

# Webudvikling Frontend

Intro, HTML, semantics, accessibility & performance

Webudvikling

**kea**  
KØBENHAVNS ERHVERVSAKADEMI

# Dagens Formål

- Introducere dig til faget
- Opsætte de vigtigste værktøjer
- Hands-on med moderne semantisk HTML
- Viden om performance og Web Dev Tools,
- samt begreber vi skal arbejde igen og igen

# Agenda

- Introduktion til Webudvikling Frontend
- Værktøjer - Frontend Tooling
- Client-server & HTTP
- Moderne & Semantisk HTML
- Source Code Management
- Performance

# Fagmodulets Formål

... at gøre dig i stand til at udvikle dynamiske webapplikationer - fokus på client-side. Du vil lære om moderne web- og frontendteknologier.

Webudvikling

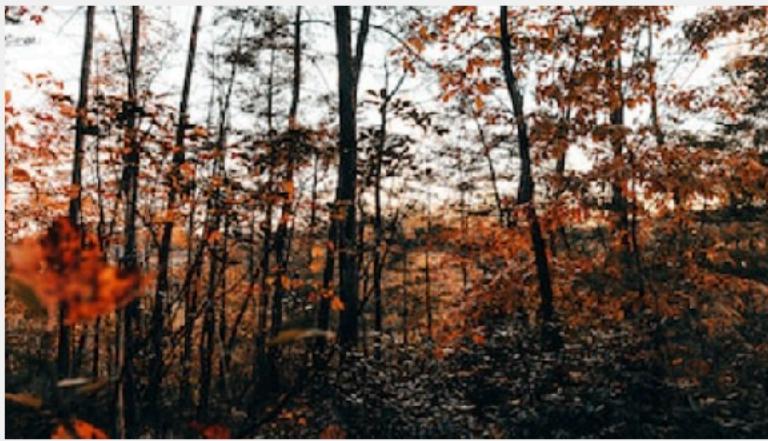
**kea**  
KØBENHAVNS ERHVERVSAKADEMI

# React CRUD App

React Firebase Post App https://race-react-firebase.web.app/

POSTS CREATE FAVORITES PROFILE

Maria Louise Bendixen  
Senior Lecturer



Dolor sint quo a velit explicabo  
quia namen

Eos qui et ipsum ipsam suscipit aut sed omnis non  
odio expedita earum mollitia molestiae aut atque  
rem suscipit nam impedit esse

[REMOVE FROM FAVORITES](#)

Jes Arbov  
Lecturer



Dolorem eum magni eos aperiam  
quia

Ut aspernatur corporis harum nihil quis provident  
sequi mollitia nobis aliquid molestiae perspiciatis  
et ea nemo ab reprehenderit accusantium quas  
voluptate dolores velit et doloremque molestiae

[ADD TO FAVORITES](#)

Birgitte Kirk Iversen  
Senior Lecturer



Magnam facilis autem

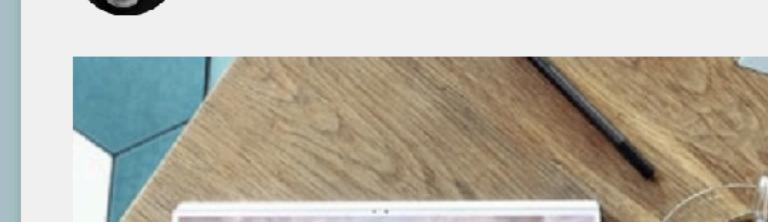
Dolore placeat quibusdam ea quo vitae magni quis  
enim qui quis quo nemo aut saepe quidem repellat  
excepturi ut quia sunt ut sequi eos ea sed quas

[ADD TO FAVORITES](#)

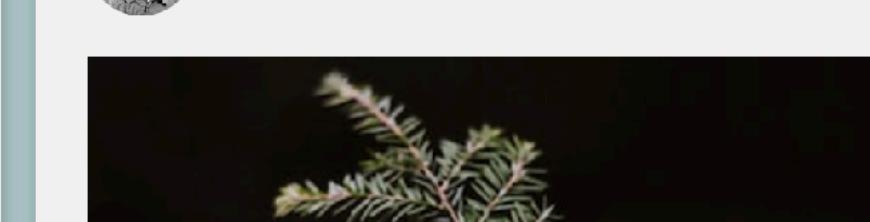
Kim Elkjær Marcher-Jepsen  
Senior Lecturer



Dan Okkels Brendstrup  
Lecturer



Morten Algy Bonderup  
Senior Lecturer



<https://react-rest-and-auth.web.app/>

# Indhold

## Webudvikling Frontend

- Moderne web og frontend teknologier
- Webapplikationer med teknologier som HTML, CSS, JavaScript og API'er
- Best practices indenfor frontend, design, accessibility og performance
- Et komponent baseret JavaScript Framework - React
- React og Single Page Apps
- Tools til frontend- og webudvikling, herunder NPM, bundlers, VS Code og debugging
- Webprotokoller, HTTP-verber, klient-server-arkitektur og BaaS
- Datakilder, REST og CRUD
- Brugervenlighed og brugertests

1. Intro, HTML, semantics, accessibility & performance
2. CSS, fonts, responsiveness & Design Principles
3. CSS Flexbox & intro to JavaScript
4. CSS Grid, JavaScript, the DOM & UX Laws
5. UI frameworks, JavaScript, forms & Usability Testing
6. JavaScript Modules & debugging
7. Async JavaScript & performance
8. Data Sources, JSON & APIs
9. Frontend Tooling & Intro to JS Frameworks
10. CSS Preprocessing & Intro to Component-Based Frameworks
11. A Component-Based Framework
12. A Component-Based Framework
13. Exam project
14. Exam project

# Undervisningsplan

[Find den her](#)



# Undervisningsform & Materiale

- Holdundervisning
- Kombination af teori, praktiske eksempler, cases og opgaver
- Visuelt materiale der komplementerer den til tider tørre programmeringsteori
- Materiale vil være på engelsk
- Tekster, videoer, tutorials og programmeringsøvelser som forberedelse

A photograph of a man and a woman looking at a laptop screen. The man is in the foreground, wearing a white t-shirt with an Adidas logo, and the woman is behind him, looking over his shoulder. They appear to be focused on the laptop screen, which is displaying some code or a technical interface.

I can teach everyone  
how to code!

... as long as you are eager to learn

Webudvikling

kea  
KØBENHAVNS ERHVERVSAKADEMI

I'm Rasmus Cederdorff  
Senior Lecturer  
Freelance Web App Developer

Programming, UI & UX

Frontend & Web Development  
“I speak JavaScript”

Websites, Webshops, Web Apps  
& Mobile Apps





I'm Rasmus Cederdorff  
Aarhus -> Copenhagen

2 Girls - Alicia & Ida

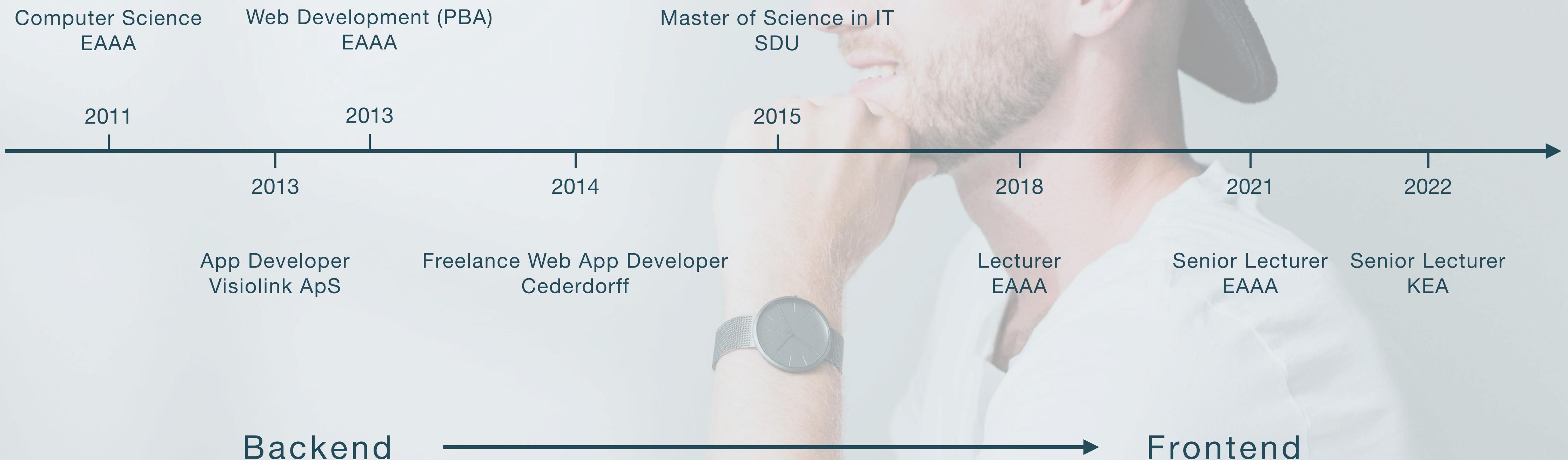
From Holstebro

I'm into sports

Love (⌚{}) gadgets, to take pictures  
& interior design projects

What's with my arm?

# Rasmus Cederdorff



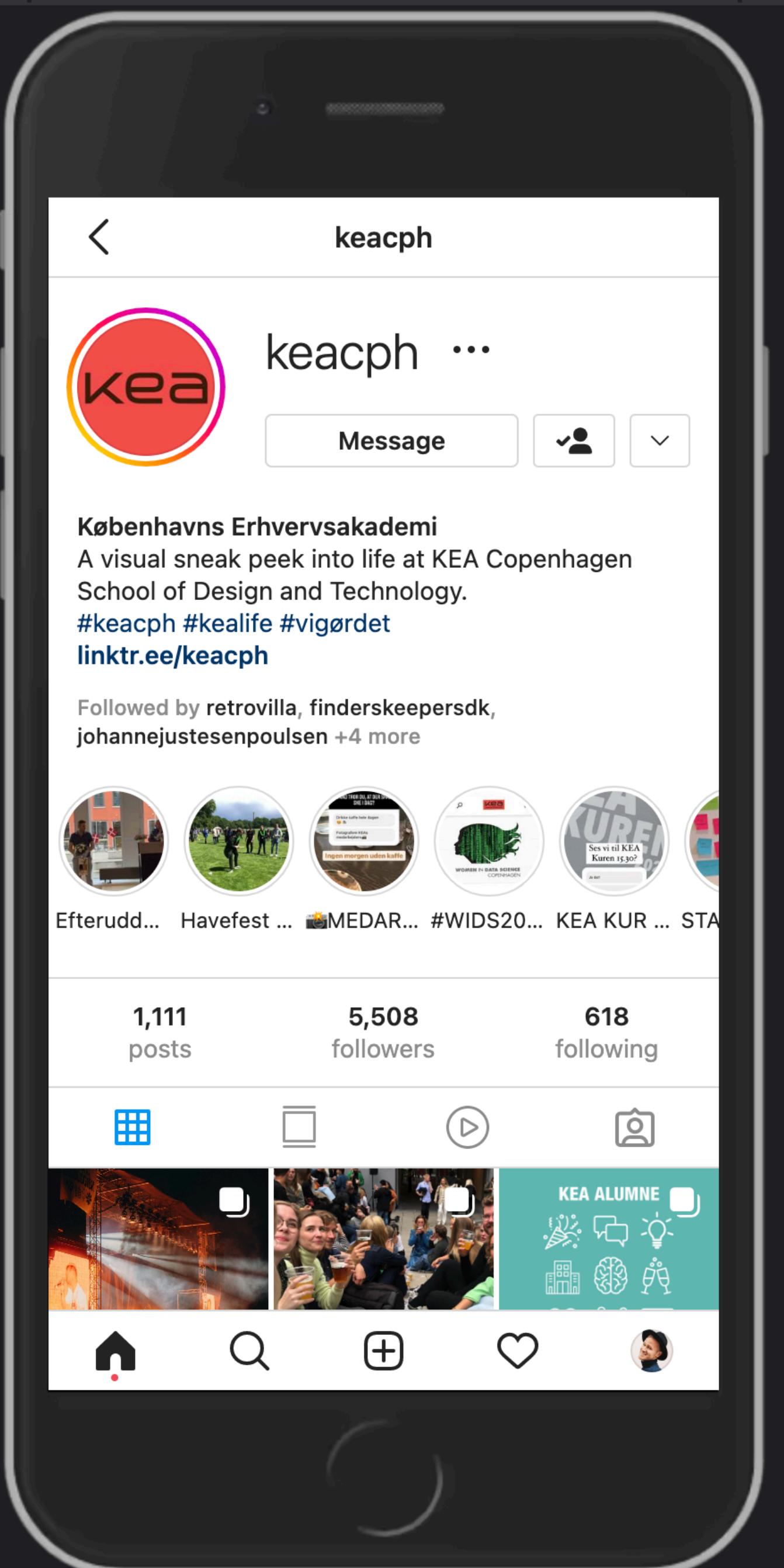
## BACKEND

DATABASE  
ARCHITECTURE  
PERFORMANCE  
SECURITY  
SCALABILITY

## FRONTEND

UI  
INTERACTIONS  
ARCHITECTURE  
PERFORMANCE  
SECURITY  
SCALABILITY

```
index.html
firebase_crud
firebase_crud_bootstrap
firebase_get_started
form_basic
form_basic_onchange
form_location
form_stepper
form_stepper_materiali
84 }
85 }
86 /*
87 * Fetches post data from my headless cms
88 */
89 function getPersons() {
90   fetch('http://headlesscms.cederdorff.com/wp-json/wp/v2/posts?_embed')
91     .then(function(response) {
92       return response.json();
93     })
94     .then(function(persons) {
95       persons;
96     })
97     .then(function(persons) {
98       let html = '';
99       persons.forEach(function(person) {
100         html += `<div>";
23                <div class='client'>
24                    <img alt='${getFeaturedImageUrl(client)}' alt='${client.title.rendered}'>
25                </div>
26                <div class='col-right even'>
27                    <div class='container'>
28                        <h3>${client.title.rendered}</h3>
29                        <h4>${client.meta.sub_title[0]}</h4>
30                        ${client.content.rendered}
31                        <div class='client-links'>
32                            <a href='${client.link.url}' target='_blank'>${client.meta.link_text[0]} <i class='ion-ios-arrow-forward'></i><i class='ion-ios-arrow-forward'></i></a>
33                        </div>
34                    </div>
35                </div>
36            } else { // index is odd
37                html += "<div class='client'>
38                    <div class='col-right'>
39                        <img src='${getFeaturedImageUrl(client)}' alt='${client.title.rendered}'>
40                    </div>
41                </div>
42            }
43        }
44    }
45
46    } else {
47        html += "<div class='client'>
48            <div class='col-right'>
49                <img src='${getFeaturedImageUrl(client)}' alt='${client.title.rendered}'>
50            </div>
51        </div>
52    }
53
54    appendClients(clients);
55
56    // Add the clients to the DOM
57    const clientsDiv = document.createElement('div');
58    clientsDiv.innerHTML = html;
59    document.body.appendChild(clientsDiv);
60
61    // Clean up the temporary clients array
62    clients = null;
63
64    // Fetch the next page of clients
65    fetch(nextPageUrl)
66        .then(function(response) {
67            return response.json();
68        })
69        .then(function(posts) {
70            appendClients(posts);
71        });
72
73    // Set the current page to the next one
74    currentPage++;
75
76    // If there are no more pages, stop the loop
77    if (currentPage > totalPages)
78        clearInterval(intervalId);
79
80    // Update the progress bar
81    const progress = document.querySelector('.progress');
82    const totalPosts = posts.length;
83    const currentPostIndex = (currentPage - 1) * postsPerPage + 1;
84    const progressValue = ((currentPostIndex / totalPosts) * 100).toFixed(2);
85    progress.style.width = progressValue + '%';
86
87    // Update the page number
88    const pageText = document.querySelector('.page');
89    pageText.textContent = `Page ${currentPage} of ${totalPages}`;
90
91    // Update the status bar
92    const statusBar = document.querySelector('.status-bar');
93    statusBar.textContent = `Page ${currentPage} of ${totalPages} | Total Posts: ${totalPosts}`;
94
95    // Update the footer
96    const footer = document.querySelector('.footer');
97    footer.textContent = `Powered by WordPress API`;
98
99    // Update the title
100   const title = document.querySelector('title');
101   title.textContent = `WP API Client`;
102
103   // Update the meta description
104   const metaDescription = document.querySelector('meta[name="description"]');
105   metaDescription.setAttribute('content', `WP API Client - Fetching posts from the WordPress API`);
106
107   // Update the meta keywords
108   const metaKeywords = document.querySelector('meta[name="keywords"]');
109   metaKeywords.setAttribute('content', `WP API Client, WordPress API, Fetch API, JavaScript`);
```



Name	Headers	Payload	Preview	Response	Initiator	Timing	Cookies
□ ?modules=PolarisBD...							
□ ?content_type=PROF...							
□ ?username=keacph							
□ reels_tray/							
□ timeline/							
□ ig_sso_users/							
□ logging_client_events							
□ bz?__d=dis							
□ falco							
□ bz?__a=1&__ccg=G...							
□ batch_fetch_web/							
□ get_encrypted_crede...							
□ highlights_tray/							
□ ?target_user_id=140...							
□ badge/							
□ story/							
□ bulk-route-definitions/							
□ bulk-route-definitions/							
□ ?query_hash=69cba4...							
□ bz?__a=1&__ccg=G...							
□ bz?__a=1&__ccg=G...							
□ bulk-route-definitions/							
□ bz?__a=1&__ccg=G...							
□ logging_client_events							
□ bz?__d=dis							

# Computer science student

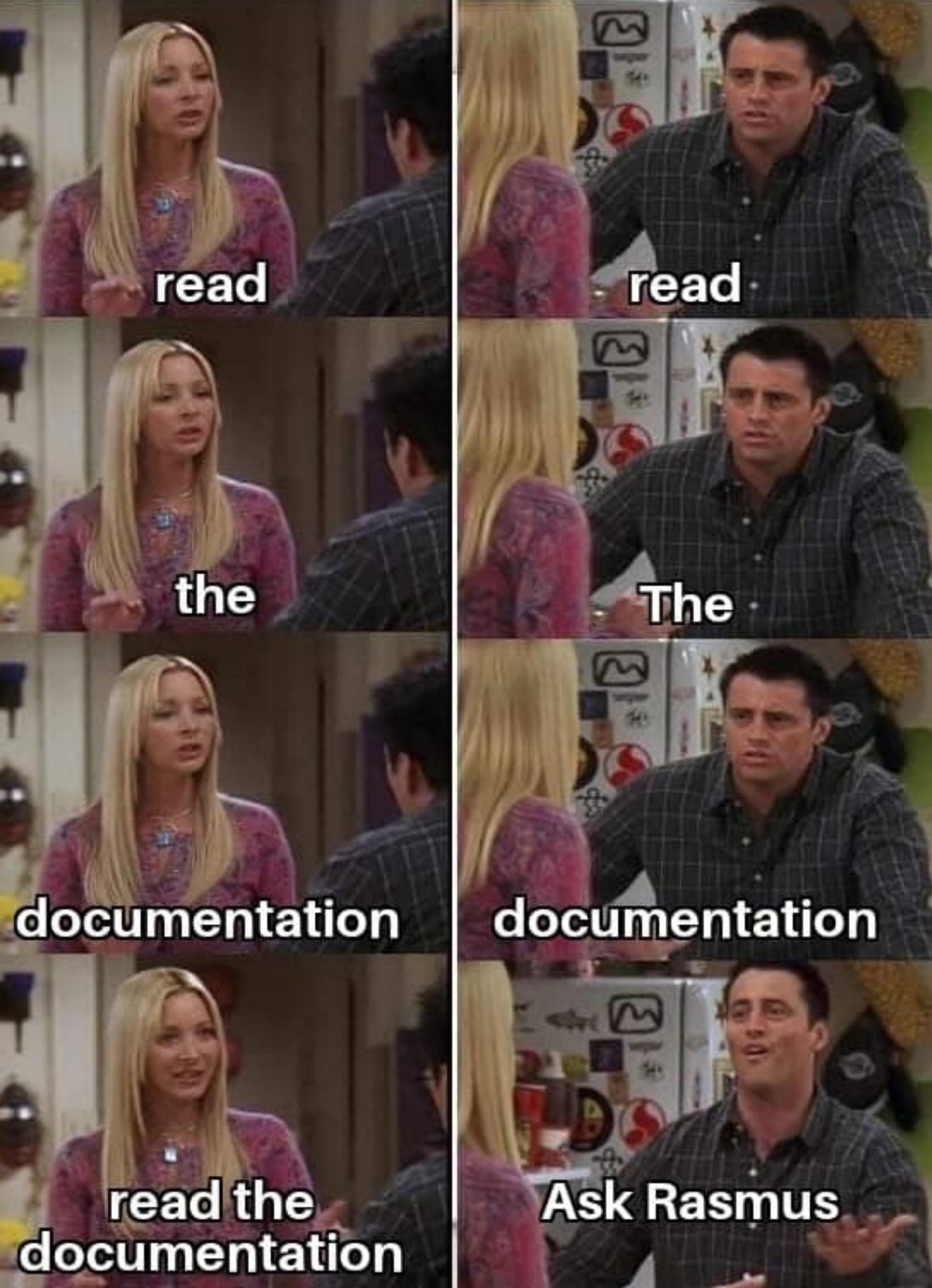


## Senior developer, 10+ years experience



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

# How to read the docs?



**“Learning to code requires a HUGE  
investment of time and energy.”**

<https://medium.com/martinsoft/learn-javascript-c1cca9db9015>



Code  
Every  
Day

# Eksamensopgave

Individuelt skal du udvikle en single page app (web app), der interagerer med en eller flere datakilder. Baserer sig på det du har lært gennem kurset: HTML, CSS, JavaScript, React, etc.

Mundtlig prøve med baggrund i en individuel synopsis samt programmeret produkt.

Eksempel på eksamensopgave

# Mere info

Website: [https://kompetence.kea.dk/kurser-fag/  
webudvikling-frontend](https://kompetence.kea.dk/kurser-fag/webudvikling-frontend)

Studieordning: [https://kompetence.kea.dk/studieordninger/  
Diplom\\_Webudvikling\\_Studieordning\\_2018\\_05.pdf](https://kompetence.kea.dk/studieordninger/Diplom_Webudvikling_Studieordning_2018_05.pdf)

Undervisningsplan

Webudvikling

**kea**  
KØBENHAVNS ERHVERVSAKADEMI

# Hvem er I?

[https://eaaa.padlet.org/race/  
web\\_diplom\\_hvem\\_er\\_i](https://eaaa.padlet.org/race/web_diplom_hvem_er_i)

- Navn
- By - hvor er du fra?
- Baggrund - uddannelse & job?
- Baggrund - HTML, CSS, JavaScript mm?
- Hvorfor Webudvikling Frontend-modulet?
- Forventninger til kurset?
- Hvad kan du særligt godt lide at arbejde med?
- Drømmejob?
- Meget gerne et profil billede.

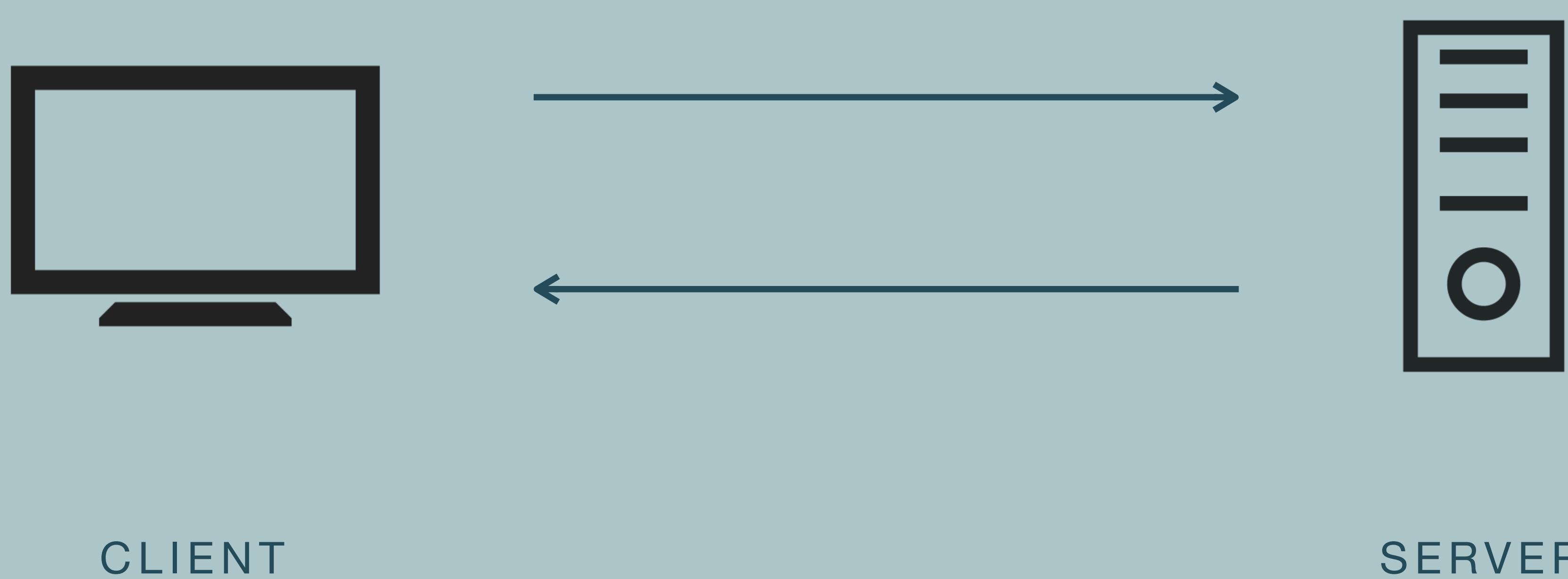
The background is a dark, slightly blurred photograph of a computer workstation. It features an iMac monitor with its characteristic rounded screen and a silver Apple keyboard. A white Apple Magic Mouse is positioned to the right of the keyboard. The overall aesthetic is clean and professional.

# Tools

Setup Tools & Dev Environment

# Client-Server Model

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



# Client-Server Model

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

**Browsers**  
Or any type of  
program or device

CLIENT

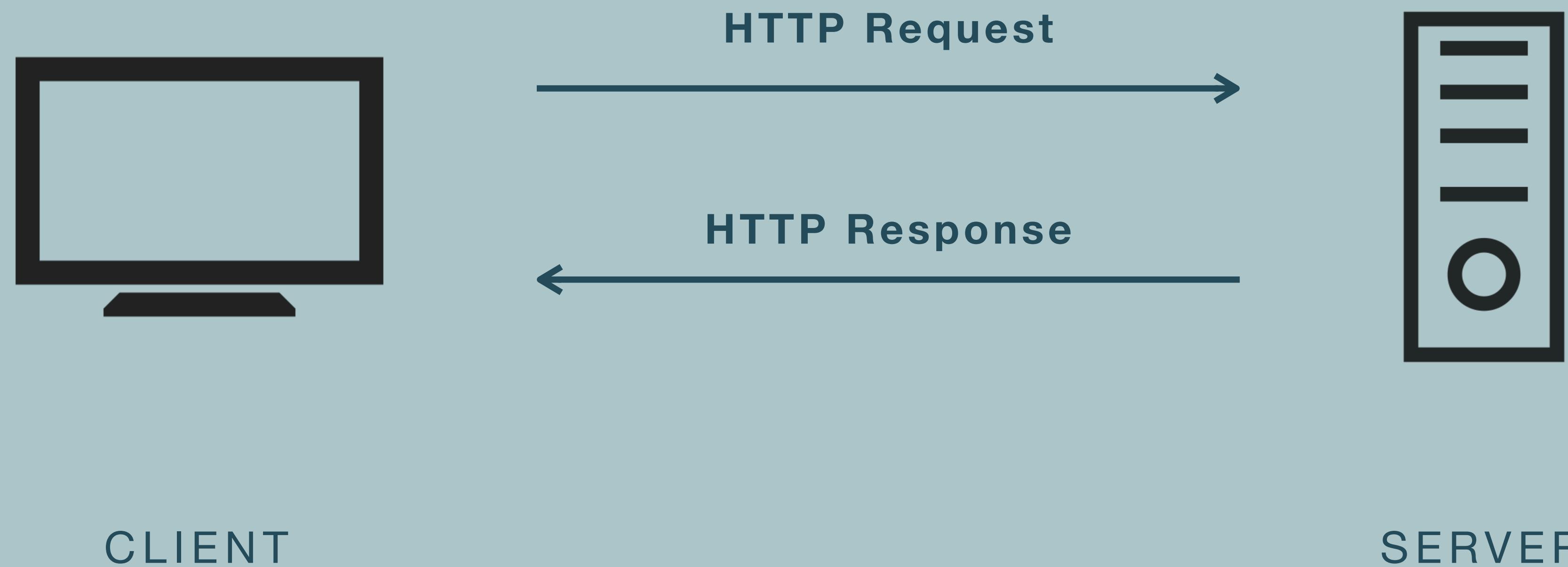


**Cloud Computers**  
Often computers  
in the cloud

SERVER

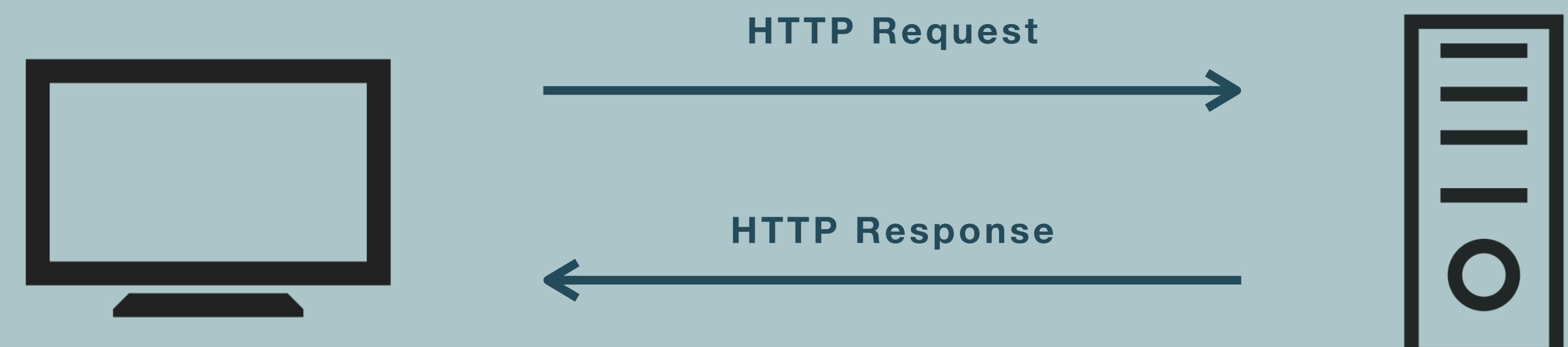
# Client-Server Model

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.



What is HTTP

Not Secure | w3schools.com/whatis/whatis\_http.asp

HTML CSS JAVASCRIPT SQL PYTHON

# HTTP Request / Response

Communication between clients and servers is done by **requests** and **responses**:

1. A client (a browser) sends an **HTTP request** to the web
2. A web server receives the request
3. The server runs an application to process the request
4. The server returns an **HTTP response** (output) to the browser
5. The client (the browser) receives the response

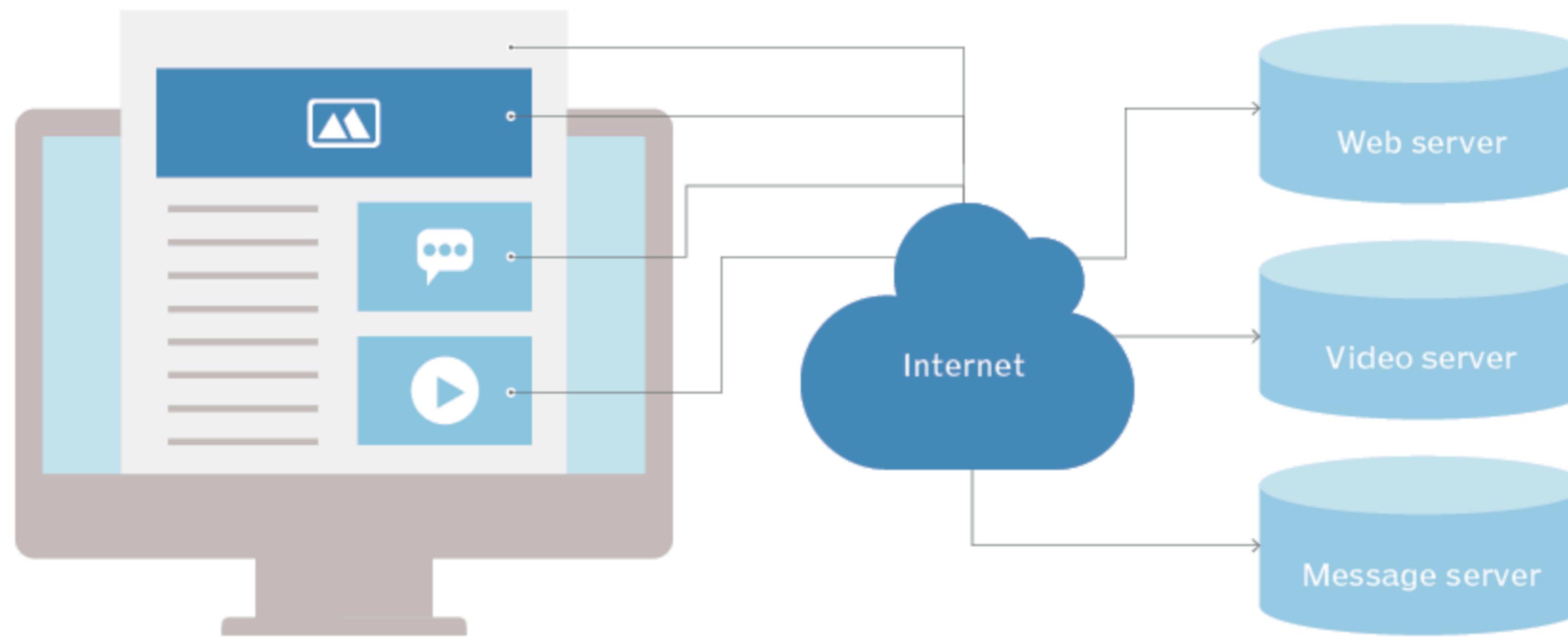
---

## The HTTP Request Circle

A typical HTTP request / response circle:

1. The browser requests an HTML page. The server returns an HTML file.
2. The browser requests a style sheet. The server returns a CSS file.
3. The browser requests an JPG image. The server returns a JPG file.
4. The browser requests JavaScript code. The server returns a JS file
5. The browser requests data. The server returns data (in XML or JSON).

# How HTTP Works



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

# Network Tab

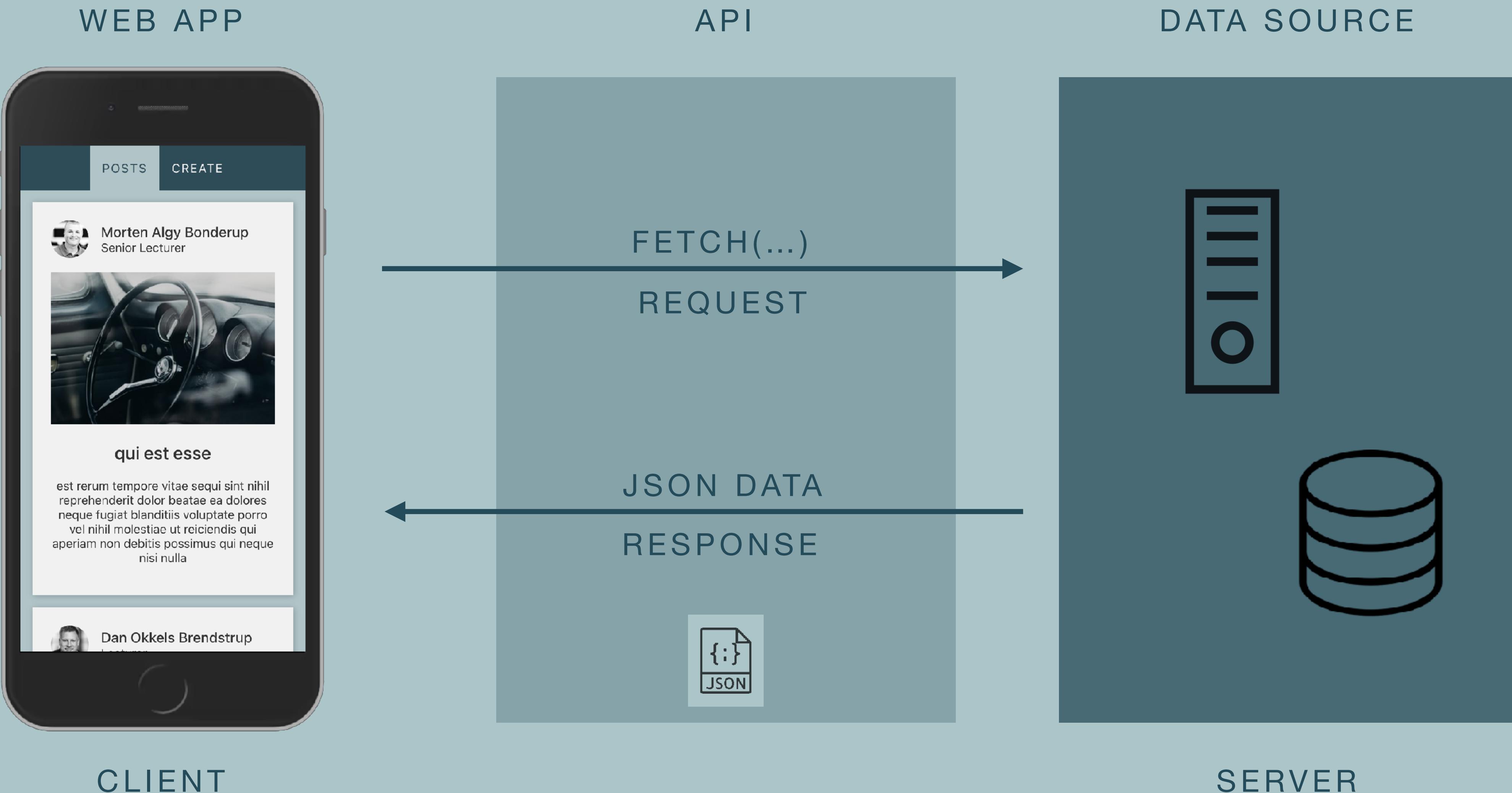
The screenshot shows a web browser window with the URL [kompetence.kea.dk/kurser-fag/webudvikling-frontend](https://kompetence.kea.dk/kurser-fag/webudvikling-frontend). The main content area displays information about the 'WEBUDVIKLING FRONTEND' course, including a 'SE HOLDSTART OG TILMELD DIG →' button and a 'LAV DIN EGEN PDF-BROCHURE' section with a 'FØJ TIL PDF' button. A red sidebar on the left contains a 'NY UDDANNELSE: DIPLOM I WEBUDVIKLING' section with text about the diploma program.

The Network tab in the developer tools shows a list of 46 requests. The requests include various files such as CSS, JS, and images, all with a status of 200. The initiator for most requests is 'webudvikling-fro...'. The 'Waterfall' column shows the duration of each request, with many requests taking less than 1 ms.

Name	Status	Type	Initiator	Size	Time	Waterfall
webudvikling-fro...	200	do...	Other	27....	90...	
quixtrap.css	200	styl...	webudvi...	(dis...	2 ms	
quix.css	200	styl...	webudvi...	(dis...	1 ms	
jquery.min.js?77...	200	script	webudvi...	(m...	0 ms	
style.css?77958...	200	styl...	webudvi...	(dis...	2 ms	
jquery-noconflic...	200	script	webudvi...	(m...	0 ms	
jquery-migrate.m...	200	script	webudvi...	(m...	0 ms	
front.css?77958...	200	styl...	webudvi...	(dis...	2 ms	
script.js?779581...	200	script	webudvi...	(m...	0 ms	
core.js?7795818...	200	script	webudvi...	(m...	0 ms	
keepalive.js?779...	200	script	webudvi...	(m...	0 ms	
content.css?779...	200	styl...	webudvi...	(dis...	2 ms	
bootstrap.min.js	200	script	webudvi...	(m...	0 ms	
font-awesome.m...	200	styl...	webudvi...	(dis...	2 ms	
kea-min.js?1.0.28	200	script	webudvi...	(m...	0 ms	
logo-main-black...	200	png	webudvi...	(m...	0 ms	
css2?family=Po...	200	styl...	webudvi...	(dis...	2 ms	
logo-small.png	200	png	webudvi...	(m...	0 ms	
template.css?1.0...	200	styl...	webudvi...	(dis...	2 ms	
finder.css?77958...	200	styl...	webudvi...	(dis...	2 ms	
.	200	.	.	.	.	

46 requests | 28.7 kB transferred | 1.9 MB resources | Finish: 1.59 s | DOMContentLoaded

# Web Development



- Brug Network-tabben til at undersøge et website (fx [kea.dk](http://kea.dk), [dr.dk](http://dr.dk), [google.dk](http://google.dk), [react-rest-and-auth.web.app](http://react-rest-and-auth.web.app) eller et helt andet).
- Åben websitet i Chrome (eller en anden browser).
- Åben Developer Tool (se: [How to open the dev tool in your browser](#)) og gå til Network/Netværk.
- Genindlæs siden imens du står i Network-tabben og undersøg hvilke ressourcer, der bliver hentet.
- Hvilke(n) type ressourcer er der tale om?
- Overvej hvordan ressourcerne hentes.



## Network-tabben

# Frontend stack

**HTML**



**JS**



**CSS**



STRUCTURE  
CONTENT

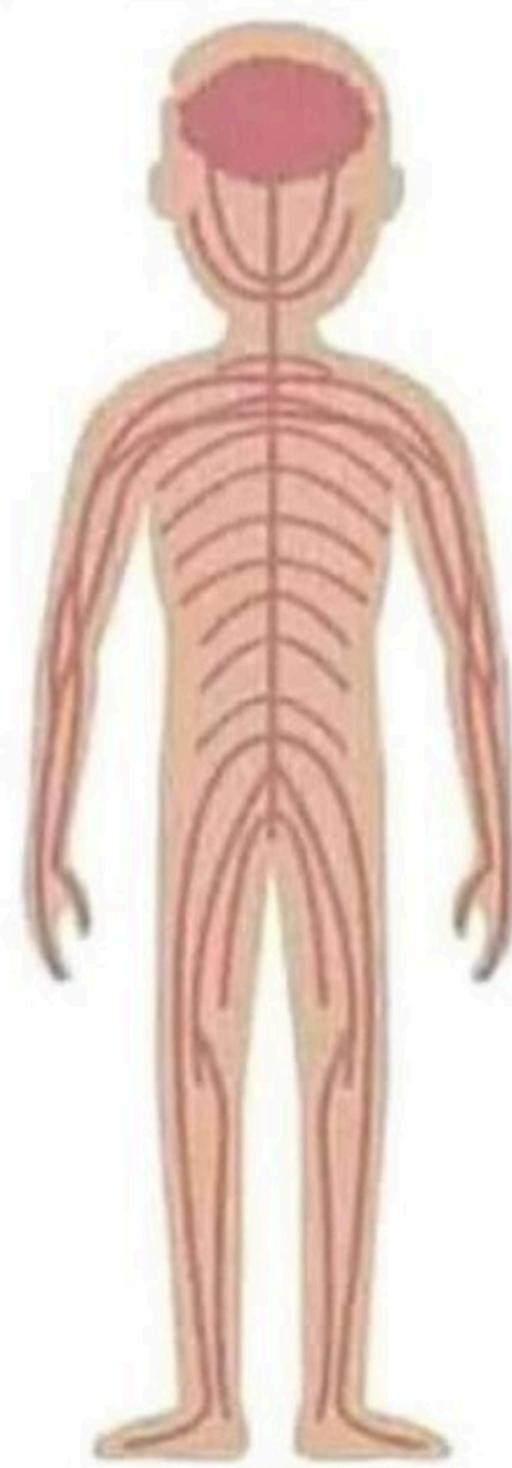
FUNCTIONALITY  
BEHAVIOR

LAYOUT/STYLING  
PRESENTATION

HTML



JS

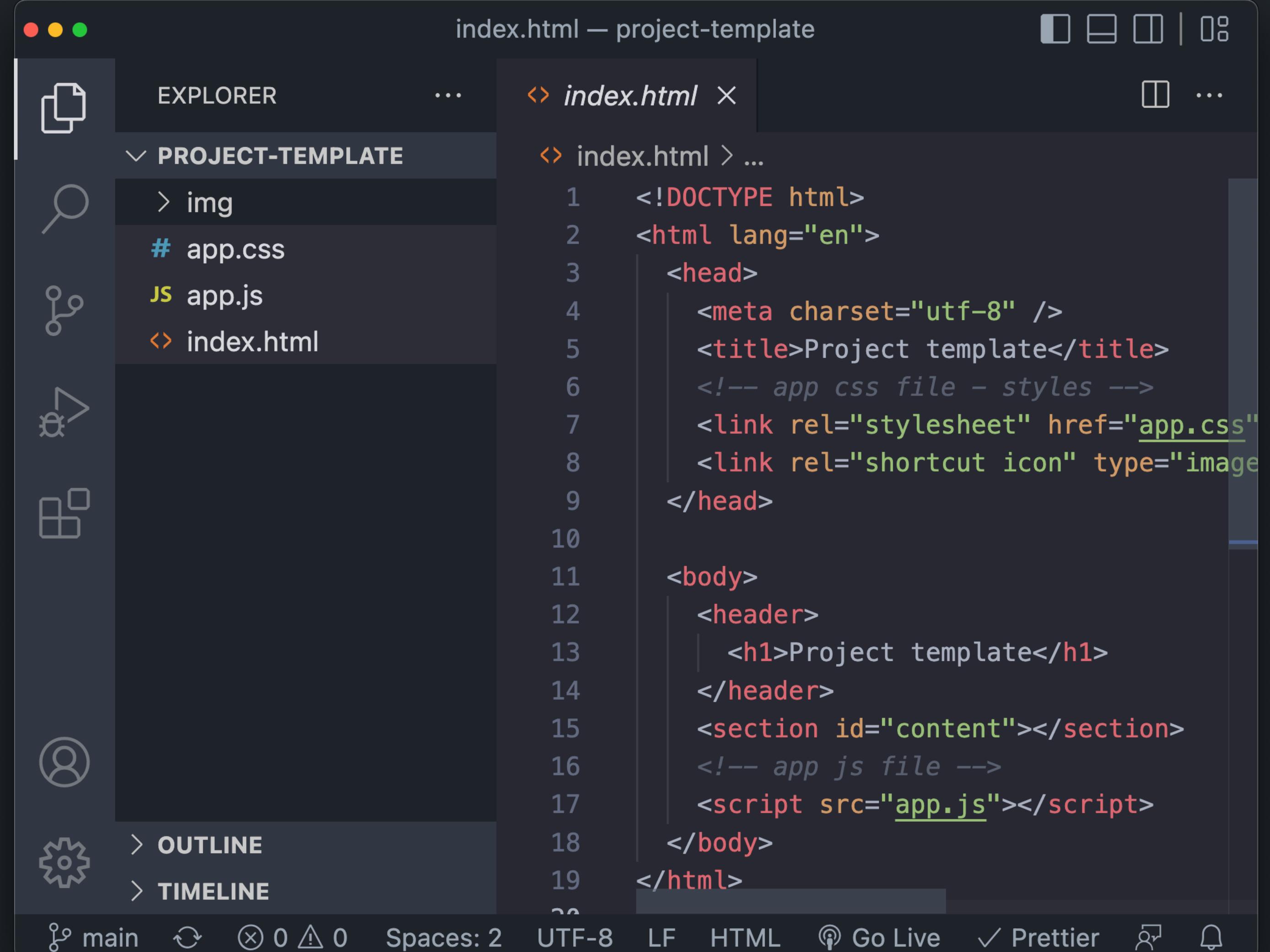


CSS



# Frontend Project Structure

- Project structure
  - HTML
  - CSS
  - JavaScript
- Keep a good structure & strive for consistency
- Separation of concerns



The screenshot shows a code editor interface with a dark theme. On the left, the Explorer sidebar displays a project structure under 'PROJECT-TEMPLATE' containing 'img', '# app.css', 'JS app.js', and 'index.html'. The 'index.html' file is selected and shown in the main editor area. The editor shows the following HTML code:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>Project template</title>
    <!-- app css file - styles -->
    <link rel="stylesheet" href="app.css" type="text/css" />
    <link rel="shortcut icon" type="image/x-icon" href="img/icon.png" />
</head>
<body>
    <header>
        <h1>Project template</h1>
    </header>
    <section id="content"></section>
    <!-- app js file -->
    <script src="app.js"></script>
</body>
</html>
```

The status bar at the bottom indicates the file is 'index.html – project-template', has 22 lines, and is saved with 'UTF-8' encoding.

# HTML, CSS & JavaScript

You can literally build anything with it!

# What is HTML?

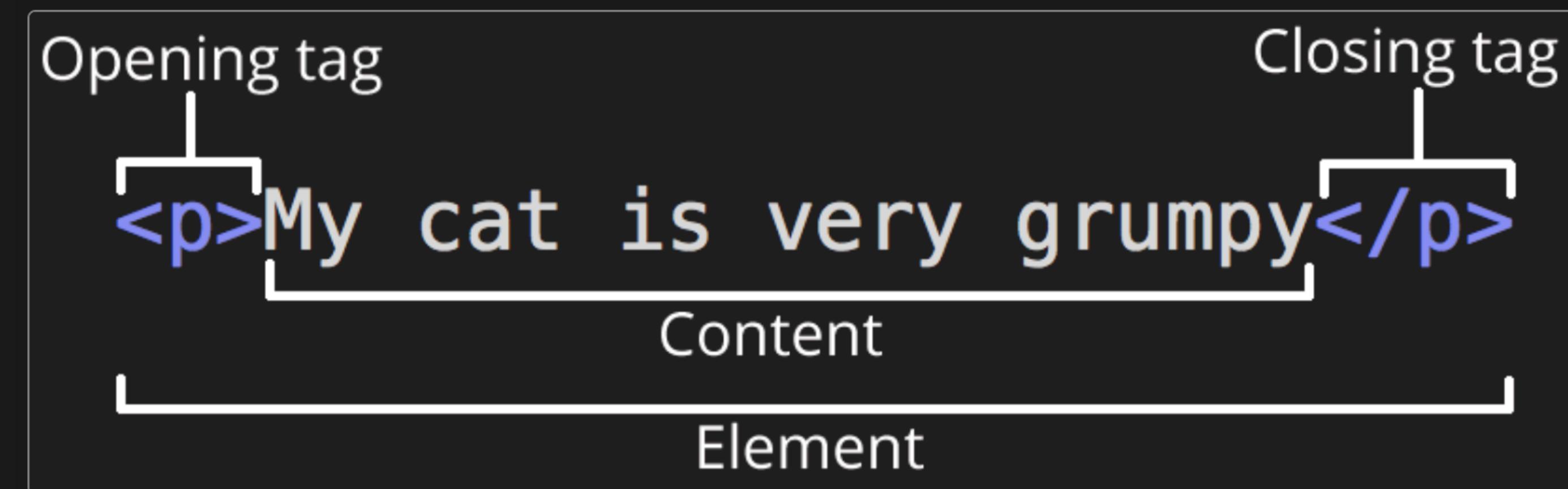
- Hyper Text Markup Language
- Standard markup language for creating Web pages
- Describes the structure of a Web page
- Consists of a series of elements (tags)
- HTML elements tell the browser how to display the content

```
index.html

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Page Title</title>
5   </head>
6   <body>
7     <h1>This is a Heading</h1>
8     <p>This is a paragraph.</p>
9   </body>
10 </html>
11
```

# Anatomy of an HTML element

Element: opening and closing tag + content



# Void elements

Element: single tag, usually to insert or embed something in a HTML

```

```



[https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction\\_to\\_HTML/Getting\\_started#void\\_elements](https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML/Getting_started#void_elements)

# Attributes

- Provides additional information and all element can have them.
- Are specified in the start tag
- name/value pairs: name="value"

[Attributes - mdn](#)

[HTML Attributes - W3Schools](#)

The screenshot shows a dark-themed web browser window. The title bar reads "Getting started with HTML - Le X". The address bar shows the URL "developer.mozilla.org/en-US/docs/Learn/HTML/Introduction\_to\_HTML/HTML结构性标记语言". The main content area has a header "Attributes" and a sub-section "Elements can also have attributes. Attributes look like this:" followed by a code example. A callout box highlights the word "Attribute" above the code. The code example is "

My cat is very grumpy

". Below the code, a text block explains that attributes contain extra information about the element and provides an example of the class attribute. At the bottom, there's a section titled "An attribute should have:" with a bulleted list of rules.

## Attributes

Elements can also have attributes. Attributes look like this:

Attribute

```
<p class="editor-note">My cat is very grumpy</p>
```

Attributes contain extra information about the element that won't appear in the content. In this example, the `class` attribute is an identifying name used to target the element with style information.

An attribute should have:

- A space between it and the element name. (For an element with more than one attribute, the attributes should be separated by spaces too.)
- The attribute name, followed by an equal sign.
- An attribute value, wrapped with opening and closing quote marks.

The screenshot shows a web browser window with the title "HTML Attributes". The URL in the address bar is "w3schools.com/html/html\_attributes.asp". The browser interface includes standard controls like back, forward, and search, along with a user profile icon.

The main content area has a dark background with white text. It features a navigation bar at the top with tabs for "HTML", "CSS", "JAVASCRIPT", "SQL", "PYTHON", "JAVA", and "PHP".

## The href Attribute

The `<a>` tag defines a hyperlink. The `href` attribute specifies the URL of the page the link goes to:

**Example**

```
<a href="https://www.w3schools.com">Visit W3Schools</a>
```

**Try it Yourself »**

You will learn more about links in our [HTML Links chapter](#).

## The src Attribute

The `<img>` tag is used to embed an image in an HTML page. The `src` attribute specifies the path to the image to be displayed:

**Example**

```

```

**Try it Yourself »**



```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Page Title</title>
5      </head>
6      <body>
7          <h1>This is a Heading</h1>
8          <p>This is a paragraph.</p>
9      </body>
10 </html>
11
```

# This is a Heading

This is a paragraph.

# Anatomy of an HTML document

```
<> index.html ×  
1  <!DOCTYPE html>  
2  <html>  
3    <head>  
4      <title>Page Title</title>  
5    </head>  
6    <body>  
7      <h1>This is a Heading</h1>  
8      <p>This is a paragraph.</p>  
9    </body>  
10   </html>  
11
```

The `<!DOCTYPE html>` declaration defines that this document is an HTML5 document

The `<html>` element is the root element of an HTML page

The `<head>` element contains meta information about the HTML page

The `<title>` element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)

The `<body>` element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.

The `<h1>` element defines a large heading

The `<p>` element defines a paragraph

```
index.html ×  
1  <!DOCTYPE html>  
2  <html>  
3  <head>  
4    <title>Page Title</title>  
5  </head>  
6  <body>  
7    <h1>This is a Heading</h1>  
8    <p>This is a paragraph.</p>  
9  </body>  
10 </html>  
11
```

```
<html>  
  
<head>  
  
  <title>Page title</title>  
  
</head>  
  
<body>  
  
  <h1>This is a heading</h1>  
  
  <p>This is a paragraph.</p>  
  
  <p>This is another paragraph.</p>  
  
</body>  
  
</html>
```

The screenshot shows a dark-themed web browser window. The title bar reads "Getting started with HTML - Le X". The address bar shows the URL "developer.mozilla.org/en-US/docs/Learn/HTML/Introduction\_to\_HTML/Getting\_started#anatomy\_of\_an\_html\_document". The main content area displays the MDN Web Docs logo and navigation bar with "Guides > Getting started with HTML". The main article title is "HTML comments". The text explains that HTML has a mechanism to write comments in the code that browsers ignore, used for explaining logic or coding. It provides an example of HTML code:

```
<p>I'm not inside a comment</p>

<!-- <p>I am!</p> -->
```

# HTML Quiz & Exercises

HTML Quiz

HTML Exercises

# HTML Semantic Elements

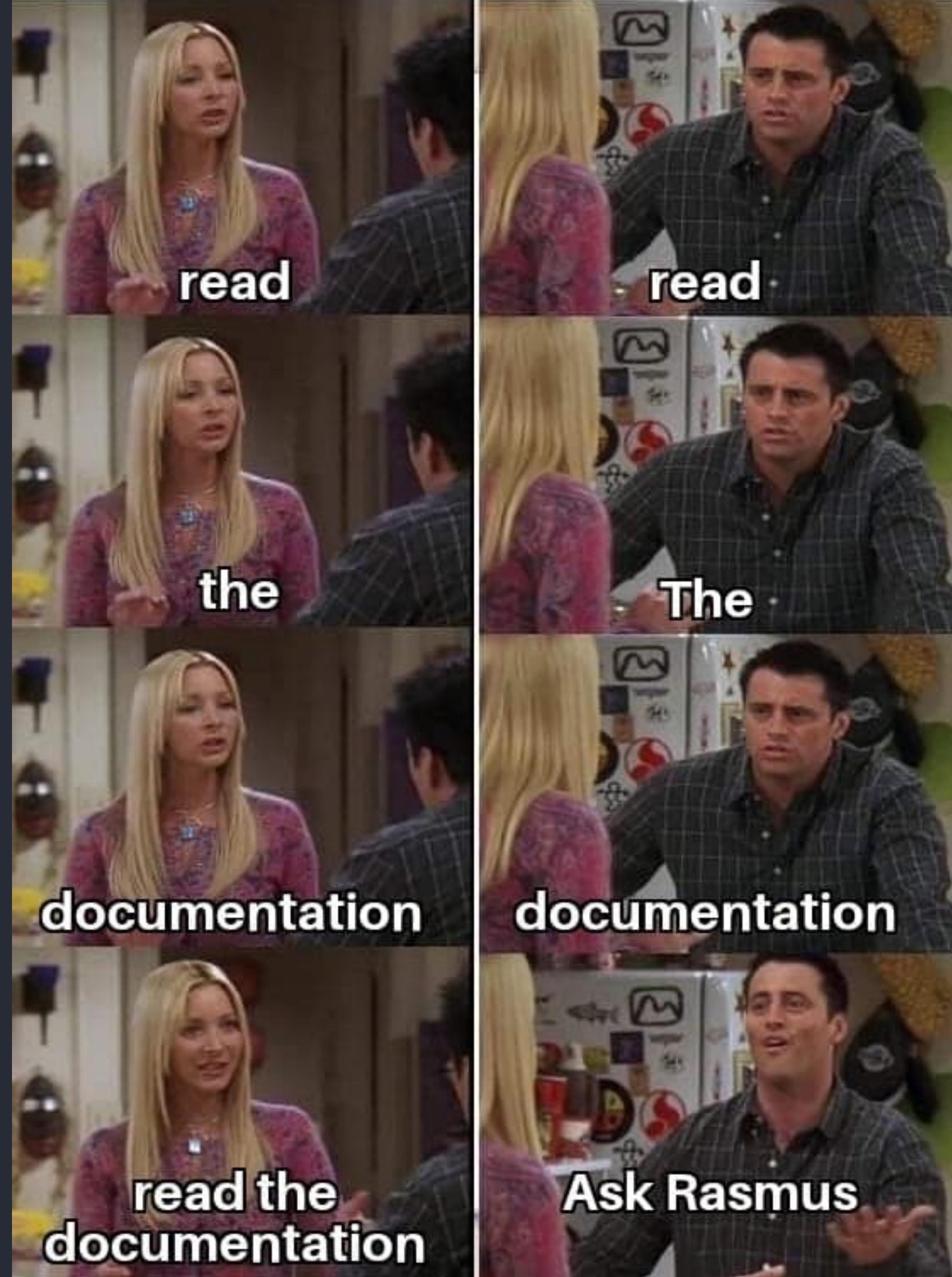
Clearly describes the meaning to both the browser and the developer.

Semantic elements = elements with a meaning

The screenshot shows a web browser window with the title "HTML Semantic Elements" and the URL "w3schools.com/html/html5\_semantic\_elements.asp". The page features a navigation bar with links for HTML, CSS, JAVASCRIPT, SQL, PYTHON, and JAVA. The main content is titled "Semantic Elements in HTML" and includes a note: "Below is a list of some of the semantic elements in HTML." A table lists 15 semantic elements with their descriptions:

Tag	Description
<code>&lt;article&gt;</code>	Defines independent, self-contained content
<code>&lt;aside&gt;</code>	Defines content aside from the page content
<code>&lt;details&gt;</code>	Defines additional details that the user can view or hide
<code>&lt;figcaption&gt;</code>	Defines a caption for a <code>&lt;figure&gt;</code> element
<code>&lt;figure&gt;</code>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<code>&lt;footer&gt;</code>	Defines a footer for a document or section
<code>&lt;header&gt;</code>	Specifies a header for a document or section
<code>&lt;main&gt;</code>	Specifies the main content of a document
<code>&lt;mark&gt;</code>	Defines marked/highlighted text
<code>&lt;nav&gt;</code>	Defines navigation links
<code>&lt;section&gt;</code>	Defines a section in a document
<code>&lt;summary&gt;</code>	Defines a visible heading for a <code>&lt;details&gt;</code> element
<code>&lt;time&gt;</code>	Defines a date/time

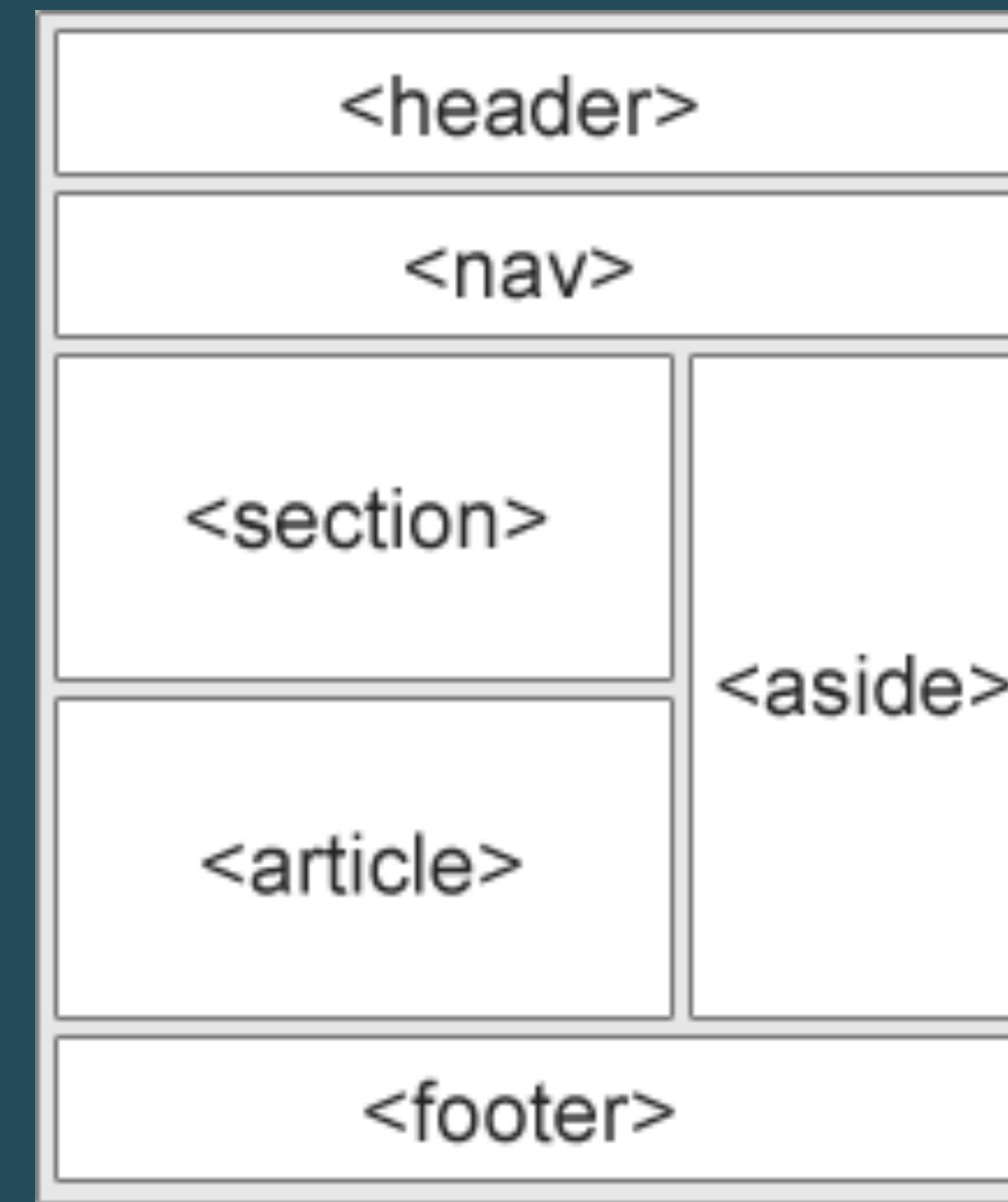
At the bottom, a yellow bar contains the text: "For a complete list of all available HTML tags, visit our [HTML Tag Reference](#)".



# 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>



# 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>
```

## NON SEMANTIC

```
<div class='header'>  
  <div class='section'>  
    <div class='article'>  
      <div class='article'>  
    </div>  
  </div>  
<div class='aside'>  
</div>  
<div class='footer'>
```

## SEMANTIC

```
<header>  
<section>  
<article>  
<article>  
<aside>  
<footer>
```



# <div> tags?

Many websites 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?

Apple x + apple.com

Store Mac iPad iPhone Watch AirPods TV & Home Only on Apple Accessories Support Q □

# iPhone 14 Pro

Pro. Beyond.

[Learn more >](#) [Buy >](#)



# iPhone 14

Big and bigger.

iPhone 14 Plus available starting 10.7

[Learn more >](#) [Shop >](#)

Elements Console » F 1 ⚙️ ⋮ ×

```
><nav id="ac-globalnav" class="js no-touch no-windows no-firefox" 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>
<div id="ac-gn-curtain" class="ac-gn-curtain"></div>
<div id="ac-gn-placeholder" class="ac-gn-placeholder">
</div>
<script type="text/javascript" src="/ac/globalnav/7/en_US/scripts/ac-globalnav.built.js"></script>
...
  ><div id="ac-gn-viewport-emitter" data-viewport-emitter-dispatch data-viewport-emitter-state="{"viewport":"medium","orientation":"portrait","retina":true}">...</div> == $0
<script src="/metrics/ac-analytics/2.15.1/scripts/ac-analytics.js" type="text/javascript" charset="utf-8"></script>
  ><main class="main" role="main">...</main>
  ><footer class="js" lang="en-US" id="ac-globalfooter" data-analytics-region="global footer" role="contentinfo" aria-...
...
  -image.no-reduced-motion.no-edge.no-ie.css-mask.inline-video.deskt ...

```

Styles Computed Layout Event Listeners DOM Breakpoints »

Filter :hov .cls + ↻

```
element.style {
}
#ac-gn-viewport-emitter { ac-globalna...built.css:1
  overflow: hidden;
  position: absolute;
  top: 0;
  left: 0;
  width: 0;
  height: 0;
  visibility: hidden;
  z-index: -1;
}
```

Google

google.com

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 ↗ 1 ↘ 1 ⚙ ⋮ X

```
<!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;02vyse:.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>
    ... <div class="L3eUgb" data-hveid="1"> flex == $0
      <div class="o3j99 n1xJcf Ne6nSd">...</div> flex
      <div class="o3j99 LLD4me yr19Zb LS80J">...</div> flex
      <div class="o3j99 ikrT4e om7nvf">...</div>
      <div class="o3j99 qarstb">...</div>
      <div class="o3j99 c93Gbe">...</div>
    </div>
    <div class="Fgvgjc">
      <style data-iml="1634195875111">
        .Fgvgjc{height:0;overflow:hidden}</style>
      <div class="gTMtLb fp-nh" id="lb">...</div>
      <div jscontroller="fKZehd" style="display:none" data-u="0" jsdata="C4mkuf;_;AzQv+I" jsaction="rcuQ6b:npT2md">
      <span style="display:none">...</span>
      <script nonce="4eyH3+1nxzKy0S3HAsRNUA==">...</script>
      <div>...</div>
    </div>
  html body div.L3eUgb
  Styles Computed Layout Event Listeners DOM Breakpoints >
```

Filter :hov .cls + [ ]

```
element.style {
}
.L3eUgb {
  display: flex;
  flex-direction: column;
}
```

The screenshot shows a web browser window with the title "HTML Semantic Elements" from the website w3schools.com. The page content is as follows:

# HTML Semantic Elements

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

On the right side of the browser, a developer tools sidebar is open, showing the DOM structure and CSS styles applied to the page. The DOM tree starts with `<body>` and includes nodes for the main content area, a sidebar, and various semantic elements like `<header>`, `<nav>`, and `<article>`. The CSS styles panel shows rules for `html` and `body` elements, including font-family: Verdana, sans-serif; and font-size: 15px;.

# <div> tags?

Use semantic elements as much as possible (!)

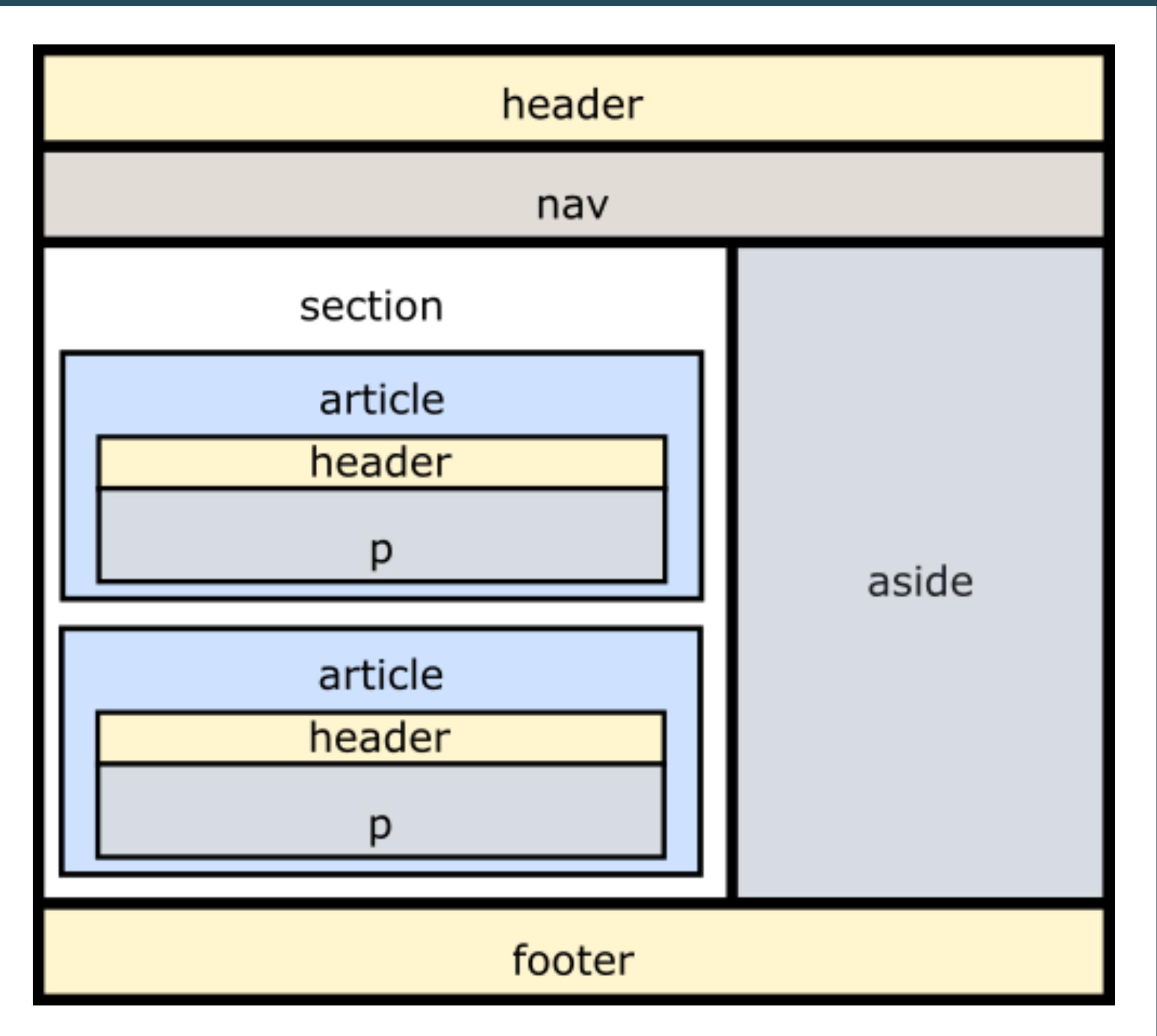
... but don't let it stop you from building

something amazing 

- RACE

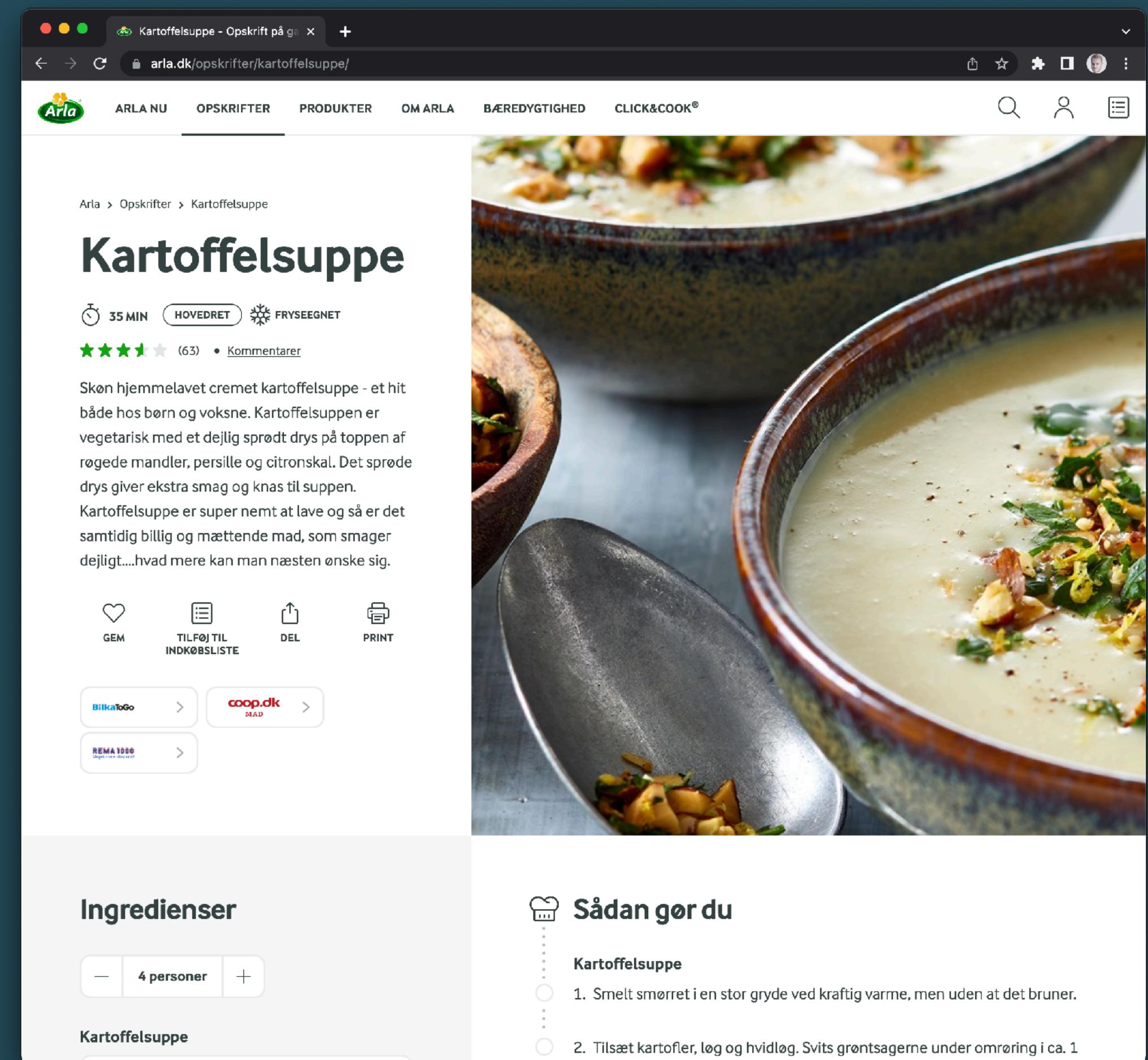
# HTML Semantics Exercise

1. Use the project-template from [GitHub](#).  
Make a copy of the project and paste the project folder into your own local dev folder.
2. Create a structure matching the image to the right. Use HTML Semantic tags.
3. Add content to the structure, like headings, links, text (paragraphs), images and contact info. Explore and use [HTML Element Reference](#) & [HTML Semantics](#).
4. Extra: Add some styling (CSS).



# HTML Recipe Exercise

- Tag udgangspunkt i dit projekt fra foregående opgave (tag eventuelt en kopi)
- Strukturér en opskriftsside med elementer som overskrift(er), beskrivelse(r), billede(r), liste af ingredienser og fremgangsmåde. Du skal bruge HTML elementer og semantiske tags.
- Fokus er struktur i HTML'en - ikke styling og layout.
- Du kan bruge elementer som h1, h2, h3, p, section, article, aside, main, ul, li og mange andre. Gå på opdagelse i "dokumentationen".
- Brug indhold fra websitet til venstre eller en anden opskrift efter eget ønske.



# Semantic HTML & SEO(indexability)

Why tell the browser what the HTML elements represents?

... to improve

# Search Engine Optimization

Tell what your content is about and  
by hierarchy, tell what's important

# Why Semantic HTML?

“Because semantic HTML uses elements for their given purpose, it’s easier for both people and machines to read and understand it.”

“Semantic HTML means using correct HTML elements for their correct purpose as much as possible. Semantic elements are elements with a meaning; if you need a button, use the `<button>` element (and not a `<div>` element).”

### Semantic

```
<button>Report an Error</button>
```

### Non-semantic

```
| <div>Report an Error</div>
```

**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.

1. Headings <h1> to <h6>
2. Document Structure
3. Textual Meaning (Bold, Italics, Highlight)
4. Media Type
5. Correlation Tags

5 Ways to Write Semantic HTML and Improve Webpage SEO and Accessibility

[Read more here](#)

# What's Accessibility?

“[...] the practice of making your websites usable by as many people as possible.”

- and machines, programs and devices.

What is accessibility?

“Web accessibility means that people with disabilities can use the Web.”

Why Web Accessibility Is Important and How You Can Accomplish It

- Semantic HTML
- Headings are Important!
- Alternative Text
- HTML Language Type (en, da, etc.)
- Clear written language
- Show the user this is a link or a clickable element
- ARIA - a set of roles and attributes to make the content more accessible

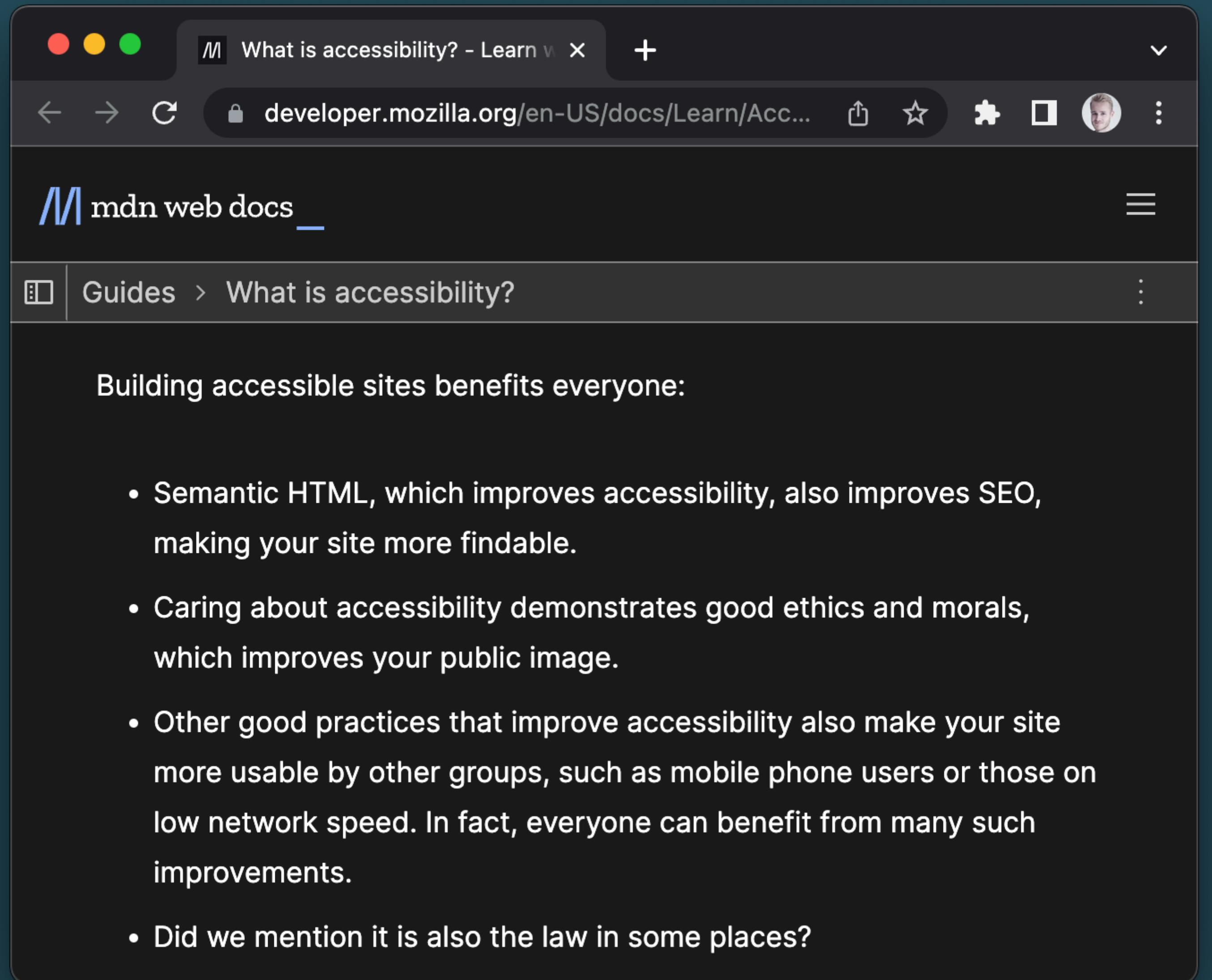
[HTML Accessibility - W3Schools](#)

[HTML: A good basis for accessibility](#)

[Accessibility and HTML](#)

[ARIA](#)

# The benefits of accessibility?



A screenshot of a web browser window displaying the MDN Web Docs page titled "What is accessibility?". The browser has a dark mode interface. The page content starts with a heading "Building accessible sites benefits everyone:" followed by a bulleted list of six items. The list discusses the benefits of accessibility, including improved SEO, good ethics, improved public image, and increased usability for various users.

What is accessibility?

MDN Web Docs

Guides > What is accessibility?

Building accessible sites benefits everyone:

- Semantic HTML, which improves accessibility, also improves SEO, making your site more findable.
- Caring about accessibility demonstrates good ethics and morals, which improves your public image.
- Other good practices that improve accessibility also make your site more usable by other groups, such as mobile phone users or those on low network speed. In fact, everyone can benefit from many such improvements.
- Did we mention it is also the law in some places?

What is accessibility?

# Lighthouse in Chrome

The image shows a screenshot of a web browser displaying the KEA - Københavns Erhvervsakademি website. The main heading on the page is "VIL DU STARTE PÅ EN UDDANNELSE TIL JANUAR?". Below it is a large button labeled "→ SØG IND NU". On the left side of the page, there are three buttons: "FÅ HJÆLP TIL AT VÆLGE UDDANNELSE", "SÅDAN ANSØGER DU", and "FIND SVAR I VORES FAQ". On the right side, there is a section titled "FÅ 10.000 TIL DIN EFTERUDDANNELSE" with a button "→ LÆS HVORDAN PÅ KEA.DK/EFTERUDDANNELSER" and a small image of a woman sitting cross-legged.

To the right of the browser window is the Lighthouse performance audit interface. The top navigation bar includes tabs for "Elements", "Console", "Sources", "Network", "Lighthouse", and "PWA". The audit score is shown as 79. Below the score, five circular icons represent different categories: Performance (79), Accessibility (89), Best Practices (67), SEO (82), and PWA (PWA). The "Performance" section is expanded, showing detailed metrics:

Metric	Value
First Contentful Paint	0.7 s
Time to Interactive	1.8 s
Total Blocking Time	0 ms
Cumulative Layout Shift	0.055

At the bottom of the Lighthouse interface, there are buttons for "View Original Trace" and "View Treemap", along with a series of thumbnail previews of the website's layout across different screen sizes.

# Git, GitHub & GitHub Desktop

Version Control, Collaboration & how to manage  
your code projects

# Git

VERSION CONTROL - SOFTWARE

A system that tracks all changes in your code

# GitHub

CLOUD-BASED HOSTING SERVICE

- The interface that uses git and helps people
- like you and me collaborate on projects

# GitHub Desktop

AN APPLICATION ON YOUR COMPUTER

A GUI to manage your project with Git  
and GitHub

# Git



Git is installed and maintained on your local system (rather than in the cloud)



First developed in 2005

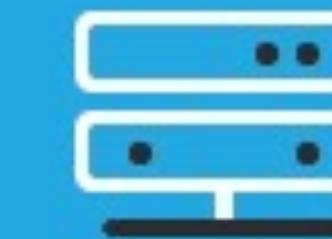


One thing that really sets Git apart is its branching model

Git is a high quality version control system

vs.

# GitHub



GitHub is designed as a Git repository hosting service

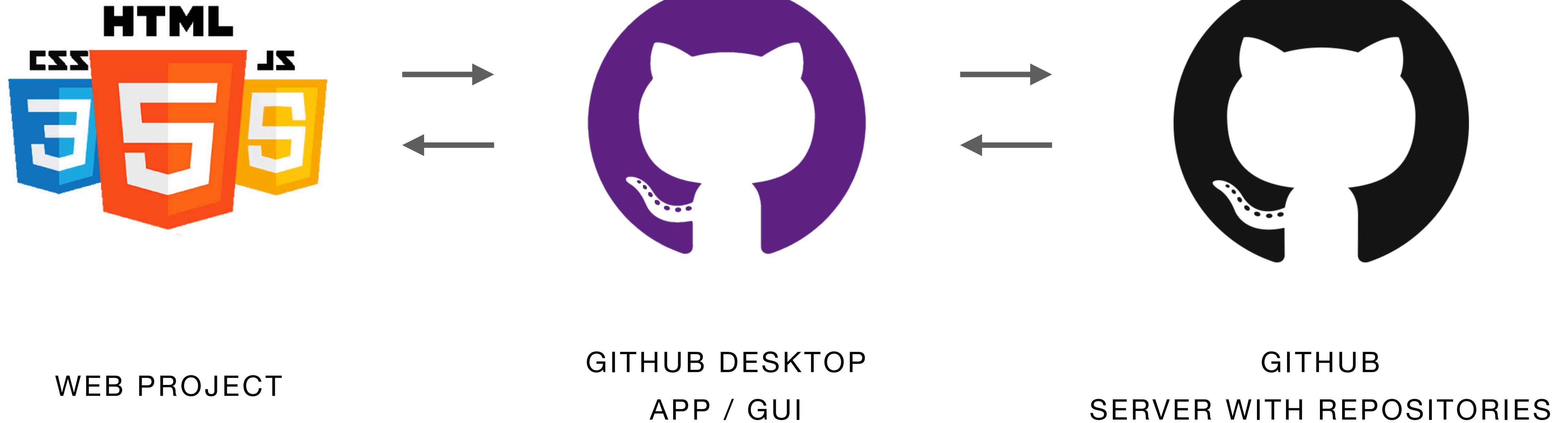


You can share your code with others, giving them the power to make revisions or edits



GitHub is a cloud-based hosting service

# Git



# Performance is about...

- Load time of your website
- Is your site usable while loading resources? (Lazy loading)
- Smoothness and interactivity
- Does your site seem fast to the user? (Perceived performance)
- Measuring performance

Reduce and minimise size (ex images),  
loading and response time

# Lighthouse in Chrome

The image shows a screenshot of the KEA - Københavns Erhvervsakademি website on the left and the Lighthouse performance audit report on the right, displayed within the Chrome DevTools.

**KEA Website Screenshot:**

- Header:** kea
- Main Call-to-Action:** VIL DU STARTE PÅ EN UDDANNELSE TIL JANUAR?
- Search Bar:** → SØG IND NU
- Text:** KEA - KØBENHAVNS ERHVERVSAKADEMI UDBYDER VIDEREGLÅNDE UDDANNELSER, DER KOMBINERER TEORI OG PRAKSIS.
- Buttons:**
  - FÅ HJÆLP TIL AT VÆLGE UDDANNELSE →
  - SÅDAN ANSØGER DU →
  - FIND SVAR I VORES FAQ →
- Image:** A woman sitting cross-legged, smiling.

**Lighthouse Report Metrics:**

Metric	Score
Performance	79
Accessibility	89
Best Practices	67
SEO	82
PWA	-

**Performance Details:**

- Values are estimated and may vary. The [performance score is calculated](#) directly from these metrics. See [calculator](#).
- ▲ 0-49   ■ 50-89   ● 90-100
- METRICS**
  - First Contentful Paint: 0.7 s
  - Speed Index: 1.8 s
  - Largest Contentful Paint: 3.0 s
  - Time to Interactive: 1.8 s
  - Total Blocking Time: 0 ms
  - Cumulative Layout Shift: 0.055

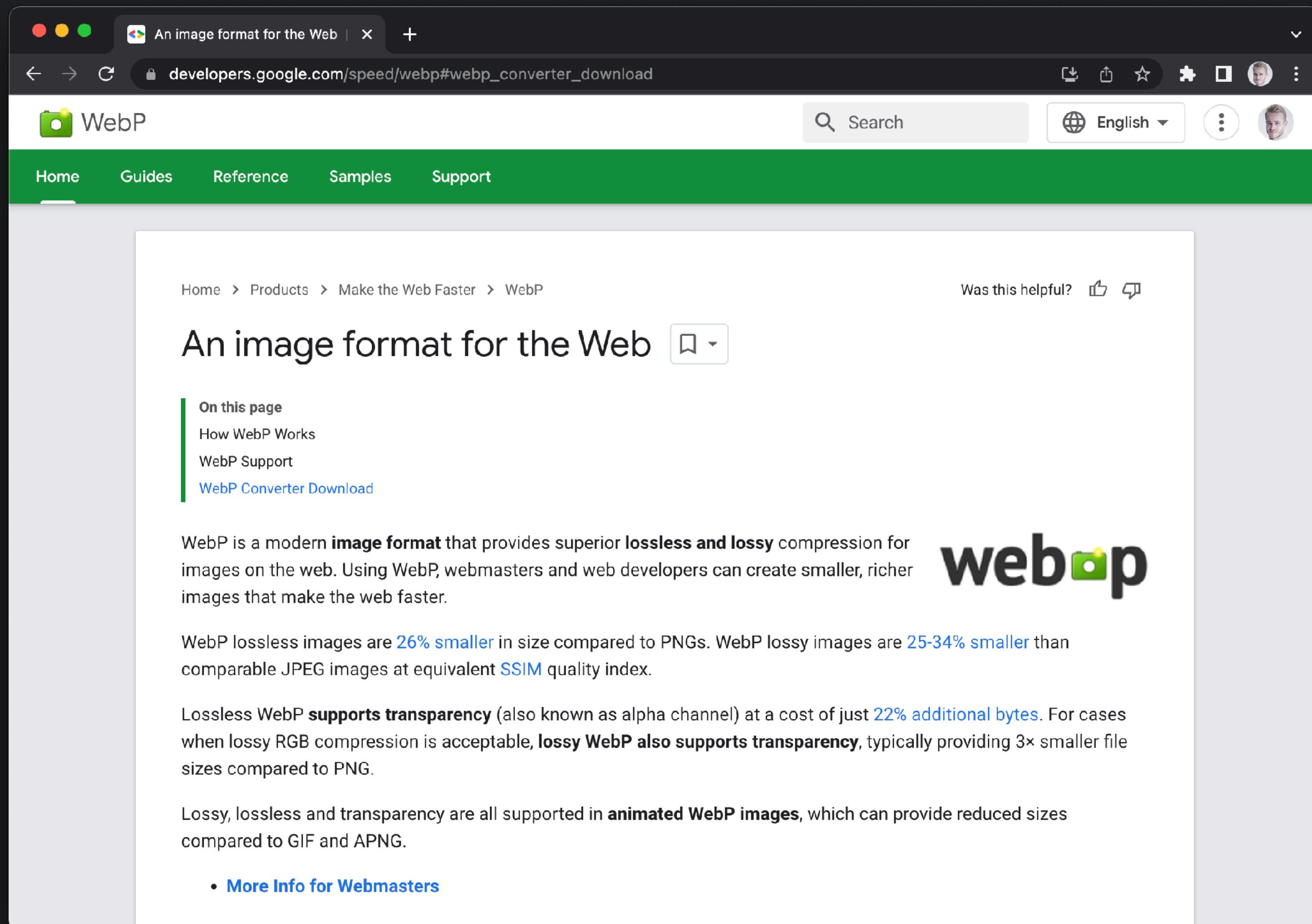
[View Original Trace](#) | [View Treemap](#)

# Not bragging



The image shows a split-screen view of a web browser. On the left is the homepage of [cederdorff.com](https://cederdorff.com), featuring a large portrait of a man wearing a black fedora and a dark turtleneck. Below the photo, the text "I'M A |" is followed by "WEB APP DEVELOPER | SENIOR LECTURER" and social media icons for Instagram, LinkedIn, and Facebook. On the right is the Lighthouse performance audit results page for the same URL. The audit summary shows scores of 95 for Performance, 94 for Accessibility, 100 for Best Practices, 100 for SEO, and a PWA badge. The "Performance" section details metrics: First Contentful Paint (0.6s), Speed Index (0.8s), Largest Contentful Paint (1.5s), Time to Interactive (0.6s), Total Blocking Time (0ms), and Cumulative Layout Shift (0.003). A small thumbnail of the homepage is shown next to the performance summary.

# Images, sizes & formats



The screenshot shows a web browser window displaying the "An image format for the Web" page from the Google Developers website. The URL in the address bar is [https://developers.google.com/speed/webp#webp\\_converter\\_download](https://developers.google.com/speed/webp#webp_converter_download). The page has a green header with the "WebP" logo and navigation links for Home, Guides, Reference, Samples, and Support. The main content area includes a breadcrumb trail (Home > Products > Make the Web Faster > WebP), a "Was this helpful?" button, and a "webp" logo. The text on the page explains that WebP is a modern image format providing superior lossless and lossy compression, with lossless images being 26% smaller than PNGs and lossy images being 25-34% smaller than comparable JPEGs. It also mentions transparency support and animated WebP images.

An image format for the Web

On this page

- How WebP Works
- WebP Support
- WebP Converter Download

WebP is a modern **image format** that provides superior **lossless and lossy** compression for images on the web. Using WebP, webmasters and web developers can create smaller, richer images that make the web faster.

WebP lossless images are **26% smaller** in size compared to PNGs. WebP lossy images are **25-34% smaller** than comparable JPEG images at equivalent **SSIM** quality index.

Lossless WebP **supports transparency** (also known as alpha channel) at a cost of just **22% additional bytes**. For cases when lossy RGB compression is acceptable, **lossy WebP also supports transparency**, typically providing 3x smaller file sizes compared to PNG.

Lossy, lossless and transparency are all supported in **animated WebP images**, which can provide reduced sizes compared to GIF and APNG.

- [More Info for Webmasters](#)

<https://developers.google.com/speed/webp>

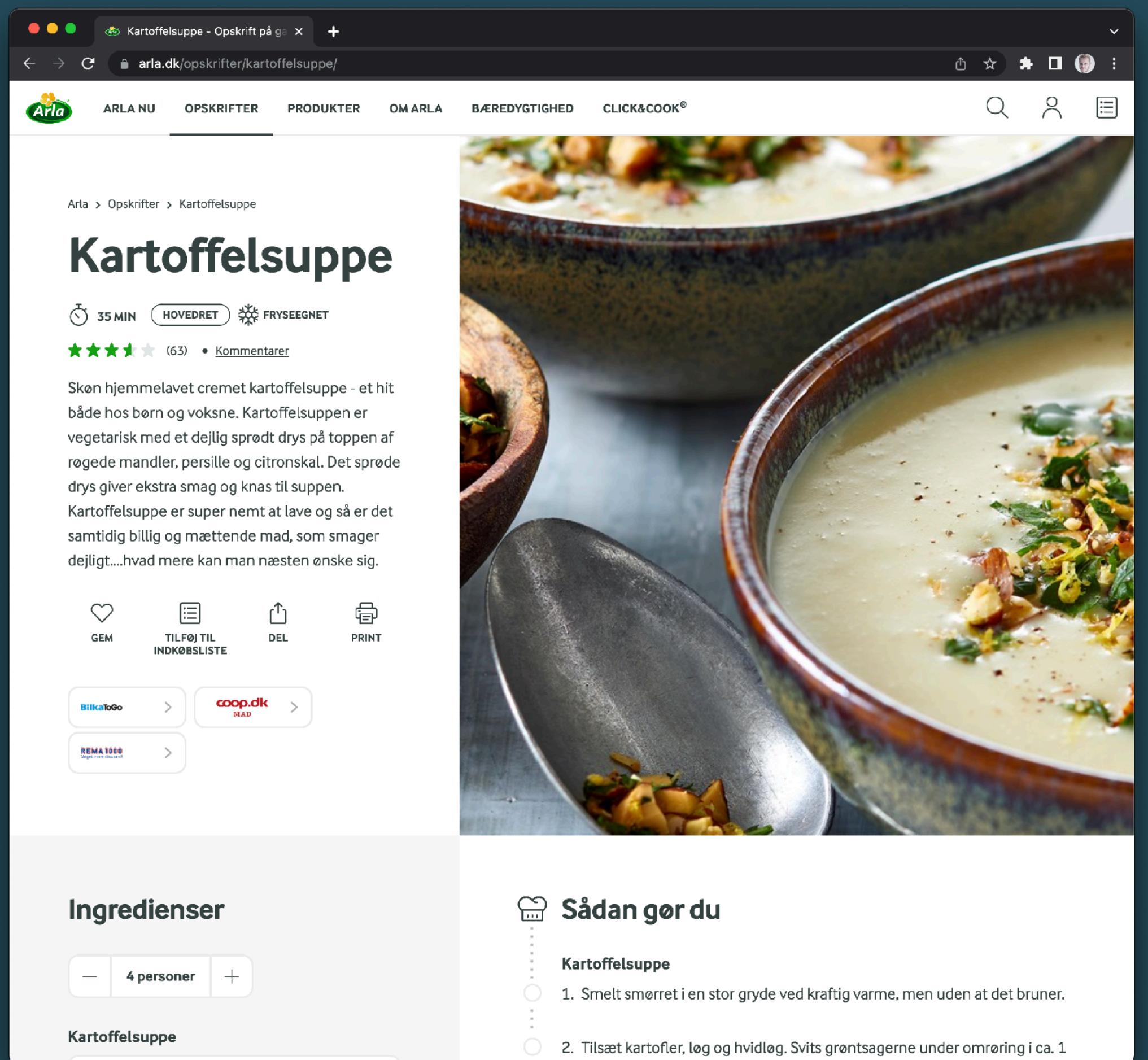
# .webp

The screenshot shows a web browser window with the title "Rasmus Cederdorff | Web App" and the URL "cederdorff.com". The main content area displays a section titled "CLIENTS" with a sub-section for "HOUSE OF VINCENT" featuring a "WEBSITE + WEBSHOP". Below this, there is a paragraph about the website's implementation and a "GO TO WEBSITE AND WEBSHOP >" button. To the right of the main content, the browser's developer tools are open, specifically the Network tab. The Network tab shows a timeline at the top with time markers at 20 ms, 40 ms, 60 ms, 80 ms, and 100 ms. Below the timeline is a table listing network requests. The table has columns for Name, Status, Type, Initiator, Size, Time, and Waterfall. The requests listed are all 200 status code responses, primarily of type "webp", initiated by "react-dom...." or "main.b9823...". The "Waterfall" column contains small blue vertical bars. At the bottom of the Network tab, there is a summary: "11 / 23 requests | 0 B / 529 kB transferred | 720 kB / 1.8 MB resources | Finish: 86 ms | DOMContentLoaded".

Name	Status	Type	Initiator	Size	Time	Waterfall
banner_web.webp	200	webp	react-dom....	(mem...)	0 ms	
rasmus_new.webp	200	webp	react-dom....	(mem...)	0 ms	
logo_inverted.webp	200	webp	react-dom....	(mem...)	0 ms	
header_mobile.6eb9647...	200	webp	main.b9823...	(mem...)	0 ms	
expertise_bg.9998200f8...	200	svg+...	main.b9823...	(mem...)	0 ms	
houseofvincent.webp	200	webp	react-dom....	(mem...)	0 ms	
boutime_web.webp	200	webp	react-dom....	(mem...)	0 ms	
sidewalk.webp	200	webp	react-dom....	(mem...)	0 ms	
thebigfridge_web.webp	200	webp	react-dom....	(mem...)	0 ms	
cphcloud_web.webp	200	webp	react-dom....	(mem...)	0 ms	
karolineshus_web.webp	200	webp	react-dom....	(mem...)	0 ms	

# Webp Exercise

1. Find og download et par billeder du kan bruge på din opskriftsside.
2. Sørg for at billeder ikke er større end nødvendigt (opløsning, height og width).
3. Test med Lighthouse. Tag et screenshot af resultatet.
4. Konverter dine billeder til .wepb (brug fx en online converter).
5. Bruge webp-filerne i stedet for dine tidligere billedfiler.
6. Test med Lighthouse igen og sammenlign resultatet med tidligere



# Hvordan er “formen”?

Tak for i dag

Webudvikling

**kea**  
KØBENHAVNS ERHVERVSAKADEMI