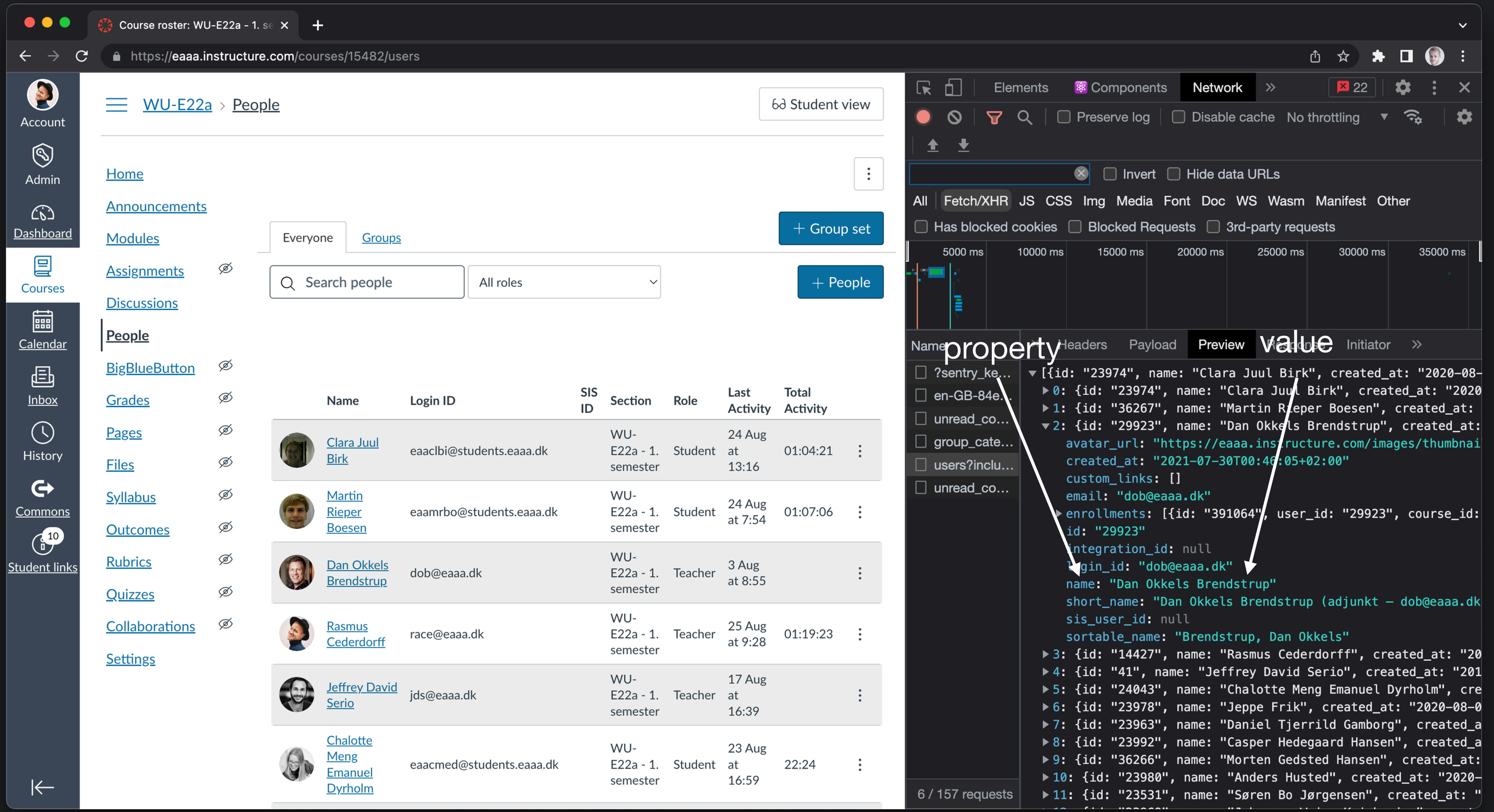
Edit Profile

Rasmus Cederdorff
Senior Lecturer & Web App Developer
Alicia & Ida
Aarhus, Denmark
cederdorff.com

AMSTER... | LONDON ... | Karla & Pi... | @villacph | IDA

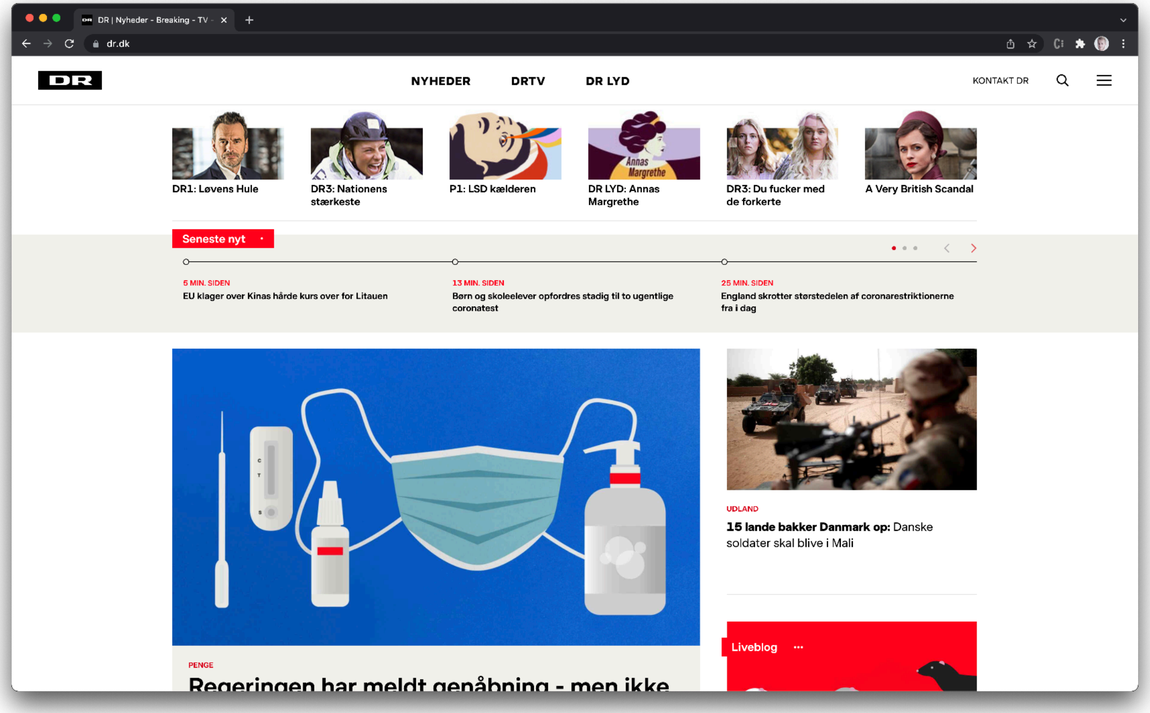
181	5,441	635
posts	followers	following

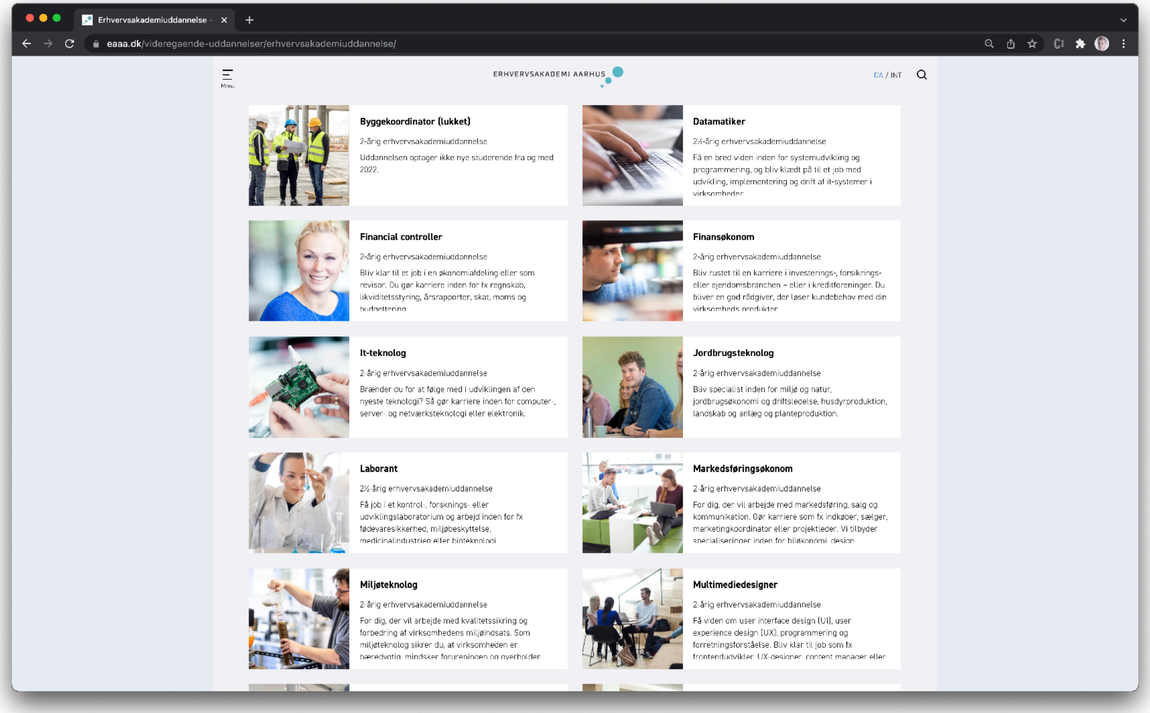




eaaa.instructure.com/api/v1/courses/15482/users?include_inactive=true&include%5B%5D=avatar_url&include%5B%5D=email&include%5B%5D=observed_users&per_page=50

Objects? Arrays?





It’s all objects & arrays!

Data Types & Data Structures

Objects & Arrays

Arrays

Loops

```
for (let teacher of teachers) {
  console.log(teacher);
}
```

```
▶ {name: "Birgitte Kirk Iversen", mail: "bki@baaa.dk"}    main.js:20
▶ {name: "Michael Hvidtfeldt", mail: "mhv@baaa.dk"}       main.js:20
▶ {name: "Rasmus Cederdorff", mail: "race@baaa.dk"}       main.js:20
```

For of loop

iterate over arrays or other iterable objects

https://scrimba.com/learn/introductiontojavascript/for-loops-cMMM8U9

https://scrimba.com/learn/introductiontojavascript/challenge-for-loops-cPkpJrcv

Loops

```
for (const familyMember of familyMembers) {
  console.log(familyMember);
}

for (let index = 0; index < familyMembers.length; index++) {
  const familyMember = familyMembers[index];
  console.log(familyMember);
}
```

HTTPS://WWW.W3SCHOOLS.COM/JS/JS_LOOP_FOR.ASP

HTTPS://JAVASCRIPT.INFO/ARRAY#LOOPS

HTTPS://JAVASCRIPT.INFO/WHILE-FOR

Array methods

https://javascript.info/array-methods#filter

filter

The `find` method looks for a single (first) element that makes the function return `true`.

If there may be many, we can use arr.filter(fn).

The syntax is similar to `find`, but `filter` returns an array of all matching elements:

```
let results = arr.filter(function(item, index, array) {
  // if true item is pushed to results and the iteration continues
  // returns empty array if nothing found
});
```

For instance:

```
let users = [
  {id: 1, name: "John"},
  {id: 2, name: "Pete"},
  {id: 3, name: "Mary"}
];

// returns array of the first two users
let someUsers = users.filter(item => item.id < 3);

alert(someUsers.length); // 2
```

Transform an array

Let's move on to methods that transform and reorder an array.

map

The arr.map method is one of the most useful and often used.

It calls the function for each element of the array and returns the array of results.

Array Methods

.FIND()

.MAP()

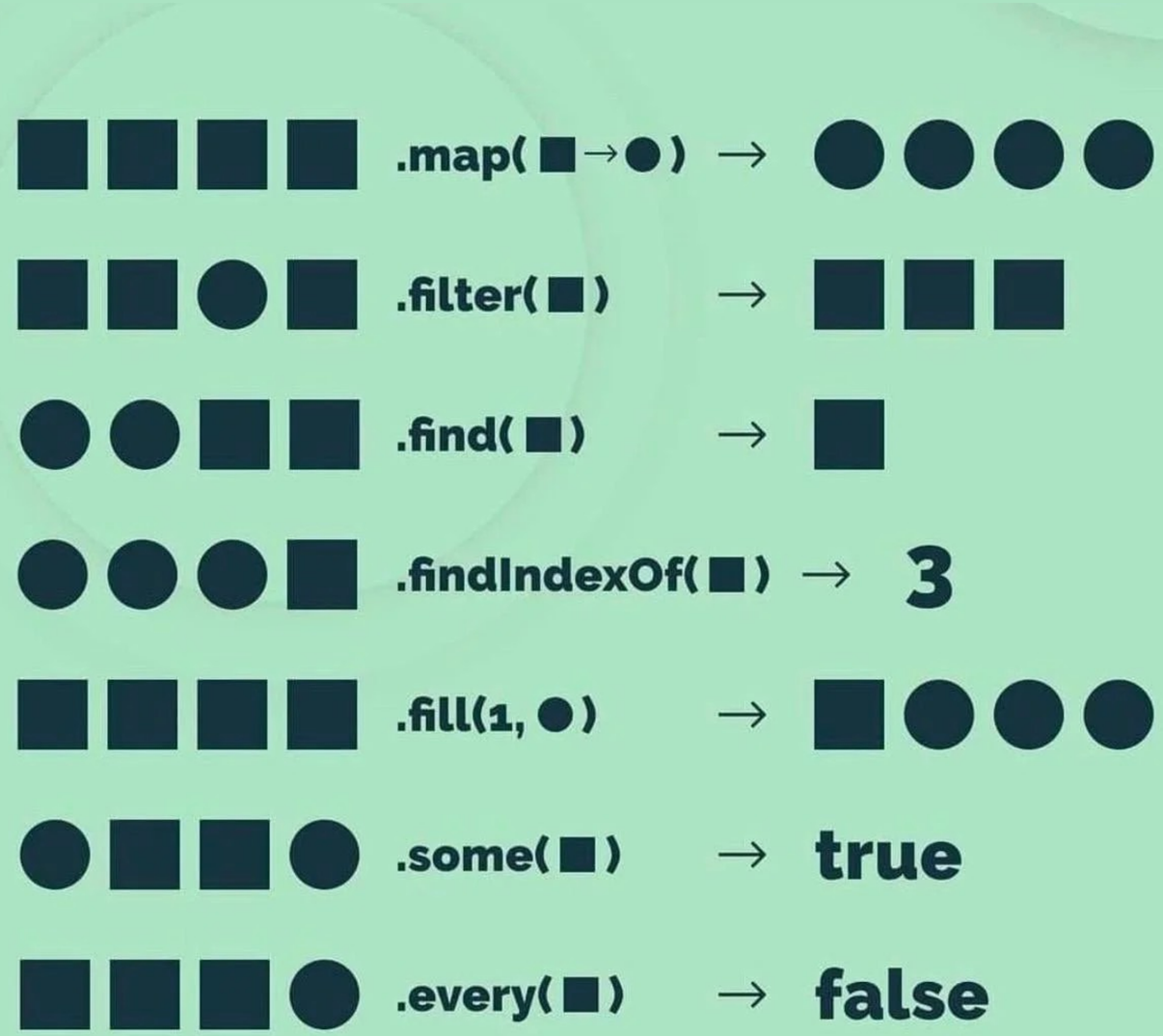
.REDUCE()

.FILTER()

.SORT()

.CONCAT()

...

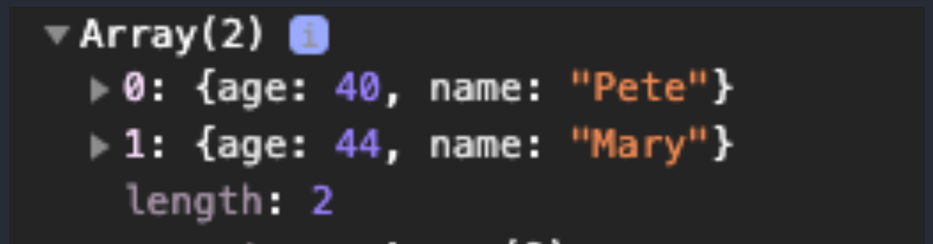


https://javascript.info/array-methods
https://medium.com/@mandeepkaur1/a-list-of-javascript-array-methods-145d09dd19a0

Arrays

.filter()

```
let users = [
  { age: 35, name: "John" },
  { age: 40, name: "Pete" },
  { age: 44, name: "Mary" }
];

// returns array of with users older than 39
let someUsers = users.filter(item => item.age > 39);

console.log(someUsers);
```



FUNCTIONS

...A BLOCK OF CODE TO PERFORM A SPECIFIC TASK & TO MAKE OUR CODE REUSABLE

A WAY OF STORING OUR CODE SO WE CAN USE IT AGAIN AND AGAIN AND REUSE IT

BEST PRACTICE: WRITE REUSABLE CODE

https://scrimba.com/p/pPPeCy/cpKZdcQ

FUNCTIONS

FUNCTION DECLARATION

```
function log(message) {
  console.log(message);
}

log("Hi Frontenders!");
```

https://javascript.info/function-basics

FUNCTIONS

FUNCTION DECLARATION

```
console.log("Hi Frontenders!");
console.log("Good job!");
console.log("I'm testing something!");
console.log("Hola");
```

```
function log(message) {
  console.log(message);
}

log("Hi Frontenders!");
log("Good job!");
log("I'm testing something!");
log("Hola");
```

https://javascript.info/function-basics

FUNCTIONS

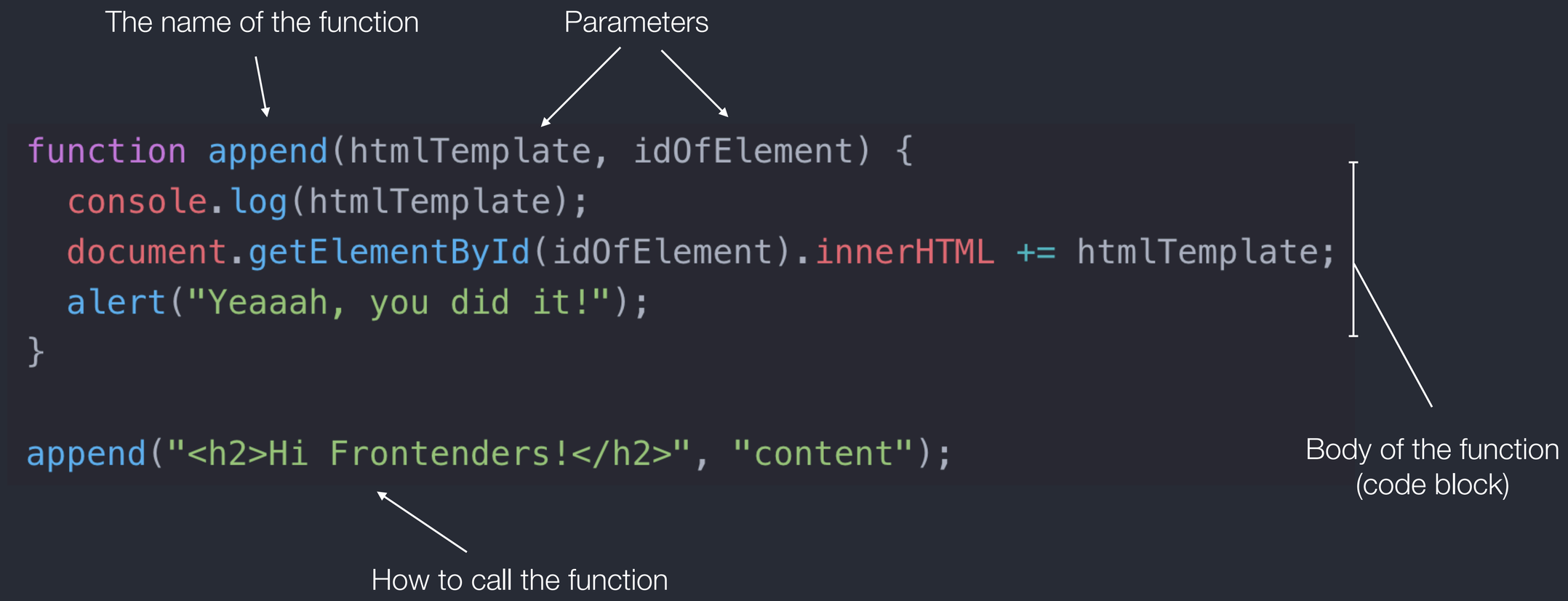
FUNCTION DECLARATION

```
function append(htmlTemplate, idOfElement) {
  console.log(htmlTemplate);
  document.getElementById(idOfElement).innerHTML += htmlTemplate;
  alert("Yeaaah, you did it!");
}

append("<h2>Hi Frontenders!</h2>", "content");
```

https://javascript.info/function-basics

FUNCTIONS

FUNCTION DECLARATION



https://javascript.info/function-basics

FUNCTIONS

ARRAY & LOOPS

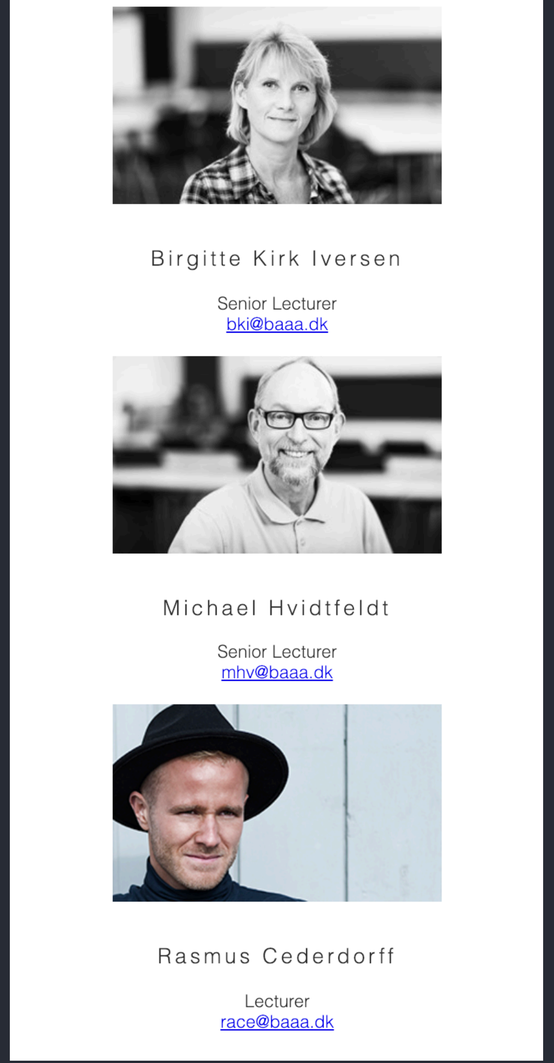
The name of the function

Parameters

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}

appendTeachers(teachers);
```

Body of the function
(code block)

How to call the function



FUNCTIONS

```
function logPersons(persons) {
  for (var i = 0; i < persons.length; i++) {
    console.log(persons[i]);
  }
}
```

FUNCTION DECLARATION

```
const logPersons = function(persons) {
  for (var i = 0; i < persons.length; i++) {
    console.log(persons[i]);
  }
}
```

FUNCTION EXPRESSION

```
const logPersons = (persons) => {
  for (var i = 0; i < persons.length; i++) {
    console.log(persons[i]);
  }
}
```

ARROW FUNCTION

`TEMPLATE STRING`

BACKTICK STRING / TEMPLATE LITERALS

...EMBED VARIABLES AND EXPRESSIONS IN A STRING

https://scrimba.com/p/p4Mrt9/c4vJdha

`TEMPLATE STRING`

BACKTICK STRING / TEMPLATE LITERALS

- Extended functionality
- Simplifies concatenating strings
- Embed values and expression into a string with `${...}`
- Simplifies the syntax and the reading
- Let us create more readable HTML templates

```
let name = "Alicia";
console.log(`Hello, ${name}`);
```

```
Hello, Alicia                                    main.js:8
>
```

`TEMPLATE STRING`

```
let name = "Alicia";
let age = 6;

console.log(name + " is " + age + " years old.");

console.log(`${name} is ${age} years old.`);
```

```
Alicia is 6 years old.          main.js:10
Alicia is 6 years old.          main.js:12
>
```

`TEMPLATE STRING`

REGULAR STRING EXPRESSION

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}
```

TEACHERS

Birgitte Kirk Iversen

Senior Lecturer
bki@baaa.dk

Michael Hvidtfeldt

Senior Lecturer
mhv@baaa.dk

Rasmus Cederdorff

Lecturer
race@baaa.dk

`TEMPLATE STRING`

...EMBED VARIABLES AND EXPRESSIONS IN A STRING

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=
      "<article>" +
      "<img src='" + teacher.img + "'>" +
      "<h3>" + teacher.name + "</h3>" +
      teacher.position + "<br>" +
      "<a href='mailto:" + teacher.mail + "'>" + teacher.mail + "</a>" +
      "</article>";
  }
}
```

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML += `
      <article>
        <img src='${teacher.img}'>
        <h3>${teacher.name}</h3>
        ${teacher.position}<br>
        <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
      </article>`;
  }
}
```

`ES6 STRING HTML`

https://marketplace.visualstudio.com/items?itemName=hjb2012.vscode-es6-string-html

```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML += `
      <article>
        <img src='${teacher.img}'>
        <h3>${teacher.name}</h3>
        ${teacher.position}<br>
        <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
      </article>`;
  }
}
```

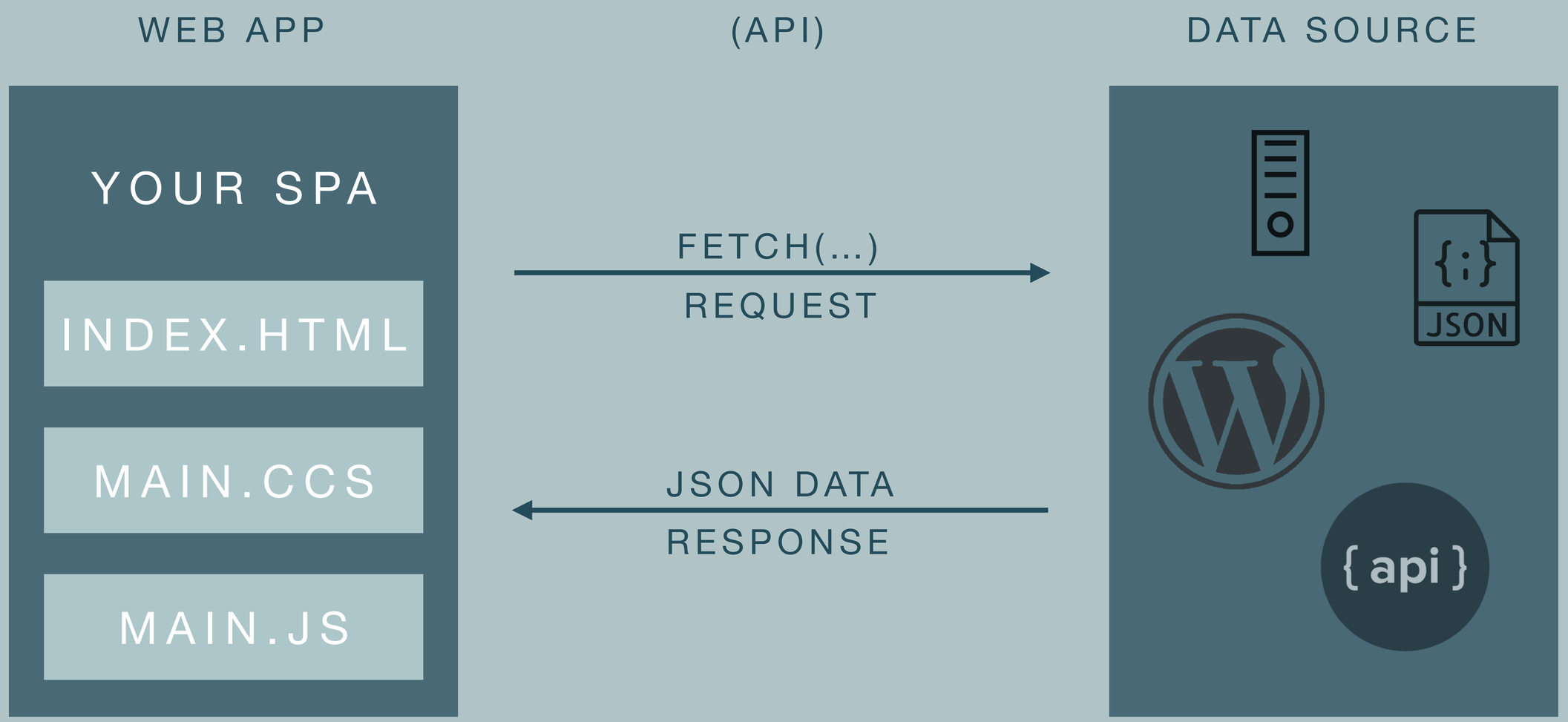
```
function appendTeachers(teachers) {
  for (let teacher of teachers) {
    console.log(teacher);
    document.querySelector("#grid-teachers").innerHTML +=/*html*/ `
    <article>
      <img src='${teacher.img}'>
      <h3>${teacher.name}</h3>
      ${teacher.position}<br>
      <a href='mailto:${teacher.mail}'>${teacher.mail}</a>
    </article>`;
  }
}
```

FETCH(...)

HTTP REQUEST IN JAVASCRIPT

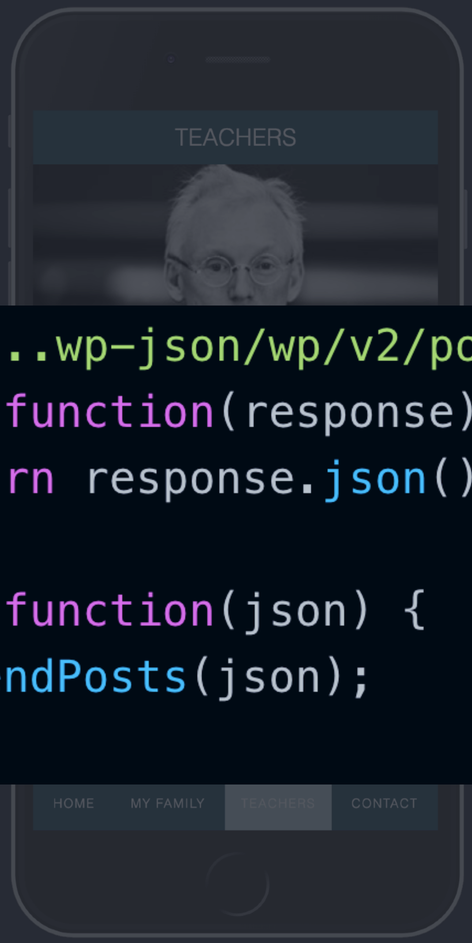
...A WAY TO GET & POST DATA FROM & TO A DATA SOURCE

FETCH


WEB APP
(API)
DATA SOURCE
YOUR SPA
INDEX.HTML
MAIN.CCS
MAIN.JS
FETCH(...)
REQUEST
JSON DATA
RESPONSE
JSON
{ api }

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"
  }
]
```

```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"
  }
]
```

```
/*
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

... IN THREE DIFFERENT WAYS

```
// 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/

Computer science student

Senior developer, 10+ years experience

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

React

React slides in Canvas/ GitHub

Code Every Day