Mapping og Databaseklient

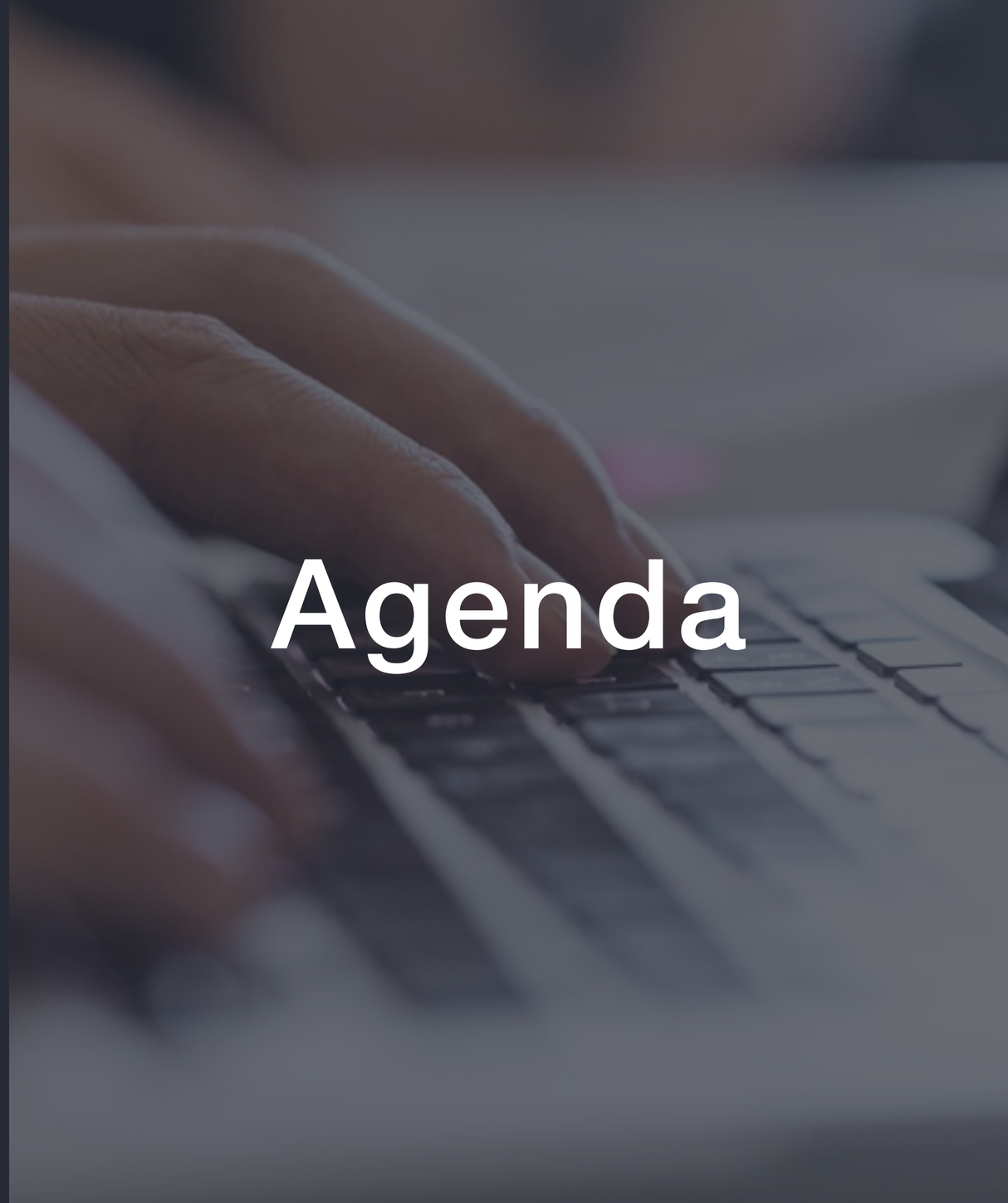
med

Formål

- Forståelse for hvordan man kan implementere en databaseklient med Node.js, Express.js og MySQL2
- Introduktion til Object Relational Mapping (ORM) - hvordan vi kan mappe objekter til database og omvendt
- At kunne anvende Sequelize som ORM for MySQL

Agenda

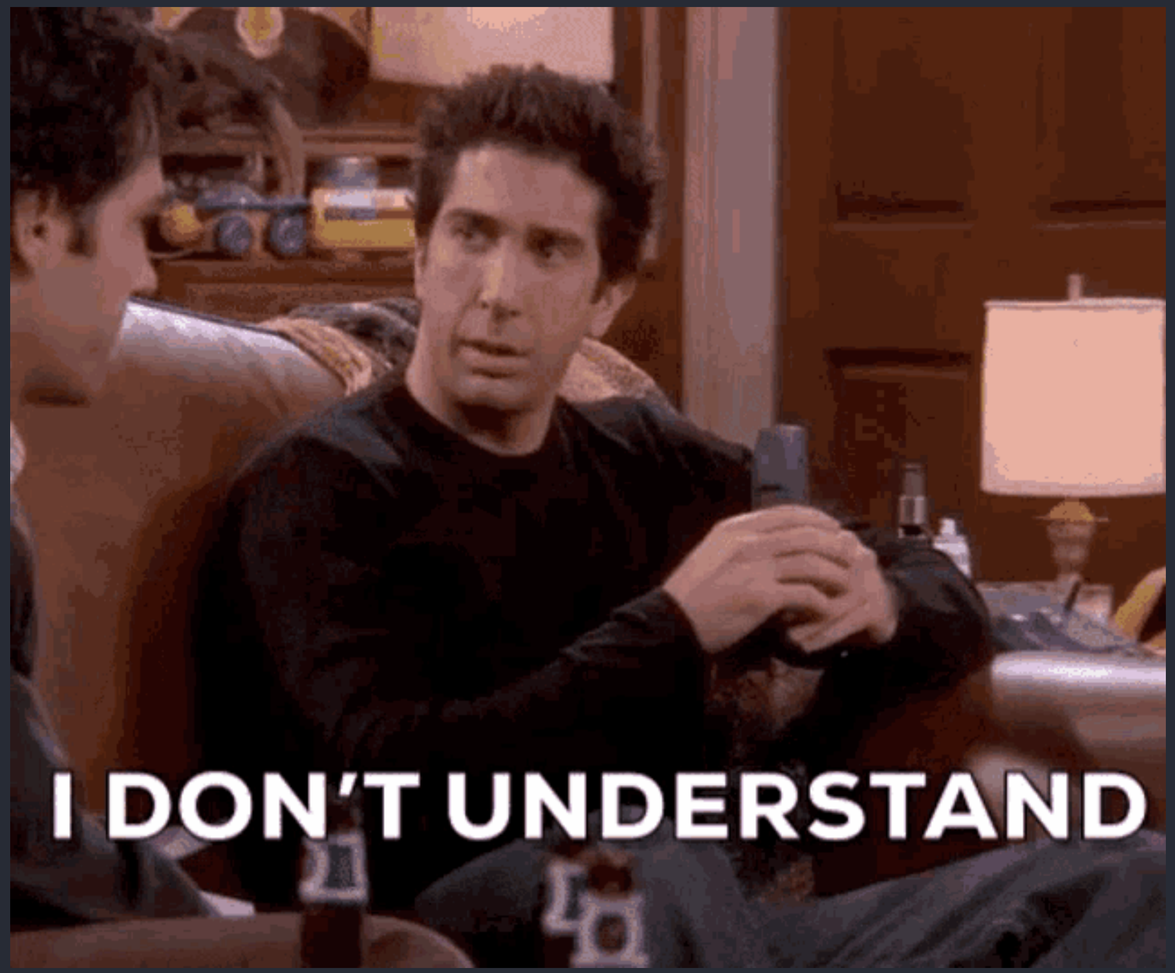
1. Intro til dagens undervisning
    - Client-server
    - Objekter og arrays
    - CRUD og REST API
2. Node.js og MySQL
3. Object Relational Mapping (ORM) med Sequlize

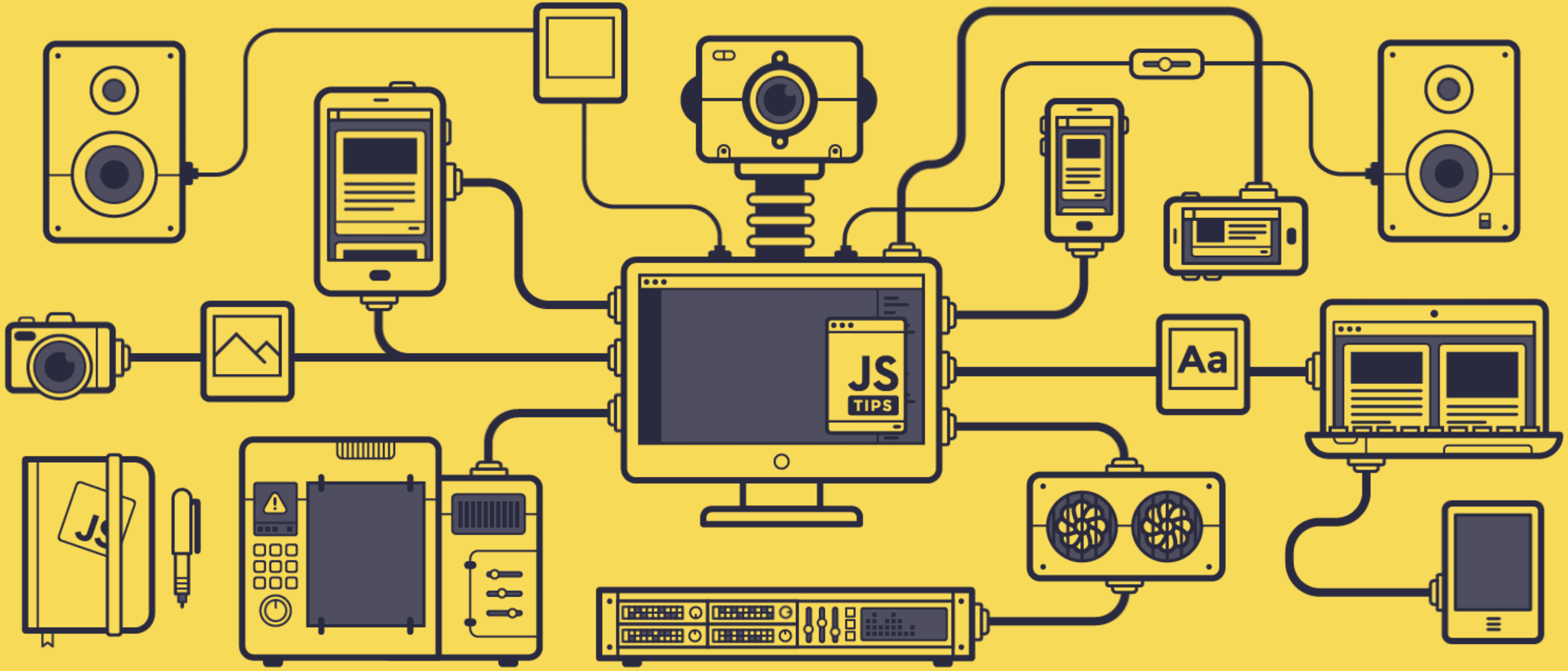


Disclaimer

Please,

Do not try to understand everything!

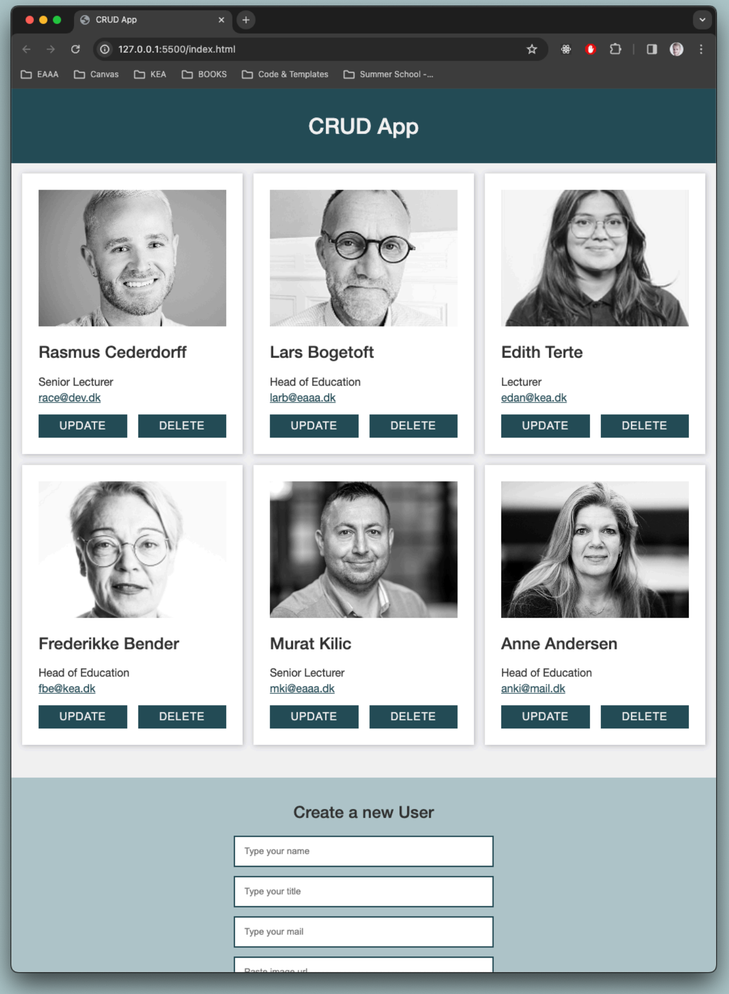



JS
TIPS
Aa

Web frameworks and technologies

https://survey.stackoverflow.co/2023/#most-popular-technologies-webframe

Web App



Database

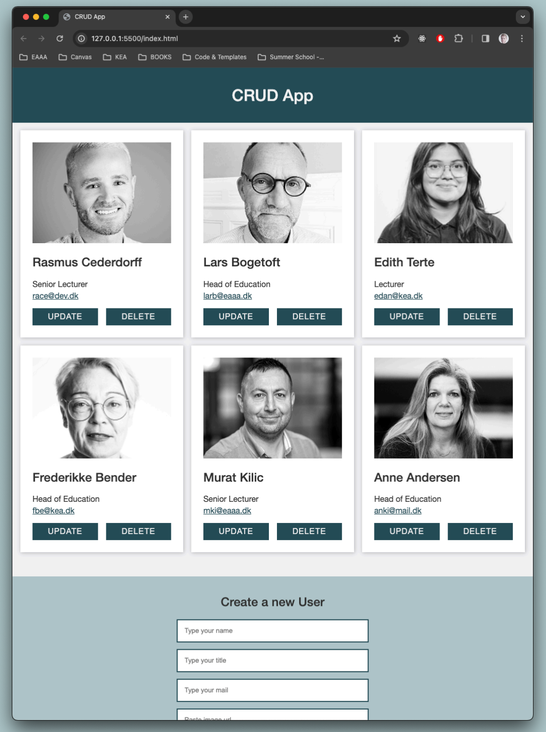
id	name	mail	title	image
1	Peter Lind	petl@kea.dk	Senior Lecturer	https://
2	Rasmus Cederdorff	race@dev.dk	Senior Lecturer	https://
3	Lars Bogetoft	larb@eaaa.dk	Head of Education	https://
4	Edith Terte	edan@kea.dk	Lecturer	https://
5	Frederikke Bender	fbe@kea.dk	Head of Education	https://
6	Murat Kilic	mki@eaaa.dk	Senior Lecturer	https://
7	Anne Kirketerp	anki@eaaa.dk	Head of Education	https://

Hvordan “mapper” vi det her?

Frontend

REST API med Node.js

Database



SQL

id	name	mail	title
1	Peter Lind	petl@kea.dk	Senior Lectu
2	Rasmus Cederdorff	race@dev.dk	Senior Lectu
3	Lars Bogetoft	larb@eaaa.dk	Head of Educ
4	Edith Terte	edan@kea.dk	Lecturer
5	Frederikke Bender	fbe@kea.dk	Head of Educ
6	Murat Kilic	mki@eaaa.dk	Senior Lectu
7	Anne Kirketerp	anki@eaaa.dk	Head of Educ

Web App
(Client)

REST API
(Udveksler JSON)

Backend App
(Server)

MySQL Database
(Database Server)

Client-Server Model

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

CLIENT

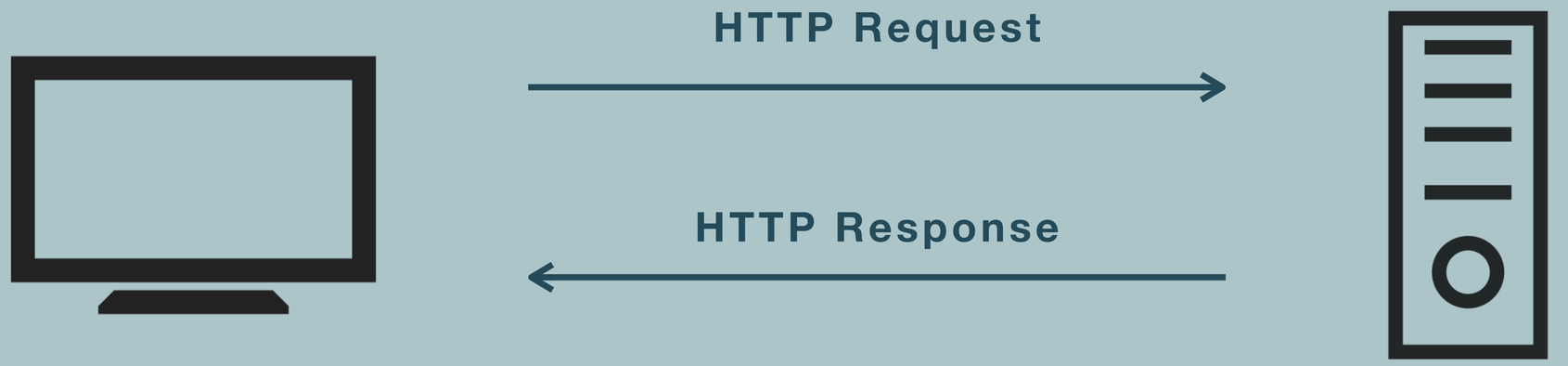
SERVER

Client-Server Model

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

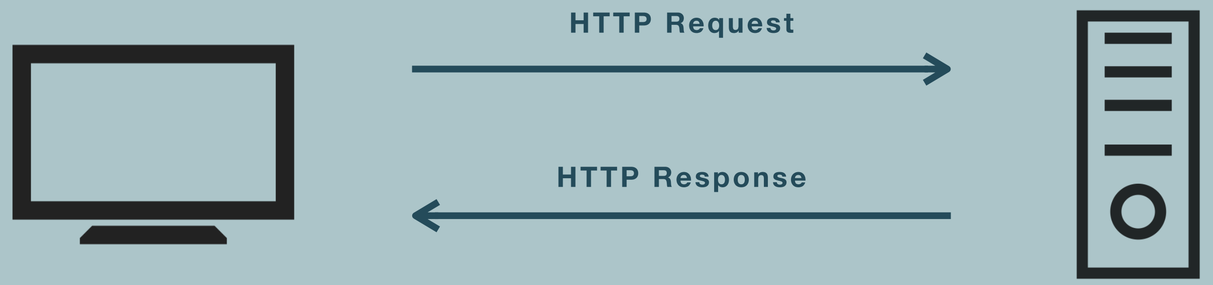
Client-Server Model

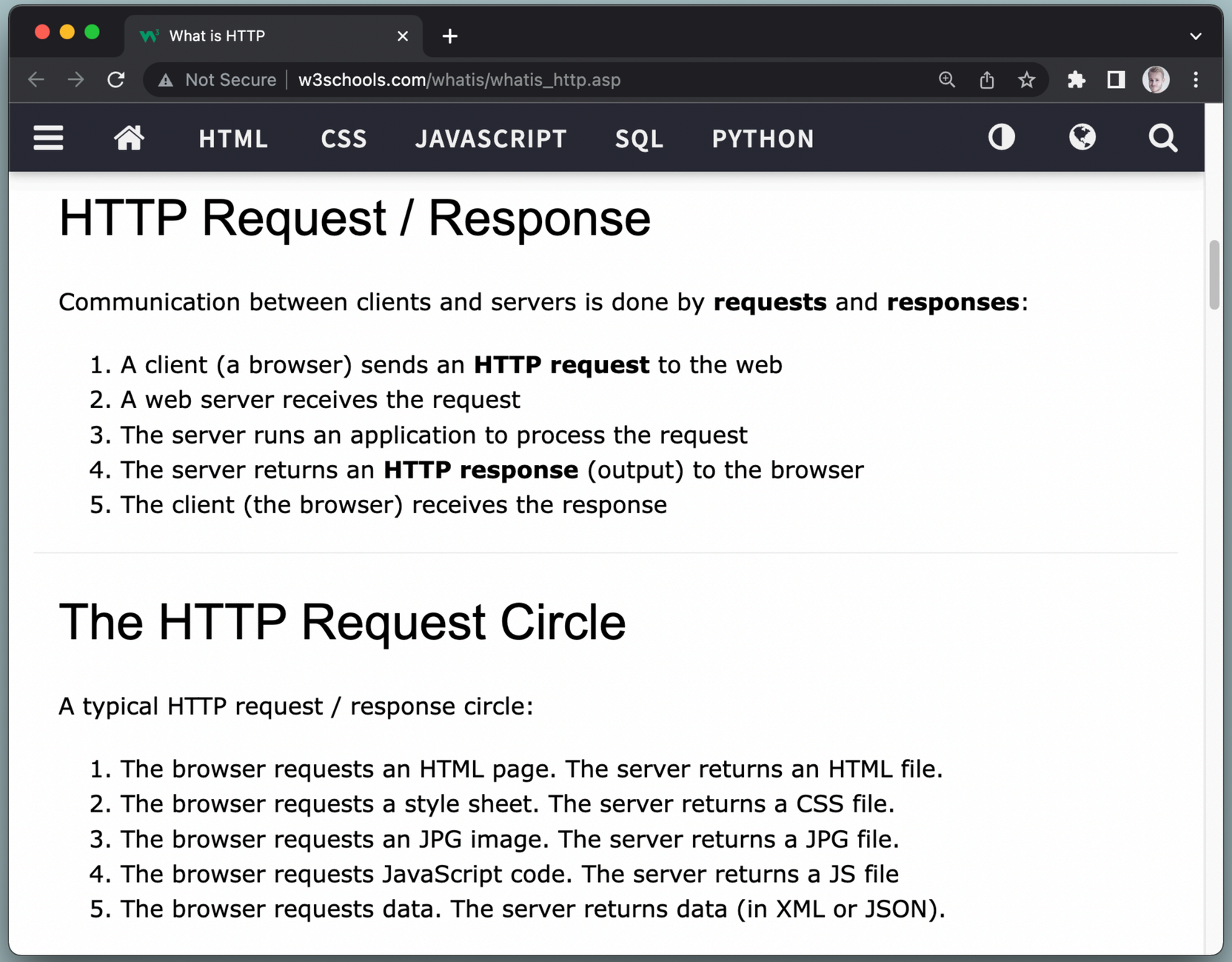
Communication between web **clients** and web **servers**.



Hyper Text Transfer Protocol

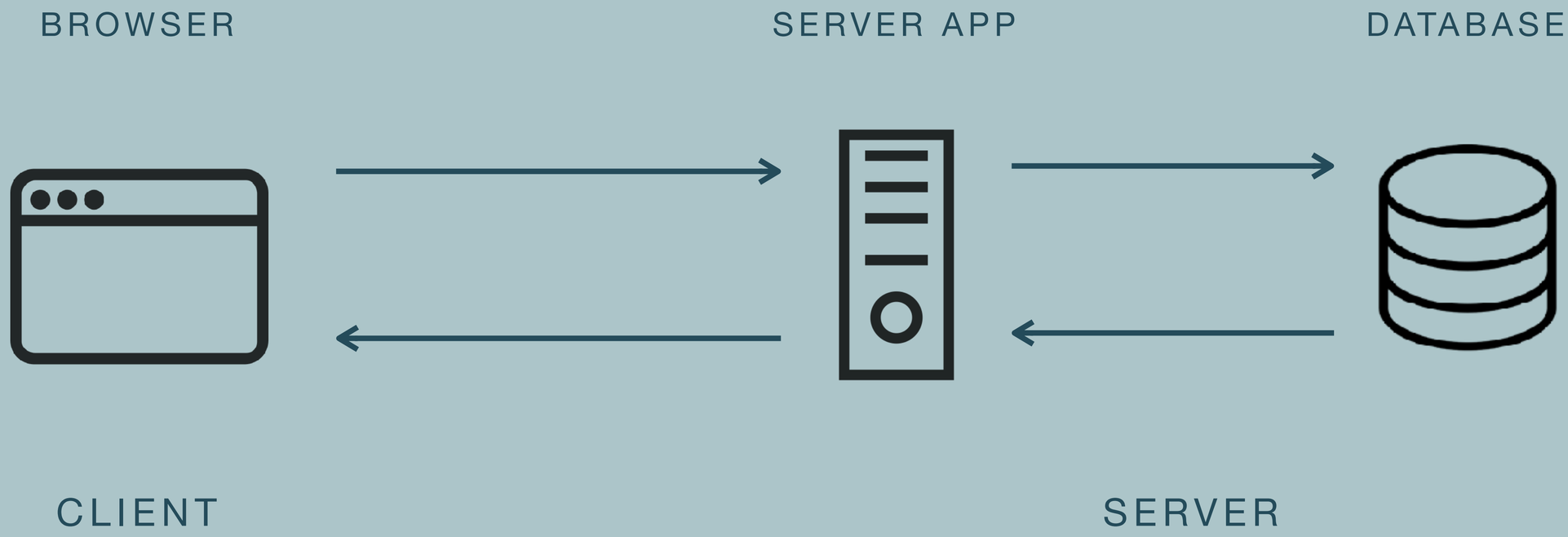
- A protocol and standard for fetching data, HTML and other resources (text, images, videos, scripts, JSON).
- The foundation of the web.





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

Web Dev Architecture

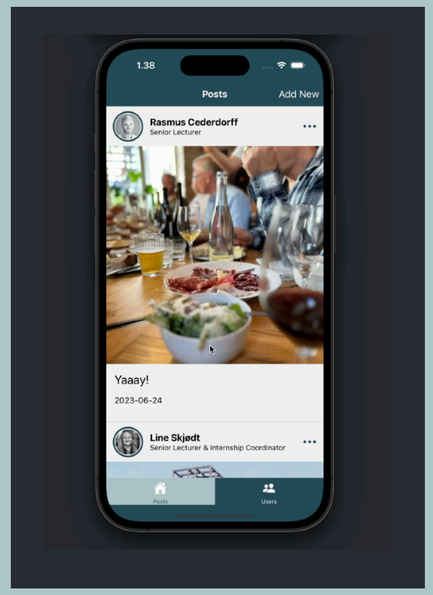


Web Dev Architecture

FRONTEND

REST API

BACKEND



REQUEST

JSON

RESPONSE

SQL

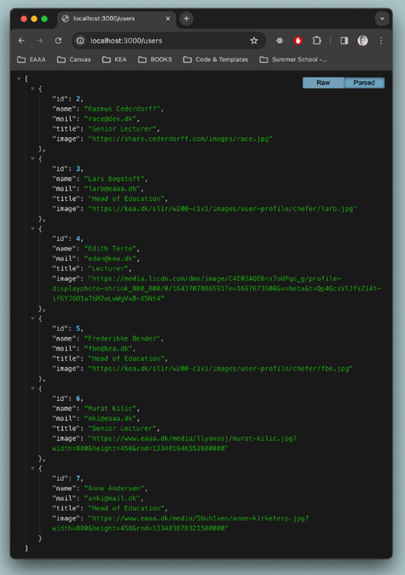
CLIENT

SERVER

Web Dev Architecture

REST API

BACKEND



REQUEST

JSON

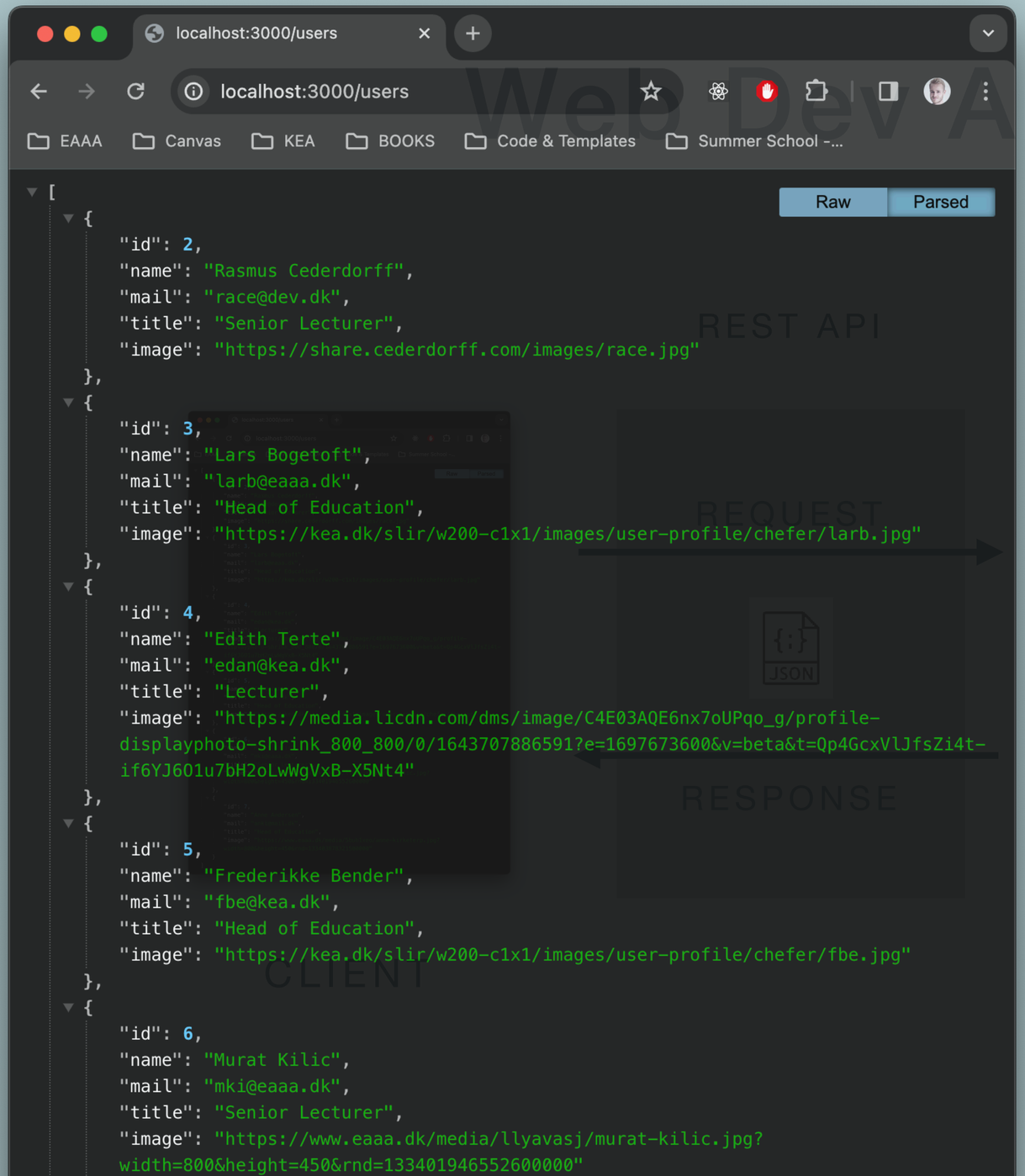
RESPONSE

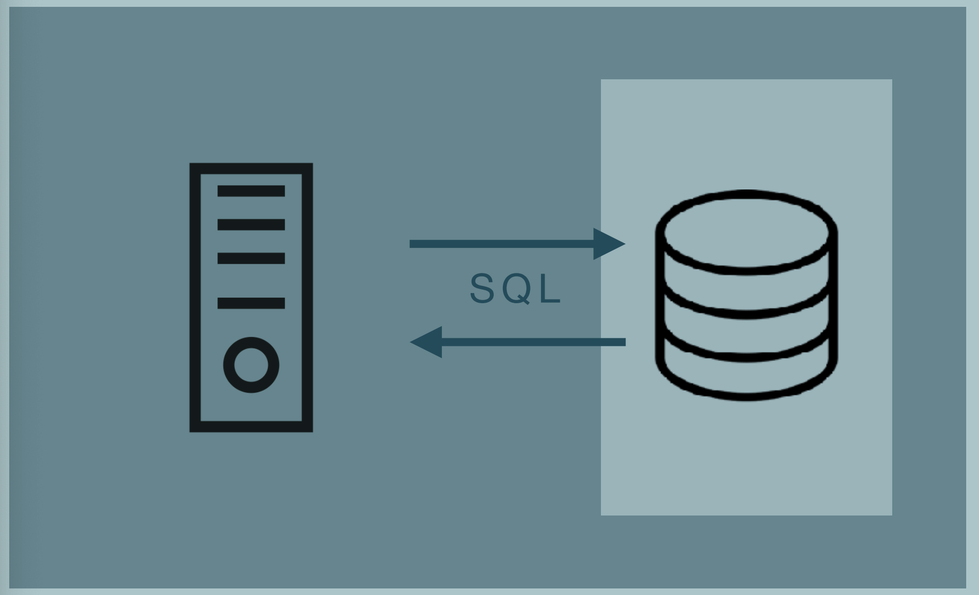
SQL

CLIENT

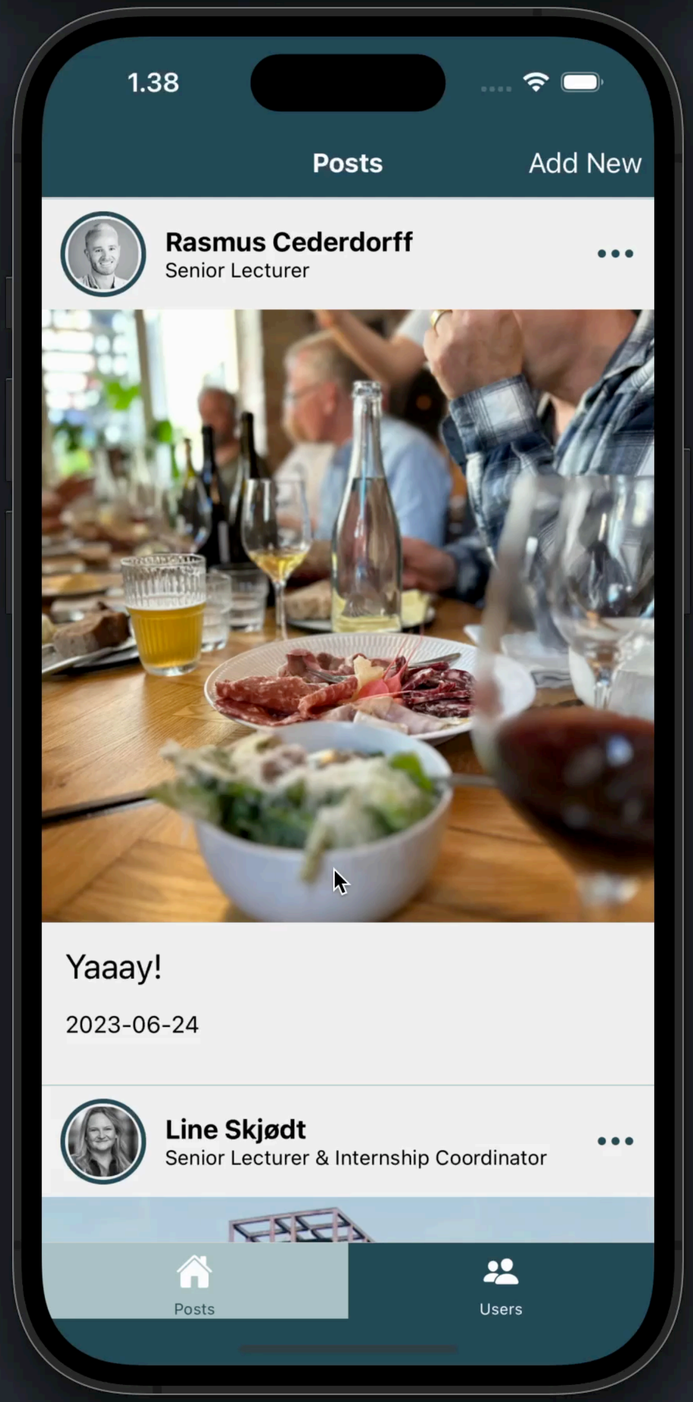
SERVER

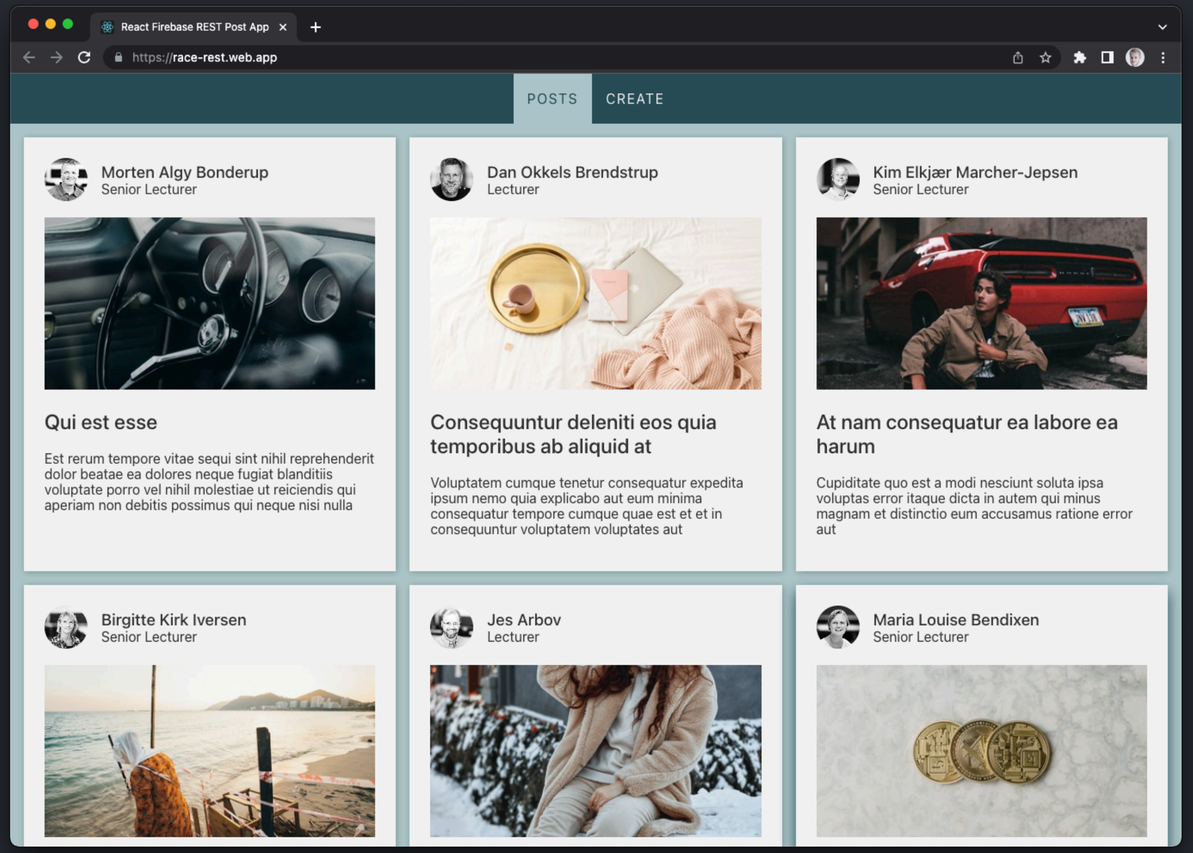
Web Dev Architecture



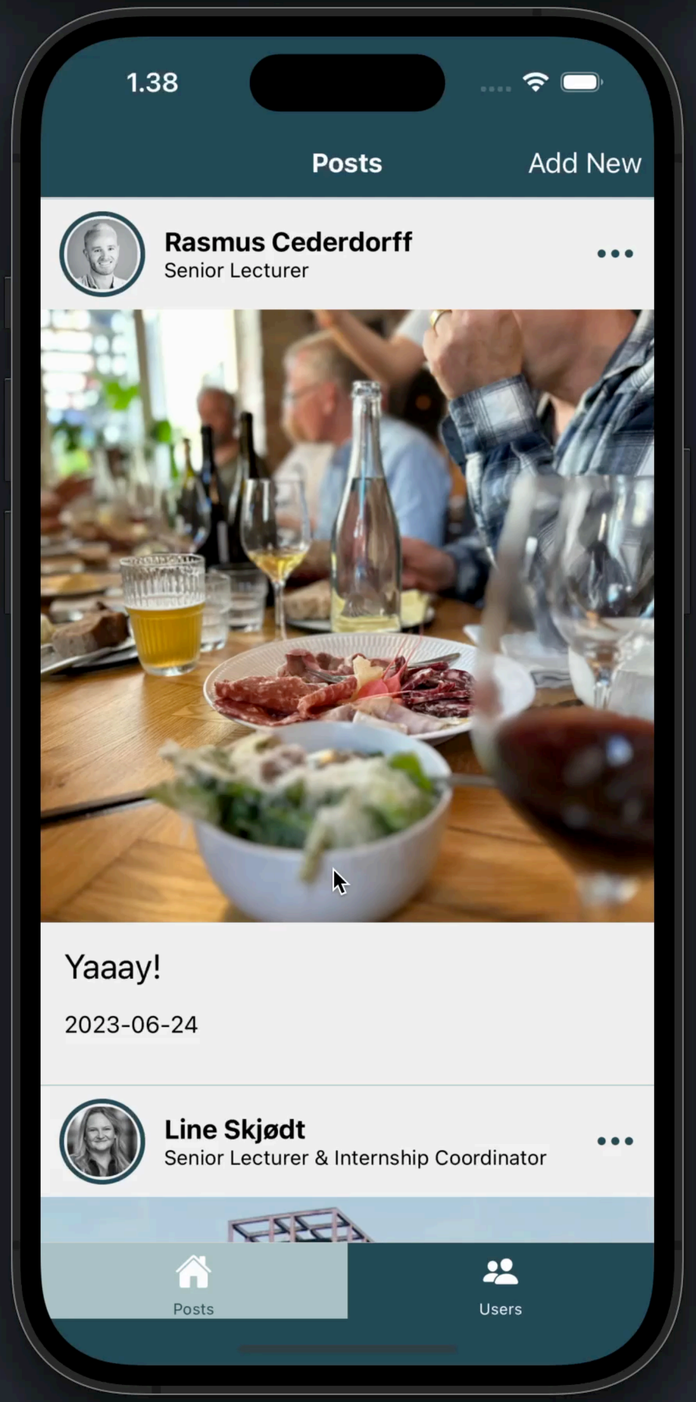


Posts & Users





posts users



Data from tables

Posts

	id	caption	image	createdAt
1	1	Beautiful sunset at the beach	https://images.unsplash.com/photo-1566241832378-917a0f30db2c?ixlib=rb-4.0.3&i…	2023-09-13 16:51:07
2	2	Exploring the city streets of Aarhus	https://images.unsplash.com/photo-1559070169-a3077159ee16?ixlib=rb-4.0.3&ixid…	2023-09-13 16:51:07
3	3	Delicious food at the restaurant	https://images.unsplash.com/photo-1548940740-204726a19be3?ixlib=rb-4.0.3&ixid…	2023-09-13 16:51:07
4	4	Exploring the city center of Aarhus	https://images.unsplash.com/photo-1612624629424-ddde915d3dc5?ixlib=rb-4.0.3&i…	2023-09-13 16:51:07
5	5	A cozy morning with coffee	https://images.unsplash.com/photo-1545319261-f3760f9dd64d?ixlib=rb-4.0.3&ixid…	2023-09-13 16:51:07
6	6	Serenity of the forest	https://images.unsplash.com/photo-1661505216710-32316e7b5bb3?ixlib=rb-4.0.3&i…	2023-09-13 16:51:07
7	7	A beautiful morning in Aarhus	https://images.unsplash.com/photo-1573997953524-efed43db70a0?ixlib=rb-4.0.3&i…	2023-09-13 16:51:07
8	8	Rainbow reflections of the city of Aarhus	https://images.unsplash.com/photo-1558443336-dbb3de50b8b2?ixlib=rb-4.0.3&ixid…	2023-09-13 16:51:07

Users

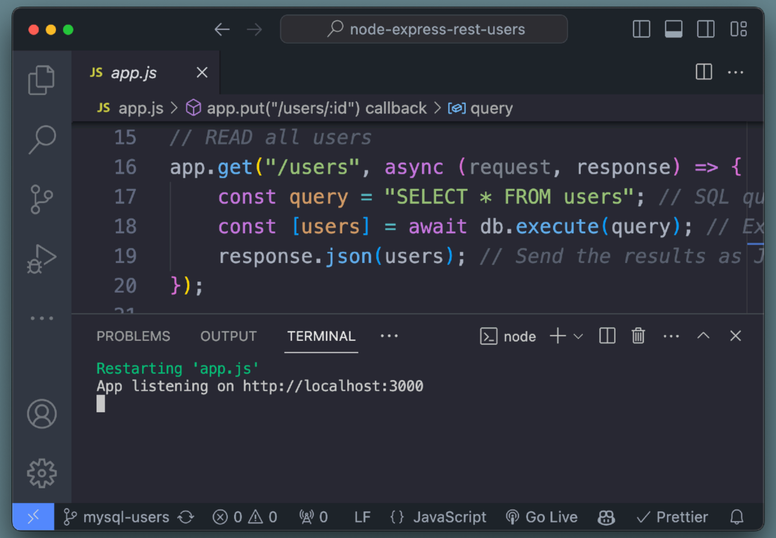
	id	name	mail	title	image
1	1	Maria Louise Bendixen	mlbe@eaaa.dk	Senior Lecturer	https://www.baaa.dk/media/b5ahrlra/maria-loui…
2	2	Rasmus Cederdorff	race@eaaa.dk	Senior Lecturer	https://share.cederdorff.com/images/race.jpg
3	3	Anne Kirketerp	anki@eaaa.dk	Head of Department	https://www.baaa.dk/media/5buh1xeo/anne-kirke…
4	4	Line Skjødt	lskj@eaaa.dk	Senior Lecturer & Internship Coord…	https://www.eaaa.dk/media/14qpfeq4/line-skjod…
5	5	Dan Okkels Brendstrup	dob@eaaa.dk	Lecturer	https://www.eaaa.dk/media/bdojel41/dan-okkels…

BACKEND

REQUEST

JSON

RESPONSE



SQL

	id	name	mail
1	1	Maria Louise Bendixen	mlbe@eaaa.dk
2	2	Rasmus Cederdorff	race@eaaa.dk
3	3	Anne Kirketerp	anki@eaaa.dk
4	4	Line Skjødt	lskj@eaaa.dk
5	5	Dan Okkels Brendstrup	dob@eaaa.dk

CLIENT

SERVER

Objects

A set of named values

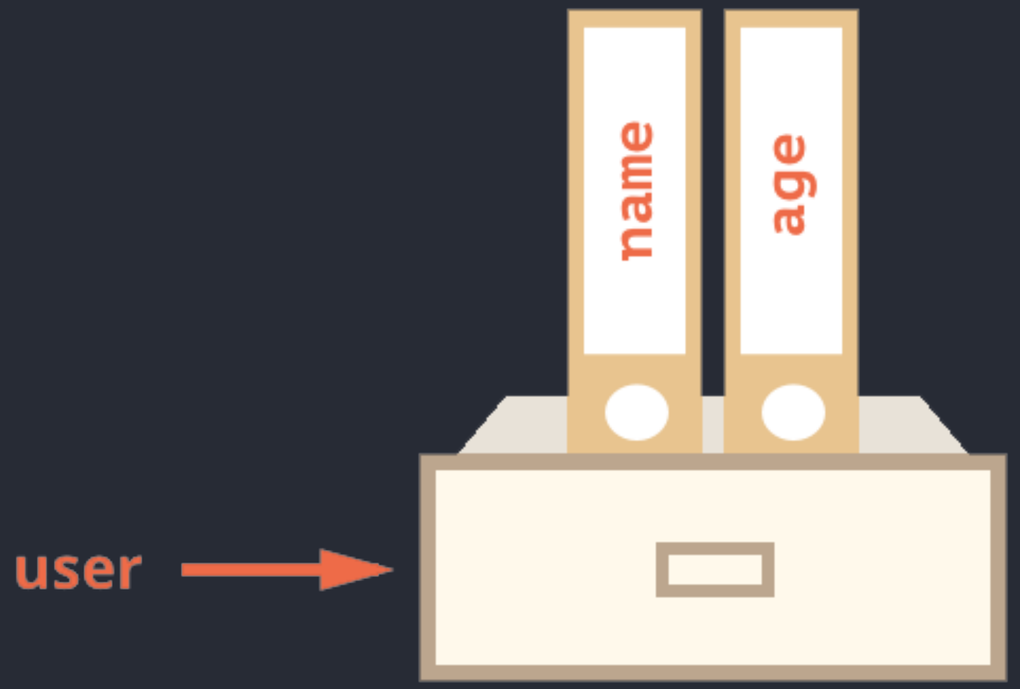
Objects are used to store keyed collections of various data

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

Objects

A set of named values

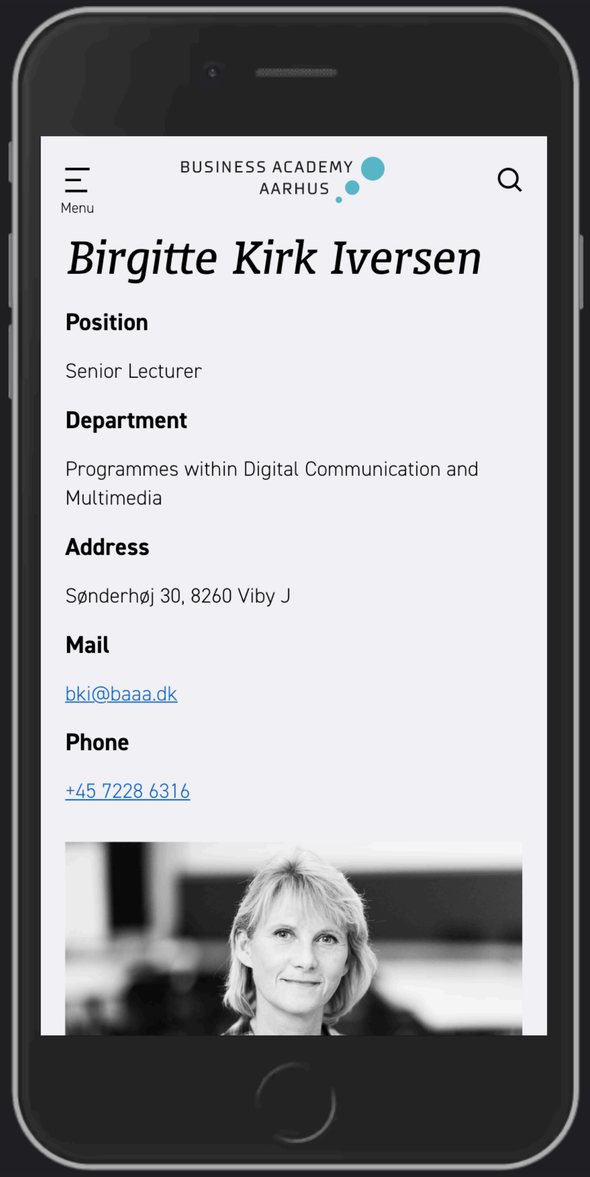
```
let user = {
  name: 'Alicia',
  age: 6
};

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




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

Objects

A set of named values



property

value

```
let person = {
    name: "Birgitte Kirk Iversen",
    position: "Senior Lecturer",
    department: "Programmes within Digital Communication and Multimedia",
    address: "Sønderhøj 30, 8260 Viby J",
    mail: "bki@baaa.dk",
    phone: "+45 7228 6316",
    image: "https://www.baaa.dk/media/u4gorzsd/birgitte-kirk-iversen2.jpg"
};
```

https://www.baaa.dk/contact/find-employee/employee/birgitte-kirk-iversen

netflix.com

NETFLIX

Hvem ser?

Personen der rent faktisk betaler for profilen

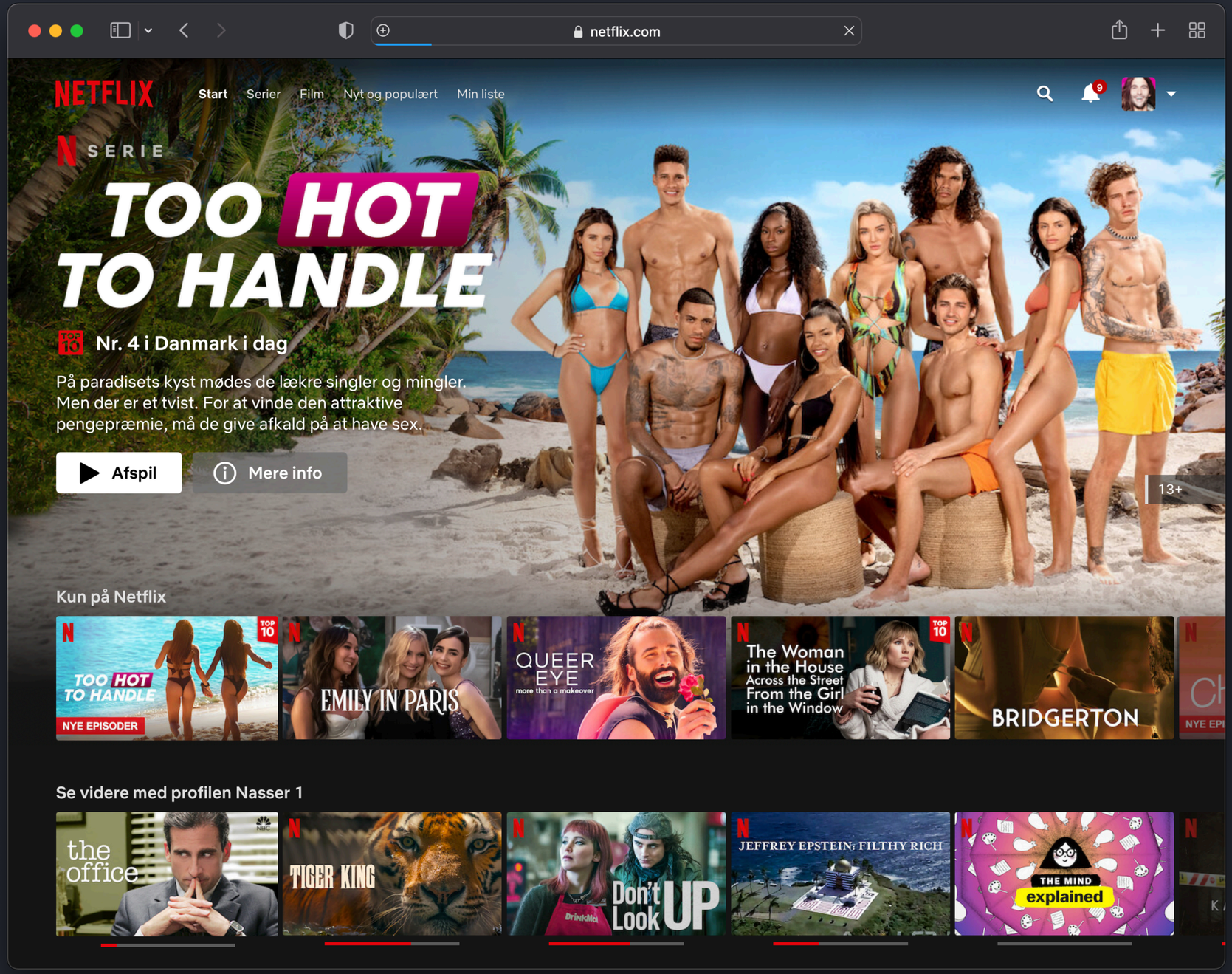
Nasser 1

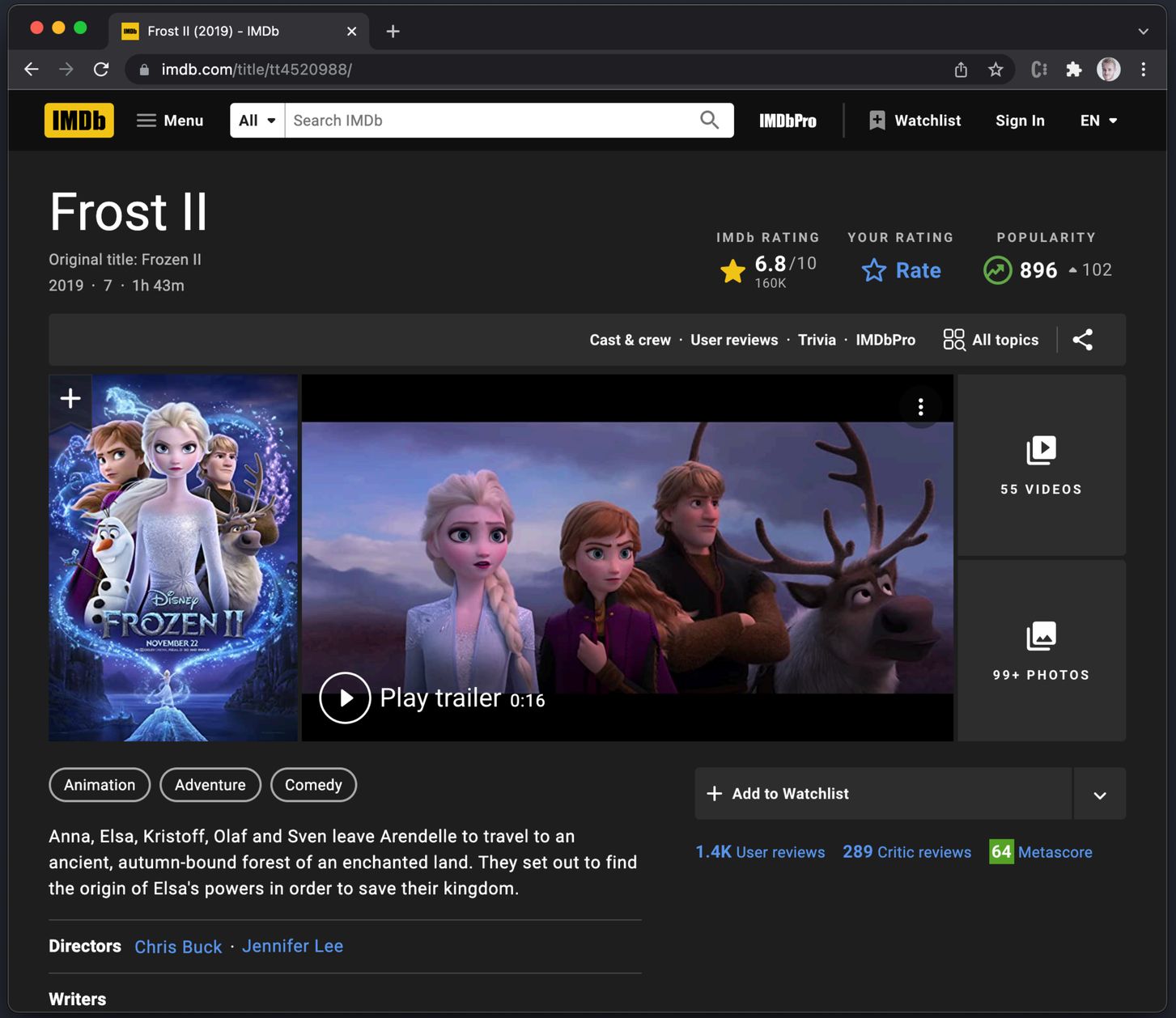
Nasser 2

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"
}
```

Define yourself as an object

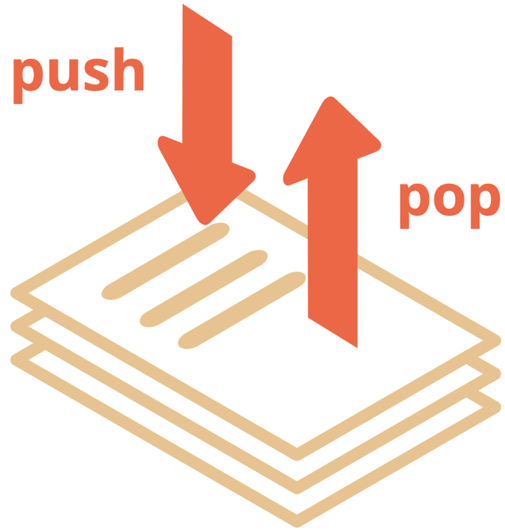
with the following properties

`name, age, mail, phone, city, address`

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.

CRUD App

127.0.0.1:5500/index.html

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

CRUD App

Rasmus Cederdorff

Senior Lecturer
race@dev.dk

UPDATE DELETE

Lars Bogetoft

Head of Education
larb@eaaa.dk

UPDATE DELETE

Edith Terte

Lecturer
edan@kea.dk

UPDATE DELETE

Frederikke Bender

Head of Education
fbe@kea.dk

UPDATE DELETE

Murat Kilic

Senior Lecturer
mki@eaaa.dk

UPDATE DELETE

Anne Andersen

Head of Education
anki@mail.dk

UPDATE DELETE

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

Preserve log | Disable cache | No throttling

Filter | Invert | Hide data URLs | Hide extension URLs

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

Blocked requests | 3rd-party requests

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

Name | Headers | Preview | Response | Initiator | Timing

users

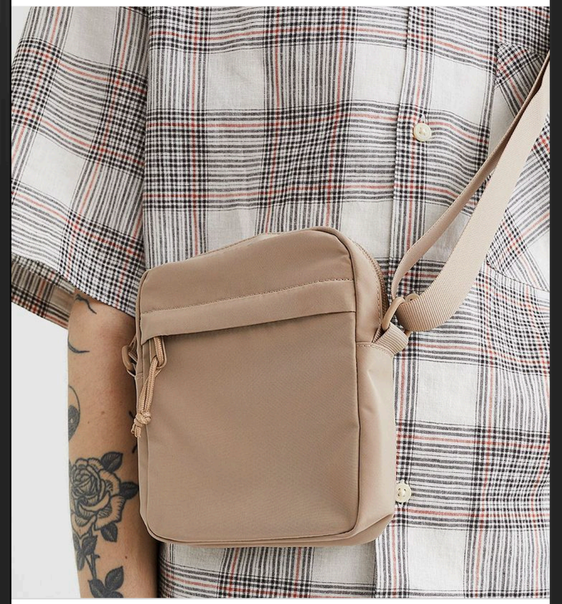

```
[
    {
        "id": 2,
        "name": "Rasmus Cederdorff",
        "mail": "race@dev.dk",
        "title": "Senior Lecturer",
        "image": "https://share.cederdorff.com/images/race.jpg"
    },
    {
        "id": 3,
        "name": "Lars Bogetoft",
        "mail": "larb@eaaa.dk",
        "title":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


1 / 11 requests | 21

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

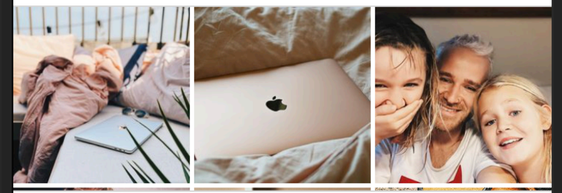
cederdorff

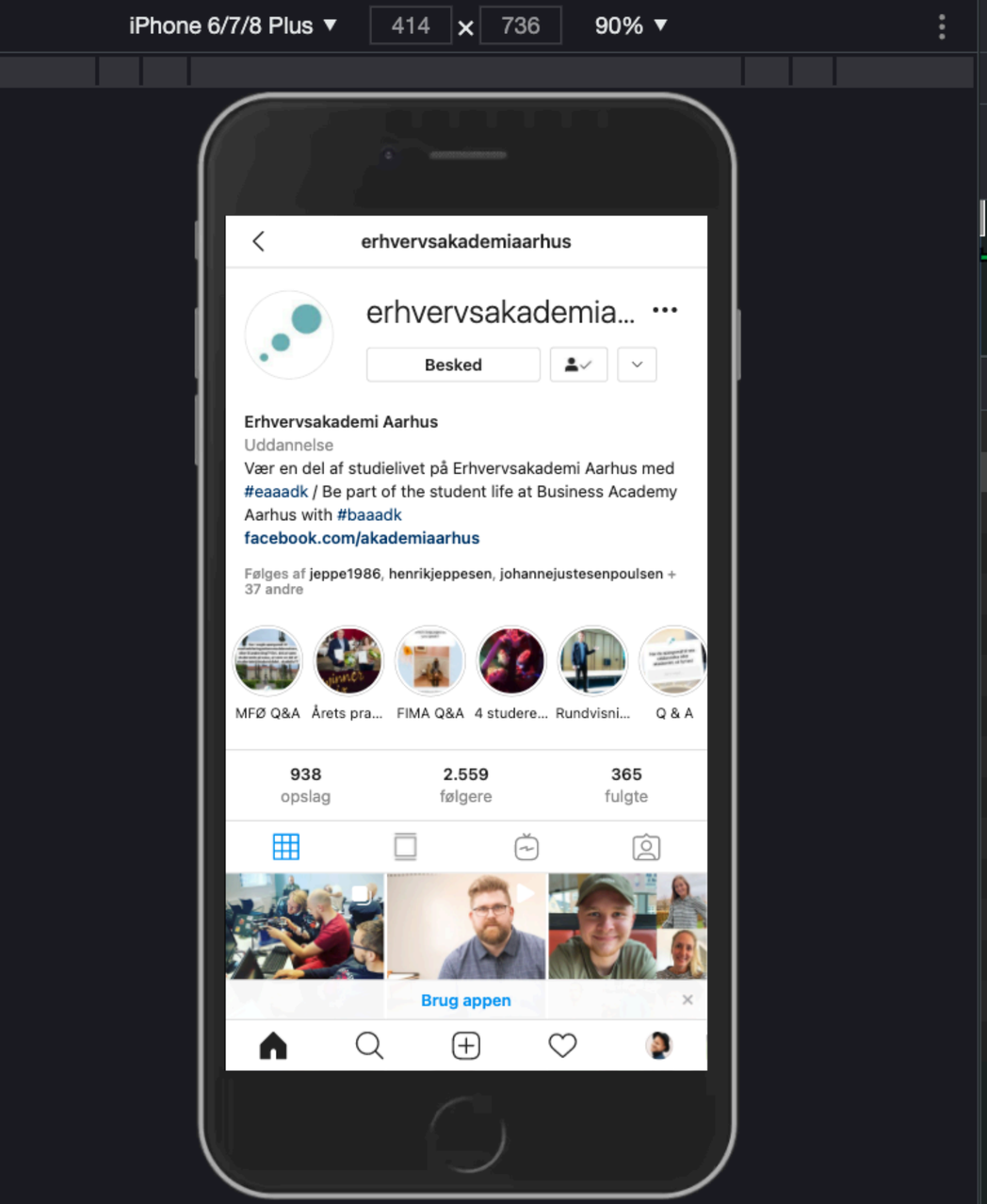
Edit Profile

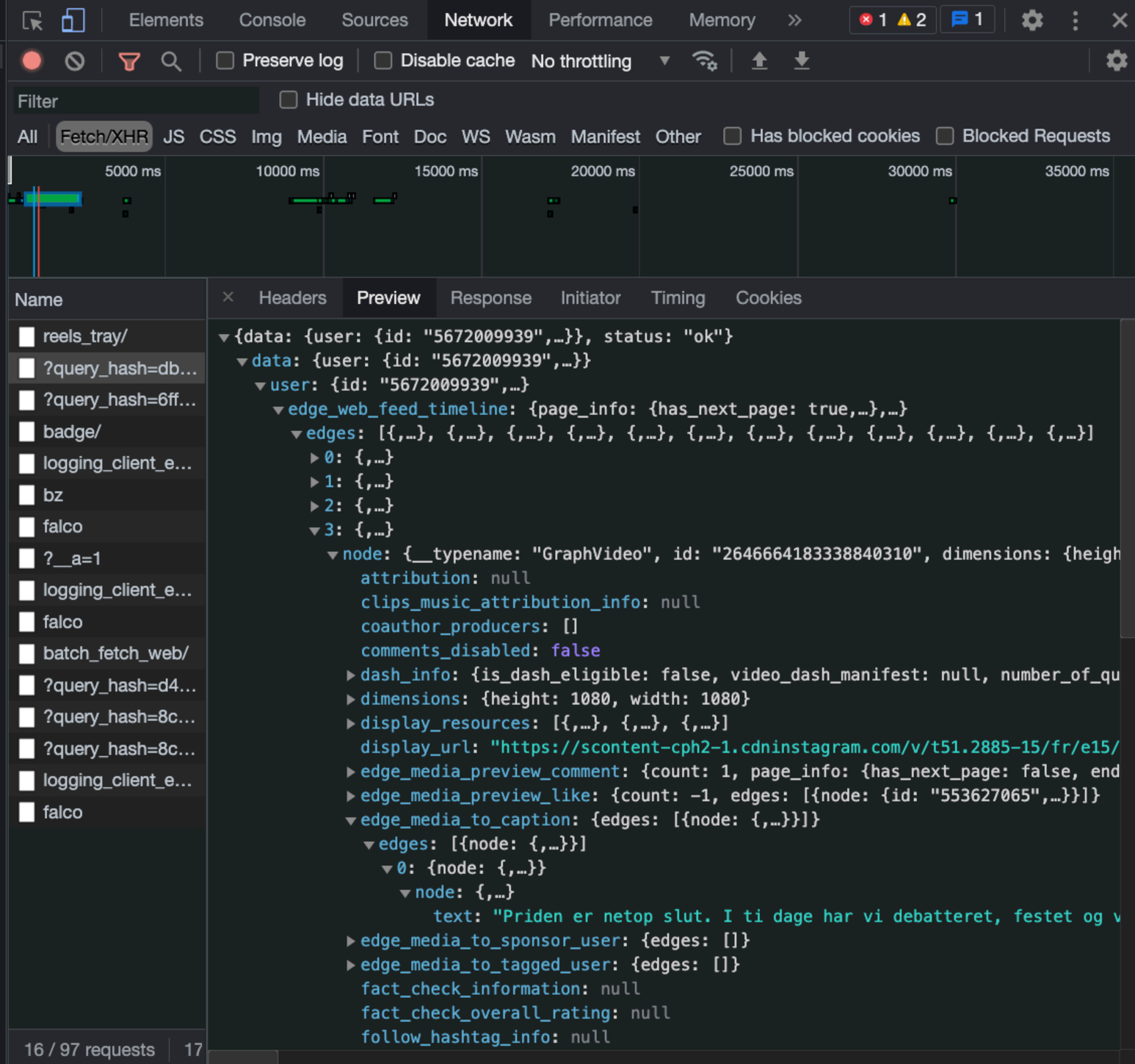
Rasmus Cederdorff
Senior Lecturer & Web App Developer
Alicia & Ida
Aarhus, Denmark
cederdorff.com

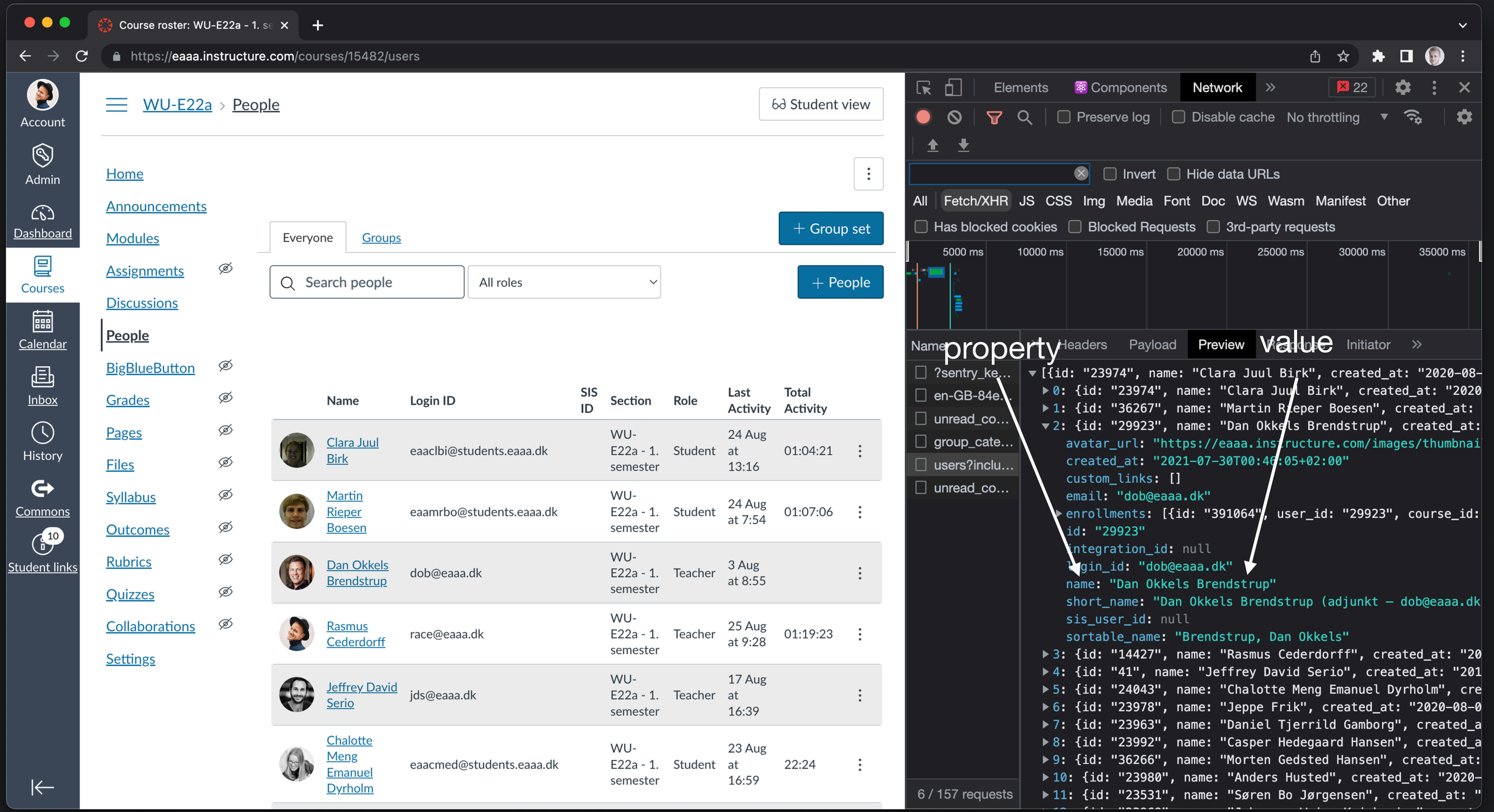
AMSTER... | LONDON ... | Karla & Pi... | @villacph | IDA

181	5,441	635
posts	followers	following



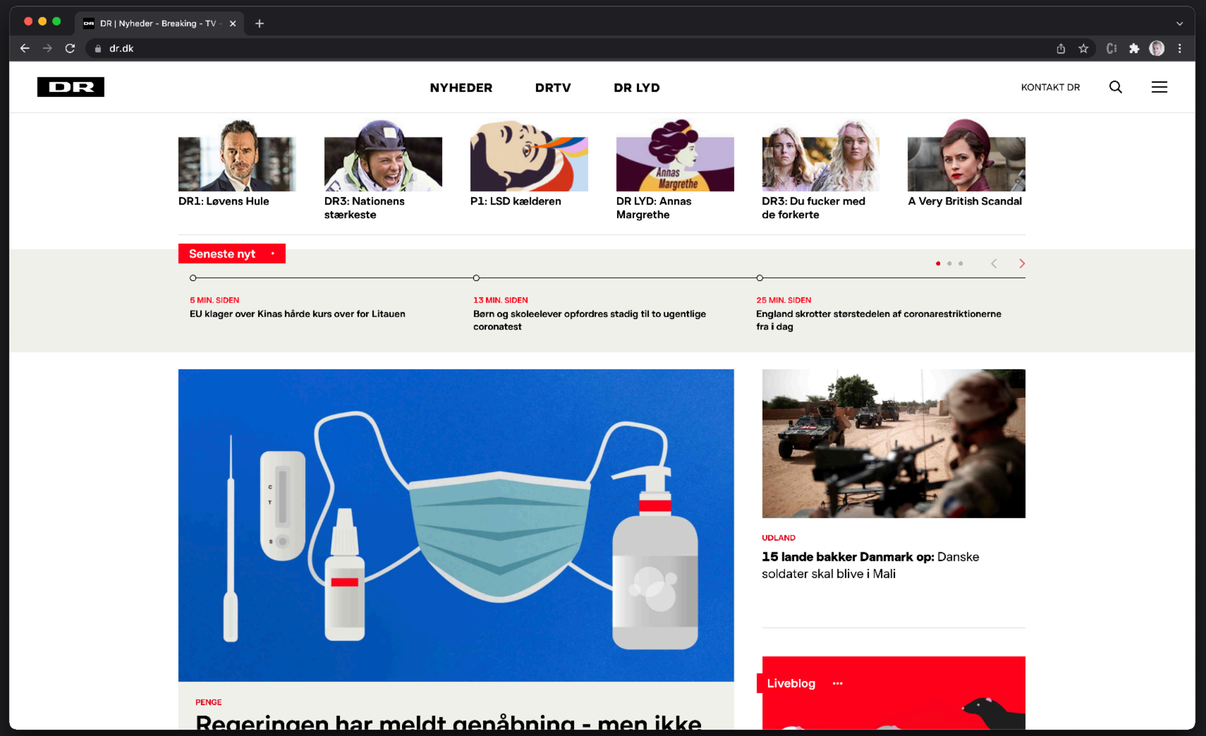

iPhone 6/7/8 Plus
414
736
90%
erhvervsakademiaarhus
erhvervsakademia...
Besked
Erhvervsakademi Aarhus
Uddannelse
Vær en del af studielivet på Erhvervsakademi Aarhus med #eaaadk / Be part of the student life at Business Academy Aarhus with #baaadk
facebook.com/akademiaarhus
Følges af jeppe1986, henrikjeppesen, johannejustesenpoulsen + 37 andre
MFØ Q&A
Årets pra...
FIMA Q&A
4 studere...
Rundvisni...
Q & A
938
opslag
2.559
følgere
365
fulgte
Brug appen

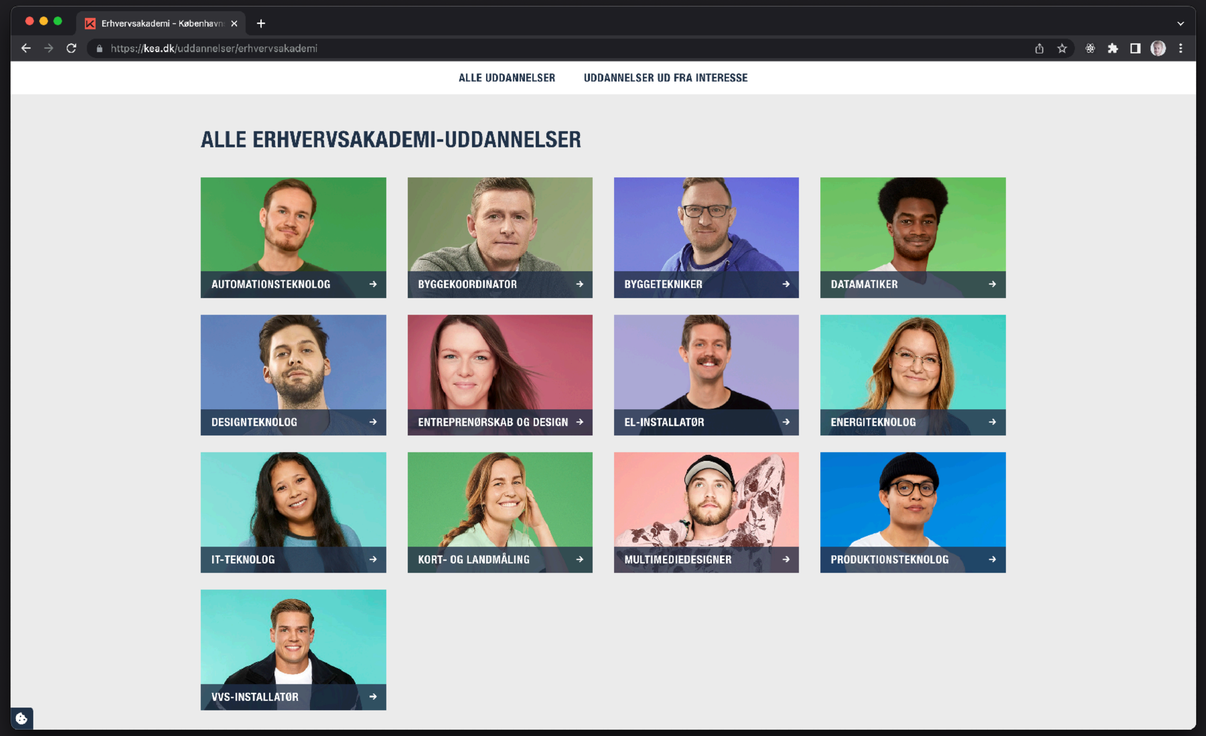

Elements
Console
Sources
Network
Performance
Memory
Preserve log
Disable cache
No throttling
Filter
Hide data URLs
All
Fetch/XHR
JS
CSS
Img
Media
Font
Doc
WS
Wasm
Manifest
Other
Has blocked cookies
Blocked Requests
5000 ms
10000 ms
15000 ms
20000 ms
25000 ms
30000 ms
35000 ms
Name
reels_tray/
?query_hash=db...
?query_hash=6ff...
badge/
logging_client_e...
bz
falco
?__a=1
logging_client_e...
falco
batch_fetch_web/
?query_hash=d4...
?query_hash=8c...
?query_hash=8c...
logging_client_e...
falco
16 / 97 requests
17
Headers
Preview
Response
Initiator
Timing
Cookies
{data: {user: {id: "5672009939",…}}, status: "ok"}
data: {user: {id: "5672009939",…}}
user: {id: "5672009939",…}
edge_web_feed_timeline: {page_info: {has_next_page: true,…},…}
edges: [{,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}, {,…}]
0: {,…}
1: {,…}
2: {,…}
3: {,…}
node: {__typename: "GraphVideo", id: "2646664183338840310", dimensions: {heigh
attribution: null
clips_music_attribution_info: null
coauthor_producers: []
comments_disabled: false
dash_info: {is_dash_eligible: false, video_dash_manifest: null, number_of_qu
dimensions: {height: 1080, width: 1080}
display_resources: [{,…}, {,…}, {,…}]
display_url: "https://scontent-cph2-1.cdninstagram.com/v/t51.2885-15/fr/e15/
edge_media_preview_comment: {count: 1, page_info: {has_next_page: false, end
edge_media_preview_like: {count: -1, edges: [{node: {id: "553627065",…}}]}
edge_media_to_caption: {edges: [{node: {,…}}]}
edges: [{node: {,…}}]
0: {node: {,…}}
node: {,…}
text: "Priden er netop slut. I ti dage har vi debatteret, festet og v
edge_media_to_sponsor_user: {edges: []}
edge_media_to_tagged_user: {edges: []}
fact_check_information: null
fact_check_overall_rating: null
follow_hashtag_info: null



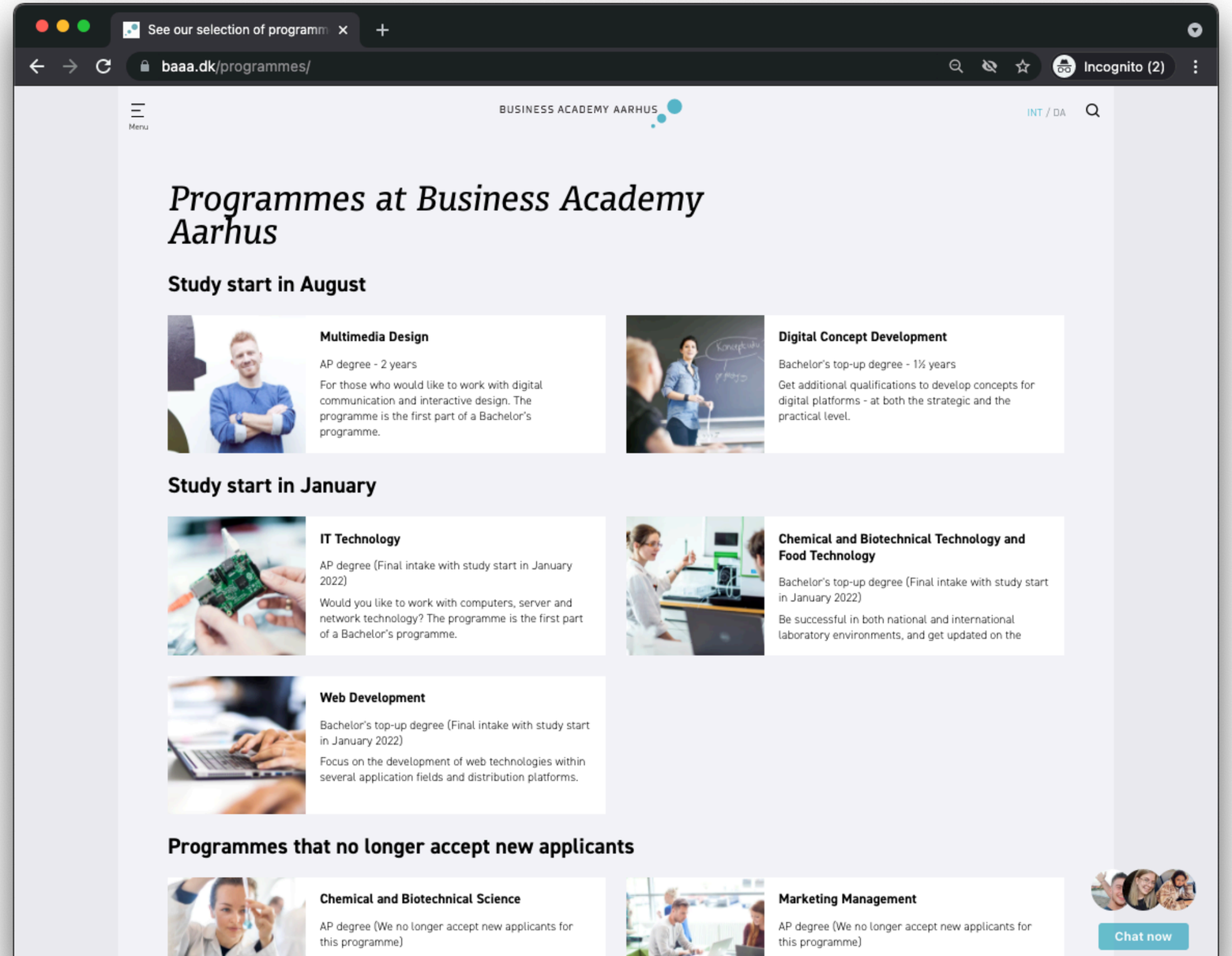
eaaa.instructure.com/api/v1/courses/15482/users?include_inactive=true&include%5B%5D=avatar_url&include%5B%5D=email&include%5B%5D=observed_users&per_page=50

Objects? Arrays?

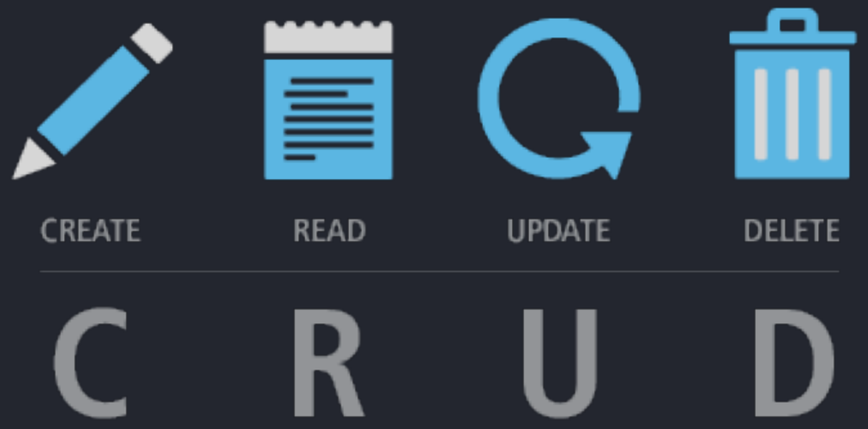




Objects with properties in arrays



It’s all objects & arrays!



What's CRUD?

- CREATE objects like a post, user, movie, product, etc.
- READ objects like an array (or object) of objects (posts, users, movies, products, etc)
- UPDATE an object, often given by a unique id.
- DELETE an object, often given by a unique id.

codecademy.com/article/what-is-crud

GET

POST

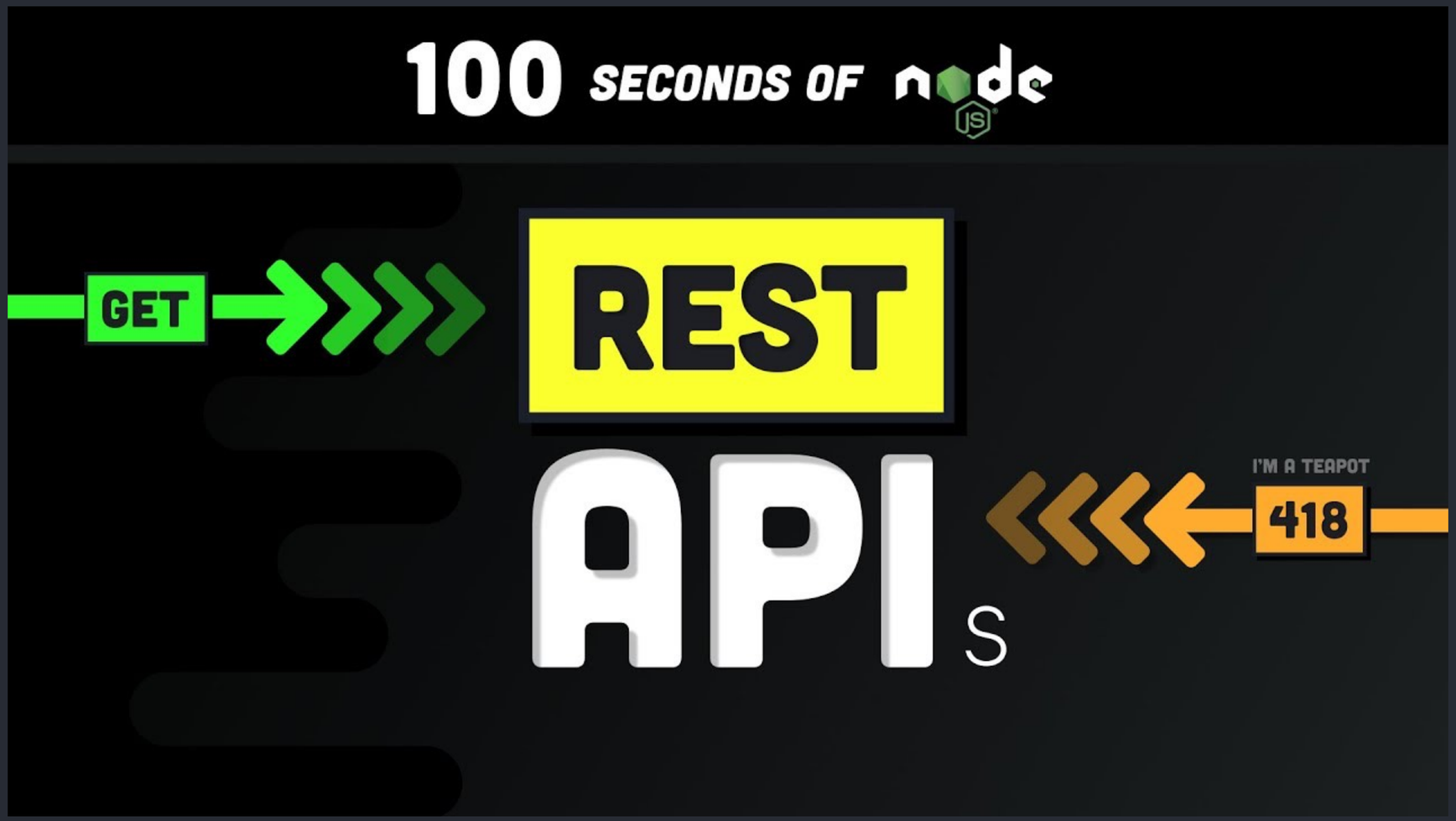
PUT

DELETE

What’s REST?

- REpresentational State Transfer
- A standard for systems (client & server) to communicate over HTTP in order to retrieve or modify (data) resources.
- Stateless, meaning the two systems doesn’t need to know anything about the state.
- The client makes the requests using the 4 basic HTTP verbs to define the operation.

codecademy.com/article/what-is-rest


100 SECONDS OF node
GET
REST
APIs
I'M A TEAPOT
418

CRUD vs REST & HTTP Verbs

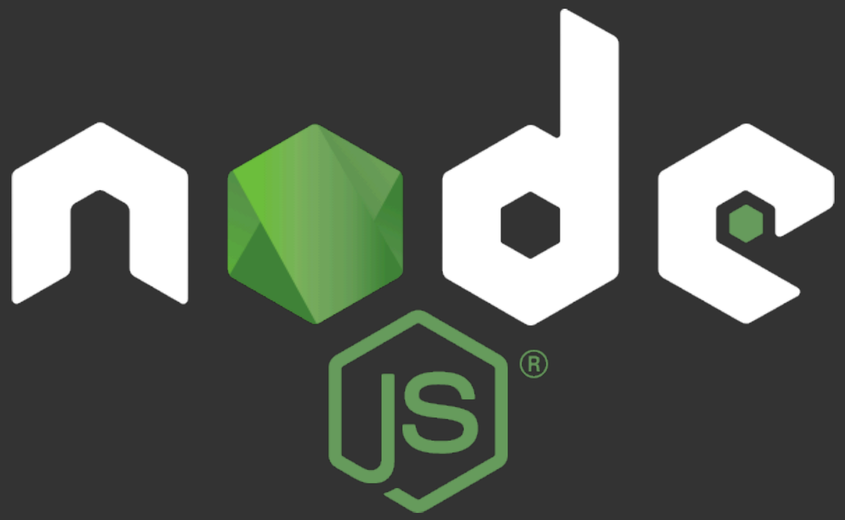
CREATE —> `POST`: create a new resource (object)

READ —> `GET`: retrieve a specific resource or a collection

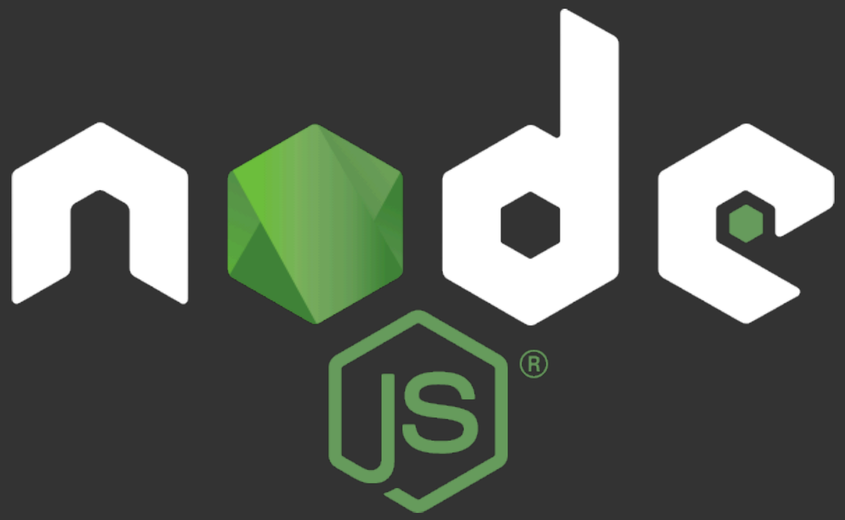
UPDATE —> `PUT`: update a specific resource (by id)

DELETE —> `DELETE`: remove a specific resource by id

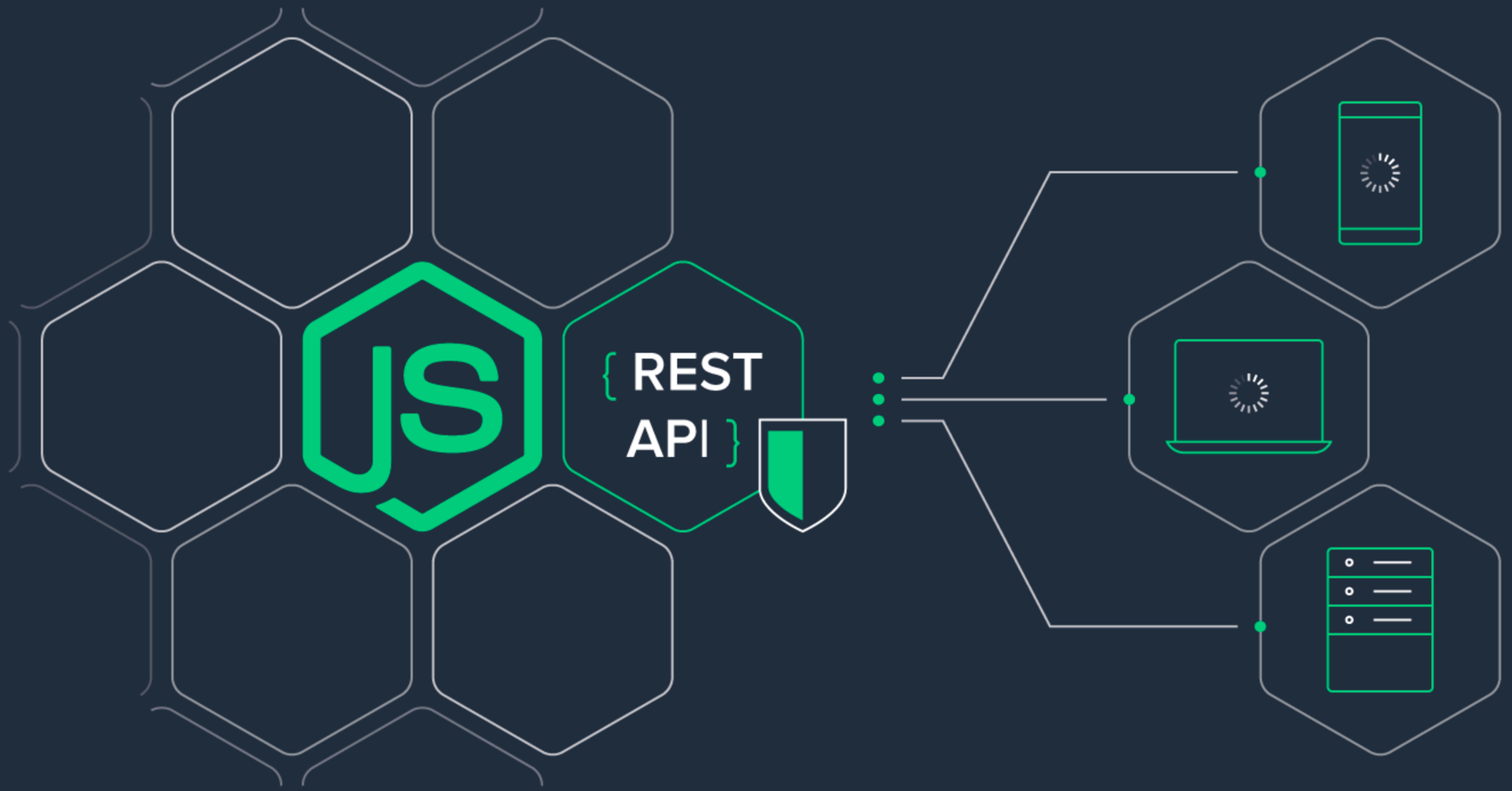
codecademy.com/article/what-is-rest



- Server-side JavaScript runtime environment.
- Executes JavaScript outside of the browser.

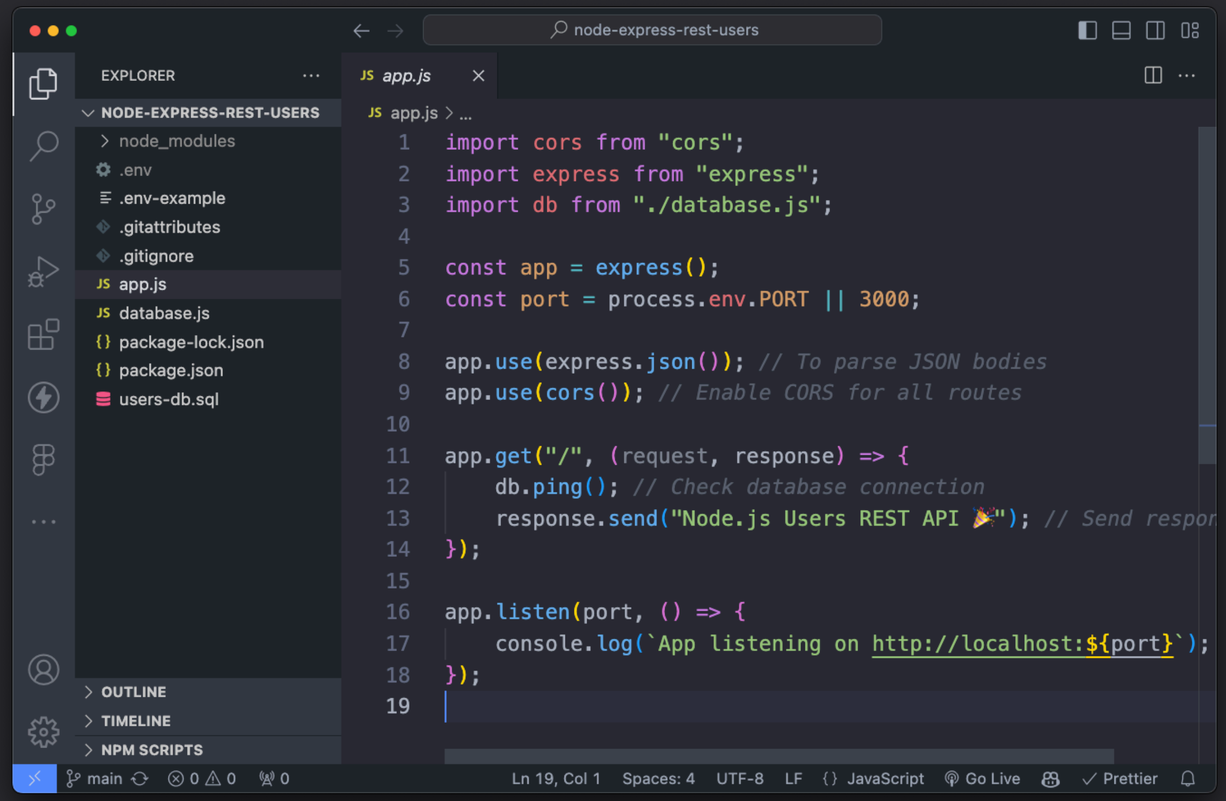


- Imagine Node.js as a powerful engine that lets you run JavaScript code outside of a web browser.
- Instead of just using JavaScript for web pages, Node.js lets you use it on your computer or on a server.
- It's like having a super-fast computer that can understand and do all the smart things you do in a web browser with JavaScript, but now it can do it on your own computer or a server.


JS
{ REST
API }

Øvelse

Node.js & MySQL



- **JavaScript Beyond Browser**
  Instead of being limited to doing things only on the internet, Node.js lets you use JavaScript to build programs and applications on your computer or server. You get to use the same language you know from web pages.
- **Fast and Efficient**
  Node.js is built to be really fast and efficient. It can handle many things at once without getting slow. This is really useful when you're working with apps that need to do lots of things at the same time, like chat apps or real-time updates.
- **Don't Wait for Things**
  Usually, when a computer does something, it has to wait until that task is finished before moving on to the next one. Node.js does things differently. It can start a task and then keep working on other things while it waits for the first task to finish.
- **Customize with Modules**
  Think of modules as little building blocks of code. Node.js gives you lots of ready-made code that you can use in your programs. You don't have to invent everything from scratch. It makes it easy to add new features to your projects.
- **Used for Many Things**
  People use Node.js to build all sorts of things, from simple web servers to complex applications. It's popular in modern web development because it allows fast and efficient communication between clients and servers.

Key Points

Runtime Environment

- Node.js provides a runtime environment that allows developers to use JavaScript beyond the browser, enabling them to create high-performance, scalable, and real-time applications for various use cases.

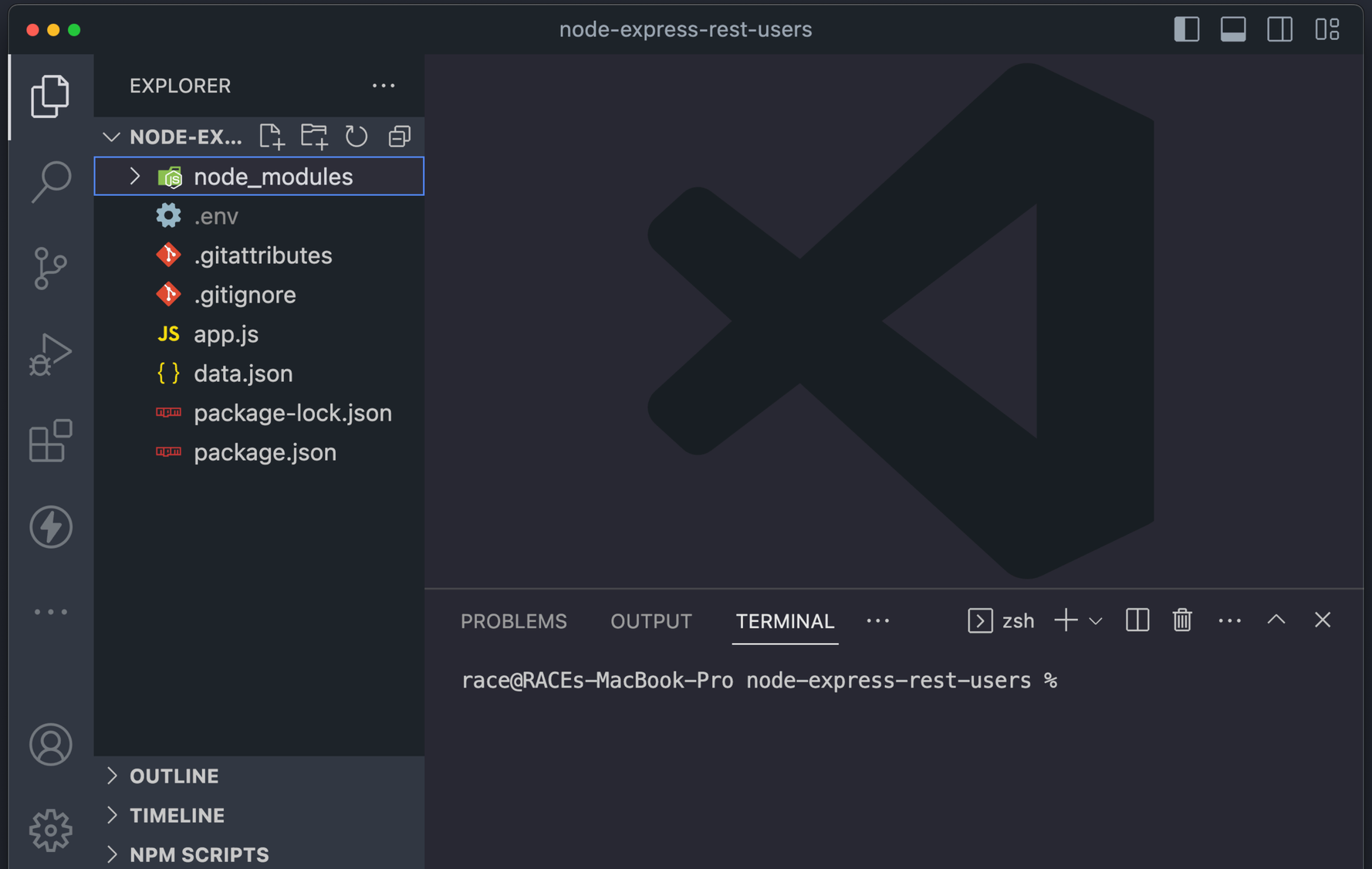
= Runtime Environment + JavaScript Library

Node.js also comes with a toolbox called NPM, which has all sorts of ready-made tools (like Lego blocks) that developers can use to build their programs. Instead of building everything from scratch, you can use these tools to save time and make your app even better.

`npm` is a package manager for Node.js packages (JS modules)

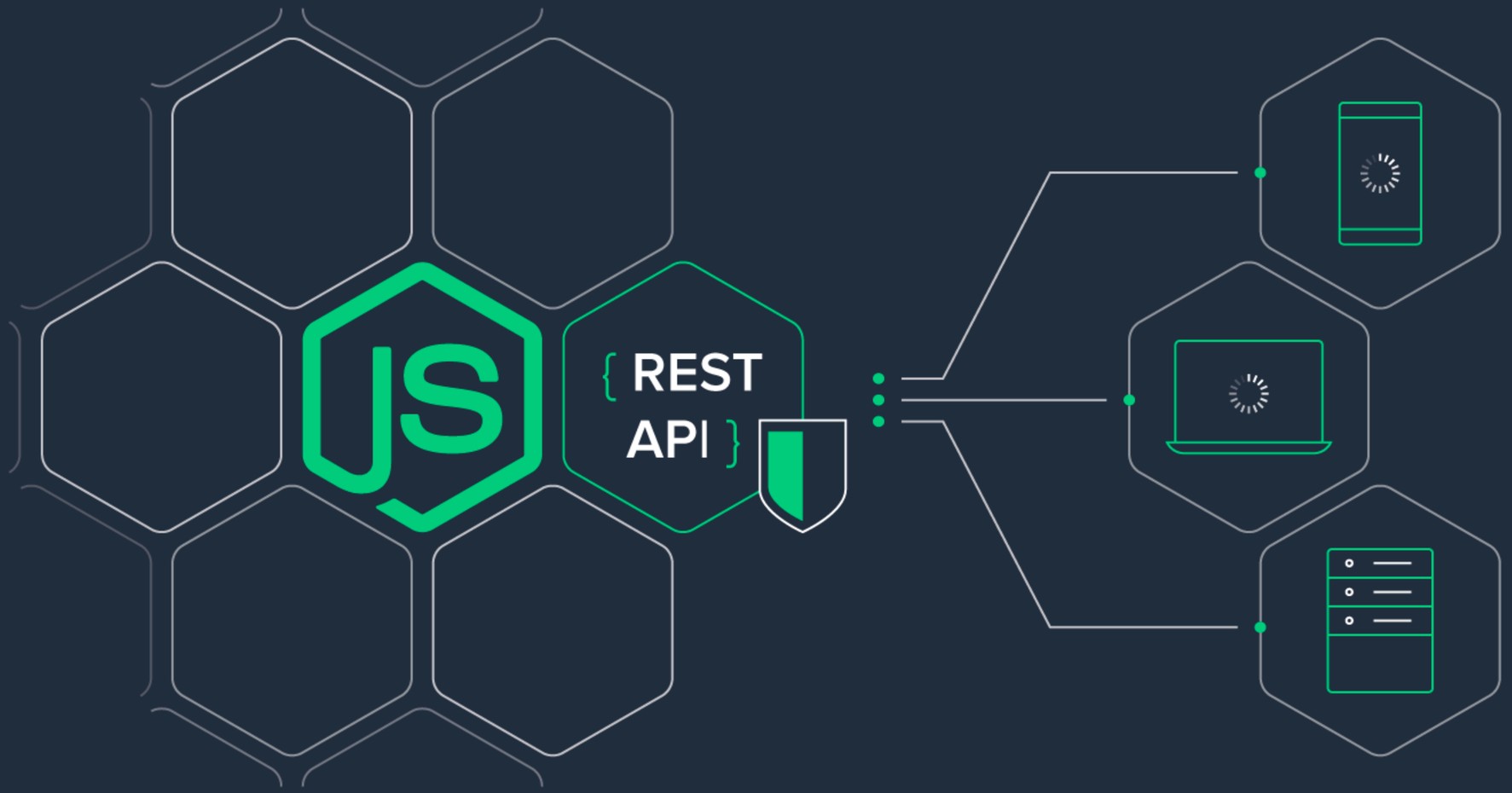
- Node.js packages contains all the files you need for a module.
- Modules are JavaScript libraries you can include in your project.

node_modules



- A popular, minimalist web application framework for building APIs and web applications using Node.js.
- It simplifies the process of building web applications and APIs by providing a set of tools and utilities.

Streamlines and simplifies the implementation of REST API



Server

Client

Incoming request URL

Router

GET “/posts”

POST “/posts”

PUT “/posts”

DELETE “/posts”

Based on the request URL and method, the router decides what to do

```
app.get("/posts", (req, res) => {
    // return all posts
});

app.post("/posts", (req, res) => {
    const post = req.body;
    // Create a new post based on th
});

app.put("/posts/:id", (req, res) =>
    const postId = req.params.id;
    const post = req.body;
    // Find and update the post with
});

app.delete("/posts/:id", (req, res)
    const postId = req.params.id;
    // Find and delete the post with
});
```

Routes in Express.js

In Node.js with Express.js, routes are a fundamental concept that helps you define **how your web application responds to different types of HTTP requests** at different URLs. Think of routes as the paths your application can take based on the URLs that users request.

Routes in Express.js

- The Express routing is built upon Node.js HTTP Module and it **simplifies the process of handling different URL paths and HTTP methods** in your web application.
- Express routing enables you to define routes for different endpoints, **simplifying request handling**.
- This enhances and streamlines code organization, and readability, and lets you focus on building functionality rather than intricate routing logic.

Routes in Express.js

- Routes determine how your application responds to specific URLs and HTTP methods.
- Each route combines an HTTP method (like GET, POST) with a URL path.
- When a client request matches a route, a corresponding route handler function is executed to process the request and send a response.

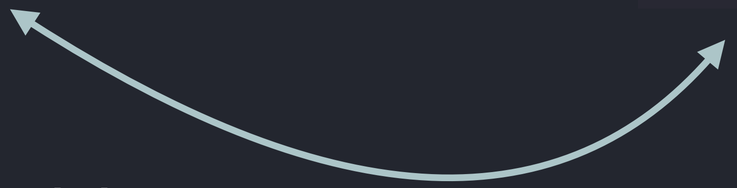
```
import express from "express";
const app = express();

// Define routes using app.METHOD(PATH, HANDLER)
app.get("/posts", (req, res) => {
    // return all posts
});

app.post("/posts", (req, res) => {
    const post = req.body;
    // Create a new post based on the data in req.body
});

app.put("/posts/:id", (req, res) => {
    const postId = req.params.id;
    const post = req.body;
    // Find and update the post with postId and data from req.body
});

app.delete("/posts/:id", (req, res) => {
    const postId = req.params.id;
    // Find and delete the post with postId
});

// ...and so on

app.listen(3000, () => {
    console.log("Server is running on port 3000");
});
```

Object Relational Mapping

ORM

id	name	mail	title	image
1	Peter Lind	petl@kea.dk	Senior Lecturer	https://share.cederdorff.com/images/pe…
2	Rasmus Cederdorff	race@dev.dk	Senior Lecturer	https://share.cederdorff.com/images/ra…
3	Lars Bogetoft	larb@eaaa.dk	Head of Education	https://kea.dk/slir/w200-c1x1/images/u…
4	Edith Terte	edan@kea.dk	Lecturer	https://media.licdn.com/dms/image/C4E0…
5	Frederikke Bender	fbe@kea.dk	Head of Education	https://kea.dk/slir/w200-c1x1/images/u…
6	Murat Kilic	mki@eaaa.dk	Senior Lecturer	https://www.eaaa.dk/media/llyavasj/mur…
7	Anne Kirketerp	anki@eaaa.dk	Head of Education	https://www.eaaa.dk/media/5buh1xeo/ann…

```
const user = {
    id: 6,
    name: "Murat Kilic",
    mail: "mki@eaaa.dk",
    title: "Senior Lecturer",
    image: "https://www.eaaa.dk/media/ll
};
```



From database to object

Object Relational Mapping

ORM

id	name	mail	title	image
1	Peter Lind	petl@kea.dk	Senior Lecturer	https://share.cederdorff.com/images/pe…
2	Rasmus Cederdorff	race@dev.dk	Senior Lecturer	https://share.cederdorff.com/images/ra…
3	Lars Bogetoft	larb@eaaa.dk	Head of Education	https://kea.dk/slir/w200-c1x1/images/u…
4	Edith Terte	edan@kea.dk	Lecturer	https://media.licdn.com/dms/image/C4E0…
5	Frederikke Bender	fbe@kea.dk	Head of Education	https://kea.dk/slir/w200-c1x1/images/u…
6	Murat Kilic	mki@eaaa.dk	Senior Lecturer	https://www.eaaa.dk/media/llyavasj/mur…
7	Anne Kirketerp	anki@eaaa.dk	Head of Education	https://www.eaaa.dk/media/5buh1xeo/ann…

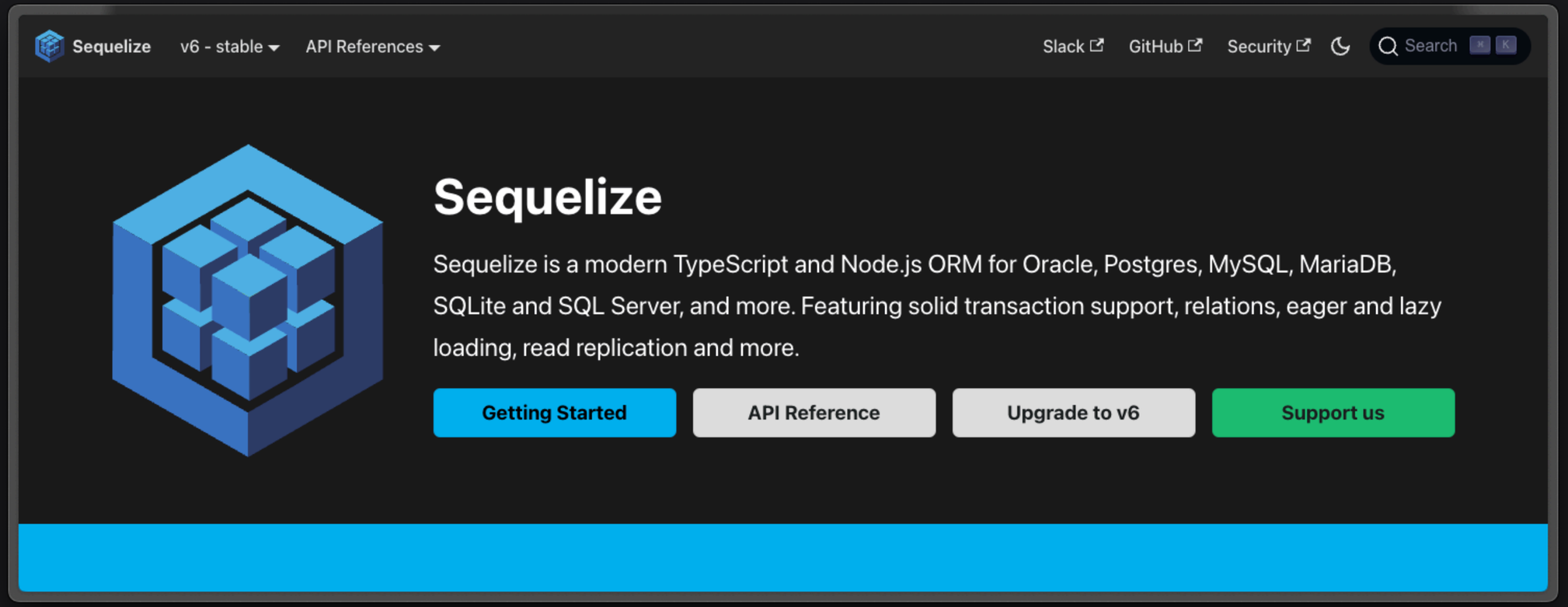
```
const users = [
    {
        id: 2,
        name: "Rasmus Cederdorff",
        mail: "race@dev.dk",
        title: "Senior Lecturer",
        image: "https://share.cederdorff.com/images/race.jpg"
    },
    {
        id: 3,
        name: "Lars Bogetoft",
        mail: "larb@eaaa.dk",
        title: "Head of Education",
        image: "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/larb.jpg"
    },
    {
        id: 4,
        name: "Edith Terte",
        mail: "edan@kea.dk",
        title: "Lecturer",
        image: "https://media.licdn.com/dms/image/C4E03AQE6nx7oUPqo_g/profile-displayphoto-shrink_800_800/
    },
    {
        id: 5,
        name: "Frederikke Bender",
        mail: "fbe@kea.dk",
        title: "Head of Education",
        image: "https://kea.dk/slir/w200-c1x1/images/user-profile/chefer/fbe.jpg"
    },
    {
        id: 6,
        name: "Murat Kilic",
        mail: "mki@eaaa.dk",
        title: "Senior Lecturer",
        image: "https://www.eaaa.dk/media/llyavasj/murat-kilic.jpg?width=800&height=450&rnd=133401946552600
    },
    {
        id: 7,
        name: "Anne Andersen",
        mail: "anki@mail.dk",
        title: "Head of Education",
        image: "https://www.eaaa.dk/media/5buh1xeo/anne-kirketerp.jpg?width=800&height=450&rnd=133403878321
    }
];
```

From database to array of objects

Object Relational Mapping

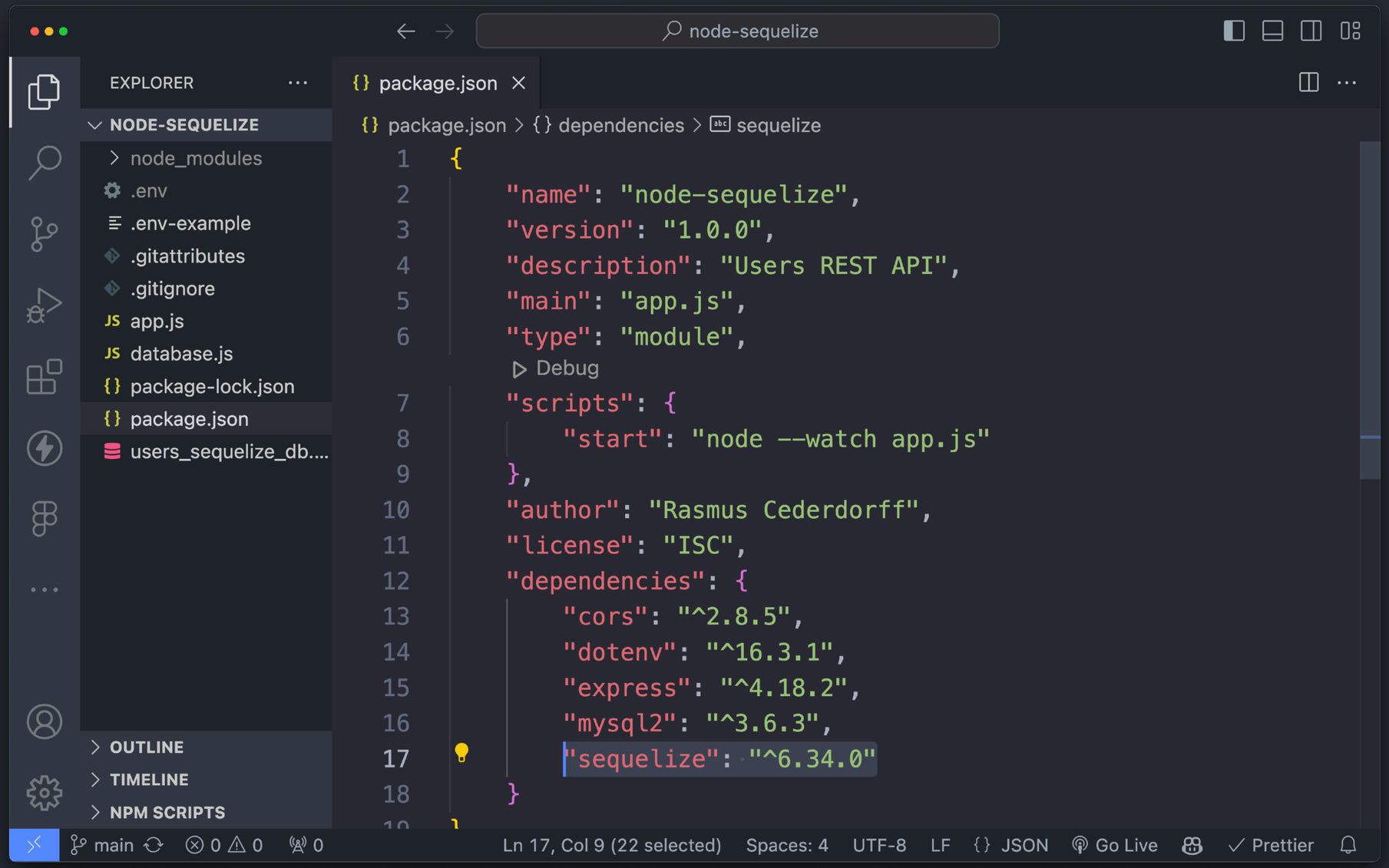
ORM

- **Simplifies Database Interaction:** Allows using objects in code to interact with a relational database.
- **Maps Objects to Tables**: Links object-oriented language structures to relational database tables.
- **Abstraction Layer**: Abstracts database operations, reducing manual SQL coding.
- **Automates Queries**: Generates and executes SQL queries based on object interactions.
- **Saves Development Time**: Reduces the need for low-level database code, streamlining development.


Sequelize
v6 - stable
API References
Slack
GitHub
Security
Search
Sequelize
Sequelize is a modern TypeScript and Node.js ORM for Oracle, Postgres, MySQL, MariaDB, SQLite and SQL Server, and more. Featuring solid transaction support, relations, eager and lazy loading, read replication and more.
Getting Started
API Reference
Upgrade to v6
Support us

... npm package

Sequelize is a dependency (node module)



Define models to handle the DB interaction

```
// ========== Define Models ========== //
// Define user model
const User = sequelize.define("user", {
    // User model attributes
    name: {
        type: DataTypes.STRING,
        allowNull: false // Name is required
    },
    title: {
        type: DataTypes.STRING
    },
    mail: {
        type: DataTypes.STRING,
        allowNull: false // Email is required
    },
    image: {
        type: DataTypes.TEXT // URL to image
    }
});
```

```
// Sample users
// Sample user 1
const user1 = await User.create({
    name: "Rasmus Cederdorff",
    title: "Senior Lecturer",
    mail: "race@eaaa.dk",
    image: "https://share.cederdorff.com/images/race.jpg"
});

// Sample user 2
const user2 = await User.create({
    name: "Anne Kirketerp",
    title: "Head of Department",
    mail: "anki@eaaa.dk",
    image: "https://www.eaaa.dk/media/5buh1xeo/anne-kirketerp.jpg?wi
});

// Sample user 3
const user3 = await User.create({
    name: "Murat Kilic",
    title: "Senior Lecturer",
    mail: "mki@eaaa.dk",
    image: "https://www.eaaa.dk/media/llyavasj/murat-kilic.jpg?width=
});
```

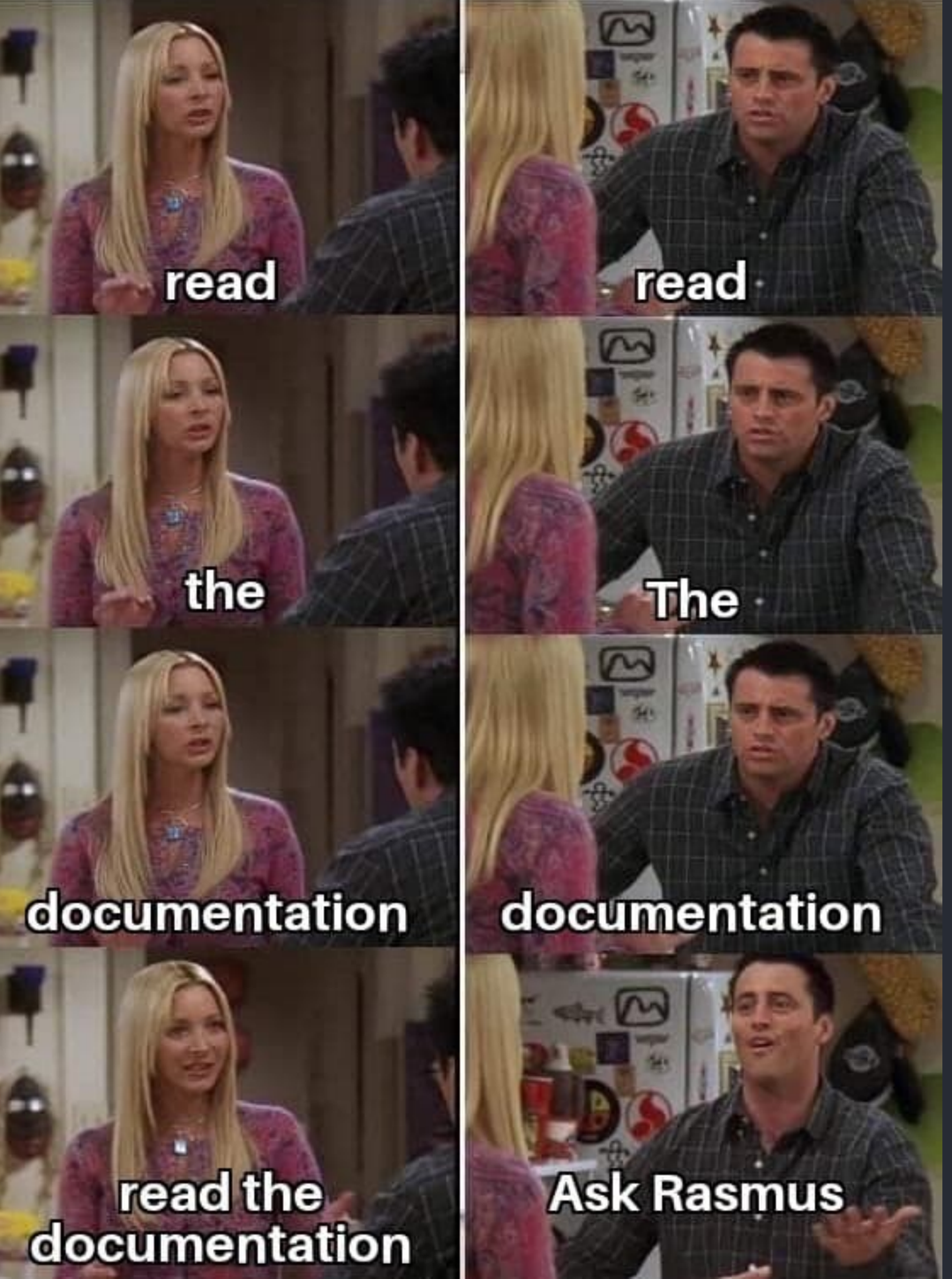
Creates users table with entities

```
// ========== Define Models ========== //
// Define user model
const User = sequelize.define("user", {
    // User model attributes
    name: {
        type: DataTypes.STRING,
        allowNull: false // Name is required
    },
    title: {
        type: DataTypes.STRING
    },
    mail: {
        type: DataTypes.STRING,
        allowNull: false // Email is required
    },
    image: {
        type: DataTypes.TEXT // URL to image
    }
});
```

id	name	title	mail	image	createdAt	updatedAt
1	Rasmus Ced…	Senior Lect…	race@eaaa.dk	https://share.cederdo…	2023-11-12 19:0…	2023-11-12 19:0…
2	Anne Kirke…	Head of Dep…	anki@eaaa.dk	https://www.eaaa.dk/m…	2023-11-12 19:0…	2023-11-12 19:0…
3	Murat Kilic	Senior Lect…	mki@eaaa.dk	https://www.eaaa.dk/m…	2023-11-12 19:0…	2023-11-12 19:0…

Methods (SQL Queries)

```
const users = await User.findAll(); // SELECT * FROM users;

const users = await User.findByPk(id); // SELECT * FROM users WHERE id = id;

const newUser = await User.create(user); // INSERT INTO users (name, title, mail, image) VAL

const [result] = await User.update(user, { where: { id: id } }); // UPDATE users SET name =

const result = await User.destroy({ where: { id: id } }); // DELETE FROM users WHERE id = id
```

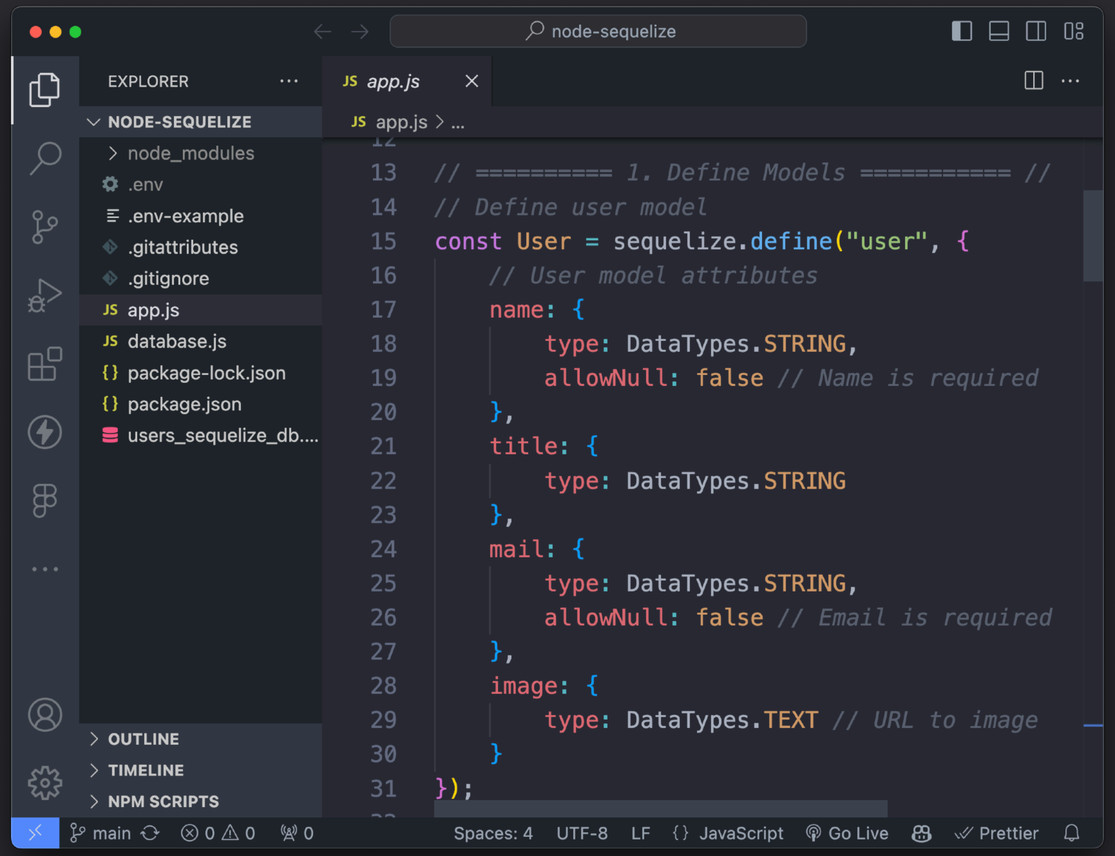


Useful Methods (see the docs)

- Core Concepts of Sequelize: https://sequelize.org/docs/v6/category/core-concepts/
- Model Basics: https://sequelize.org/docs/v6/core-concepts/model-basics/
- Model Querying - Basics: https://sequelize.org/docs/v6/core-concepts/model-querying-basics/
- Model Querying - Finders: https://sequelize.org/docs/v6/core-concepts/model-querying-finders/
- Associations: https://sequelize.org/docs/v6/core-concepts/assocs/

Øvelse

Node.js & Sequelize



Code Every Day