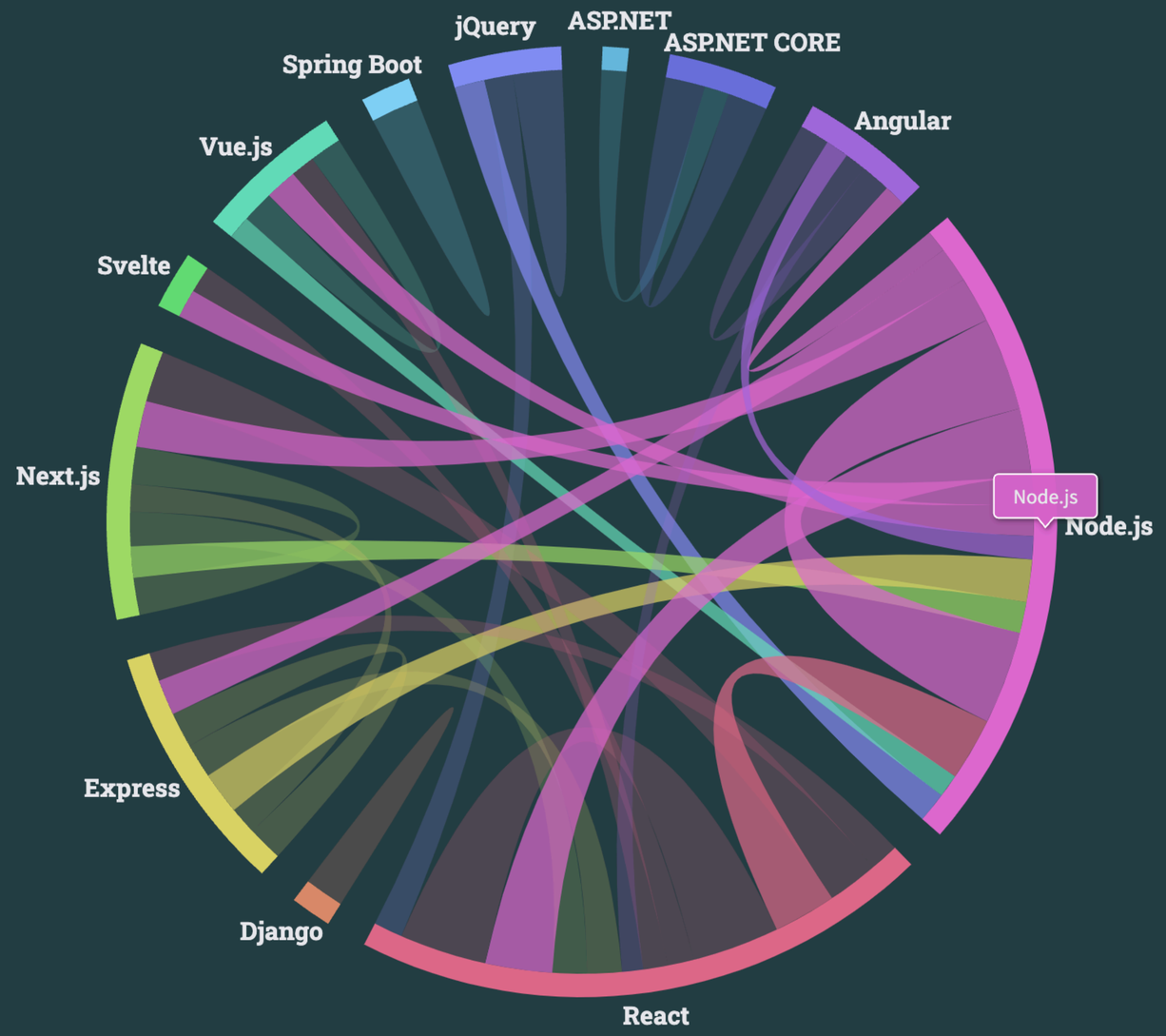
Introduction to

Web frameworks and technologies

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

Web frameworks and technologies



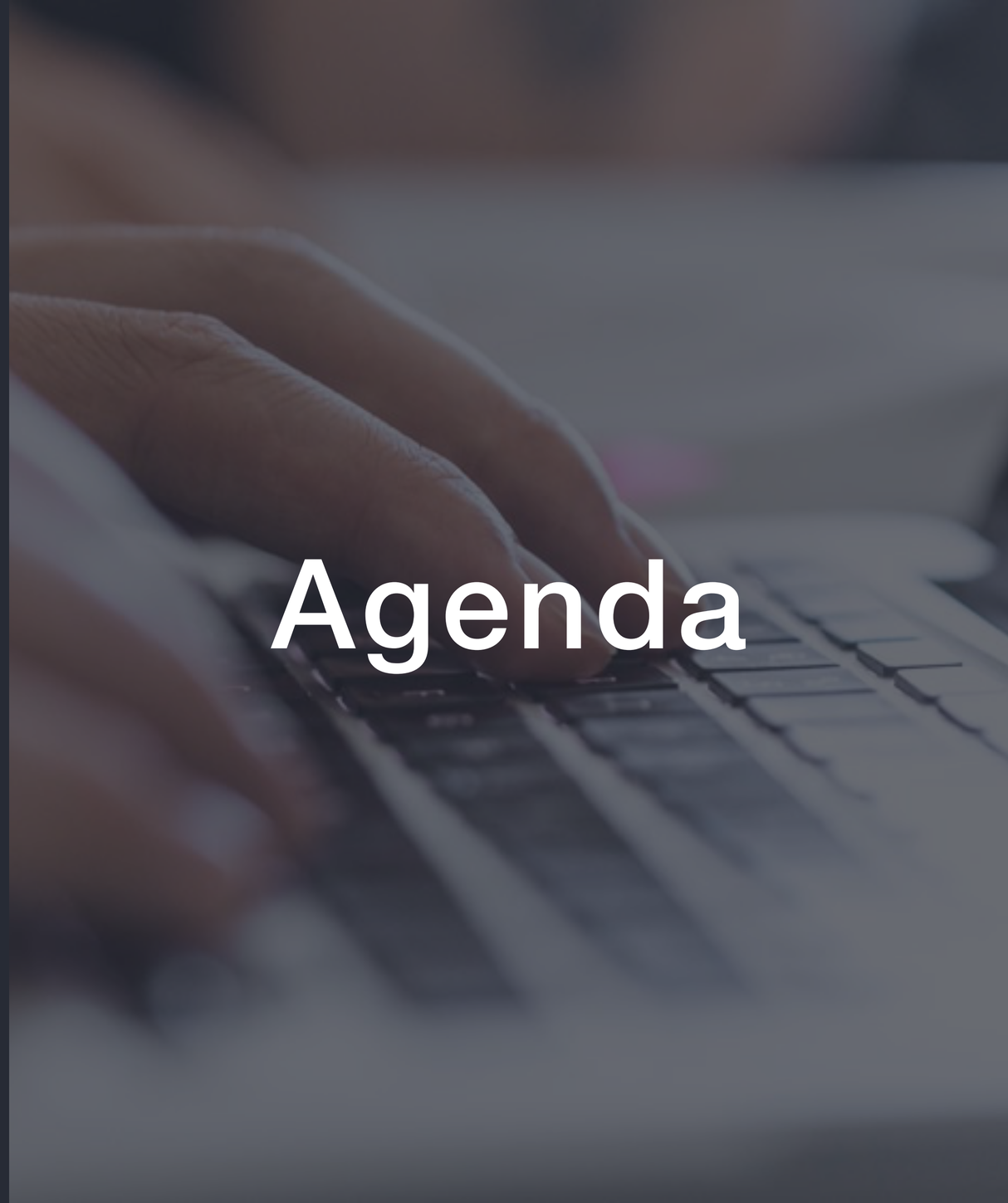
https://survey.stackoverflow.co/2023/#section-worked-with-vs-want-to-work-with-web-frameworks-and-technologies

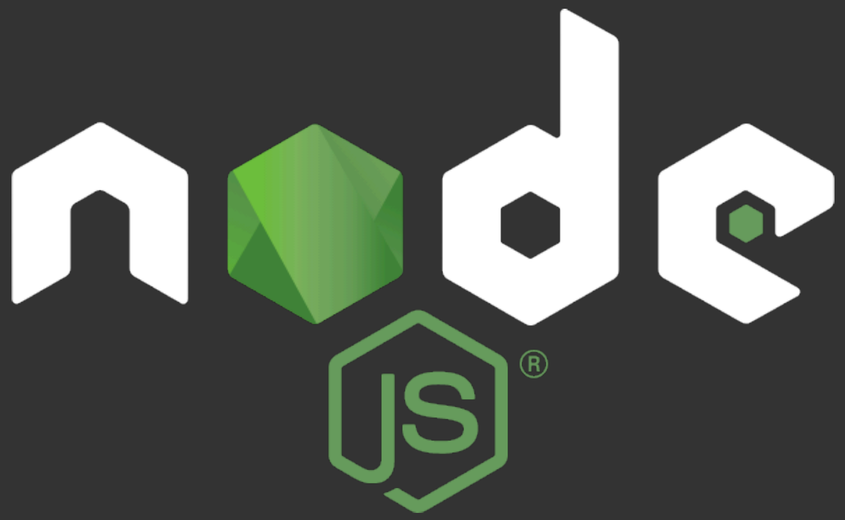
Formål

- At skabe en forståelse for hvad Node.js er og kan, samt introducere de koncepter og tools, der gør sig gældende for backendudvikling med JavaScript (Node.js).
- Praktisk tilgang: Vi skal arbejde med opgaver kombineret med mindre teorioplæg.

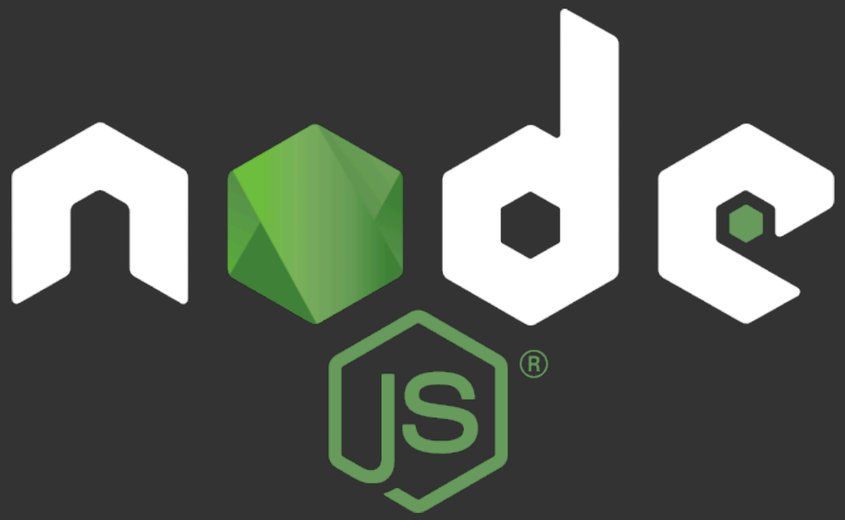
Agenda

1. What’s Node.js & Why Do We Need It?
2. Node.js Runtime Environment & Key Features
3. Node.js HTTP Module & Postman
4. Node.js File System
5. Wrap up (Libraries & Hand In)



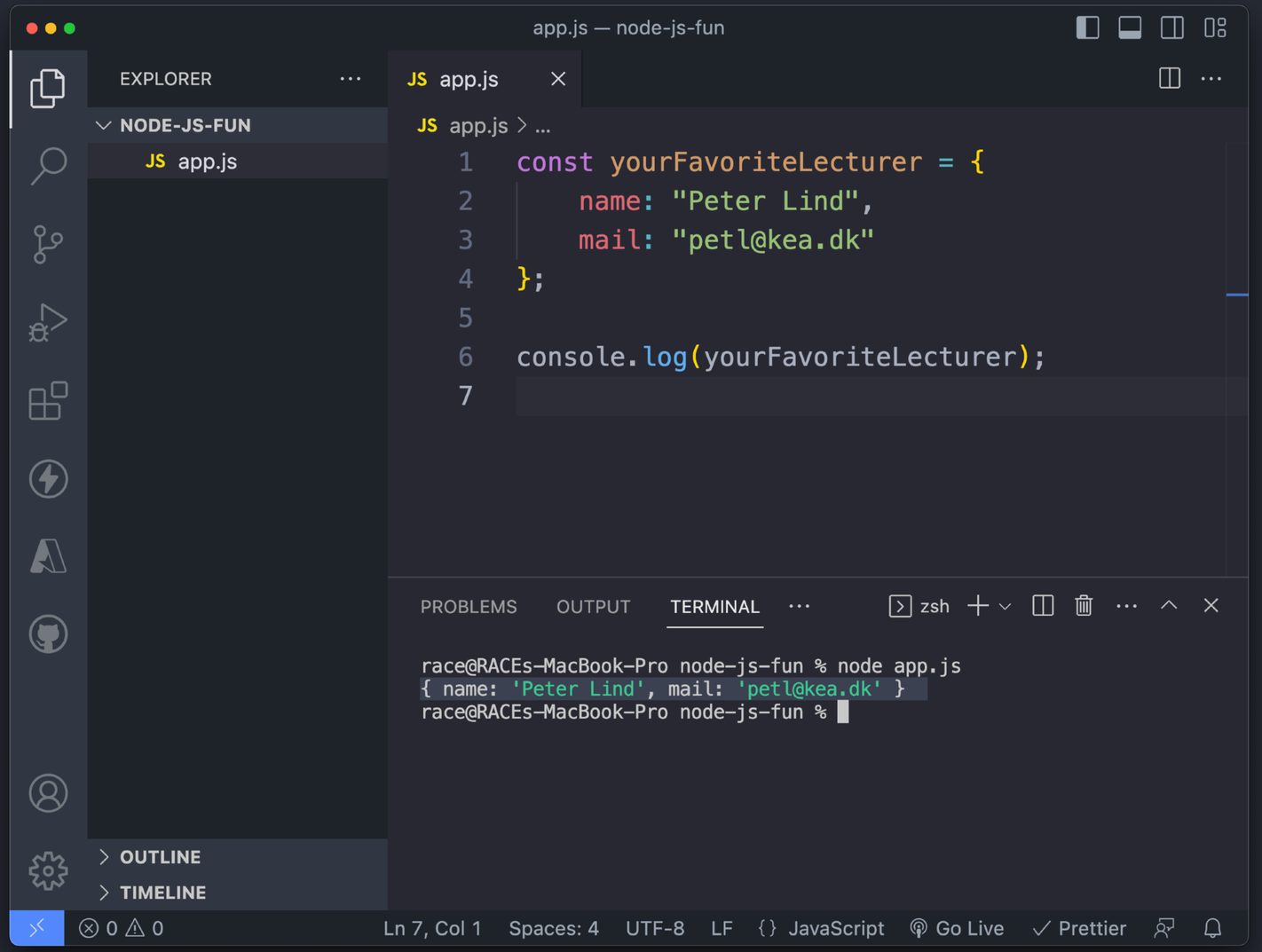


- Server-side JavaScript runtime environment.
- Executes JavaScript outside of the browser.



- 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.

Run JavaScript, not just in browsers 🎉

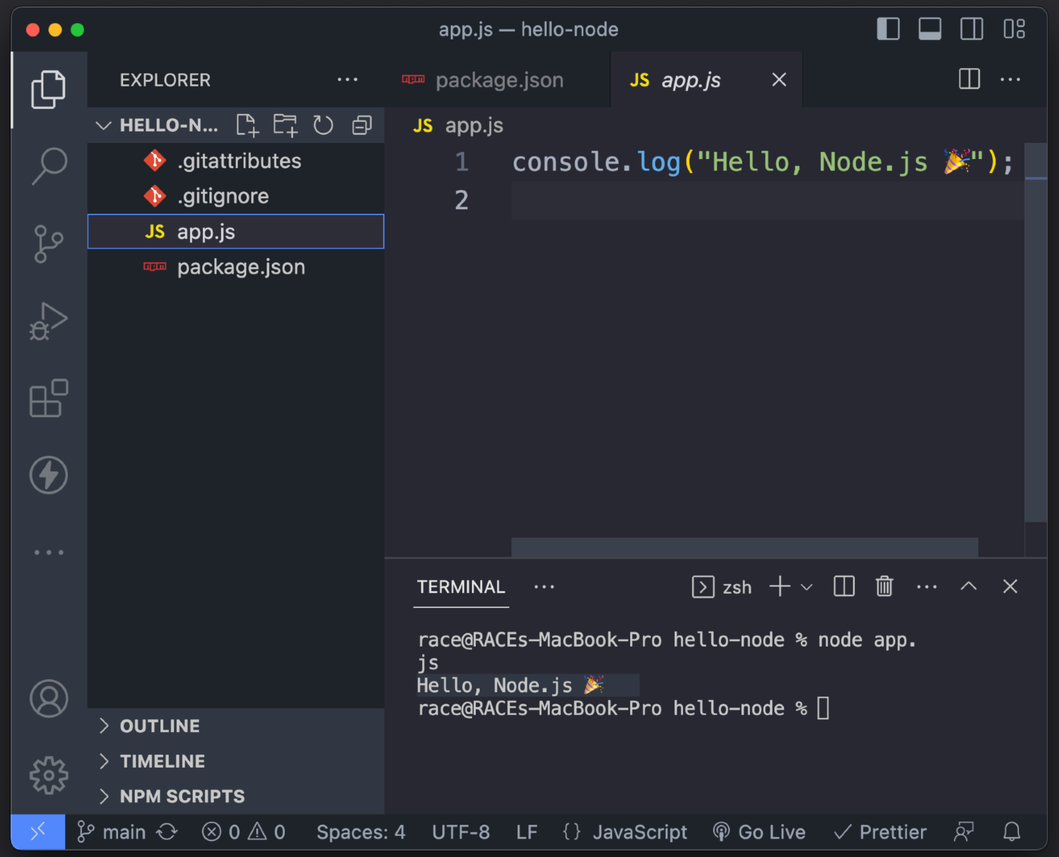


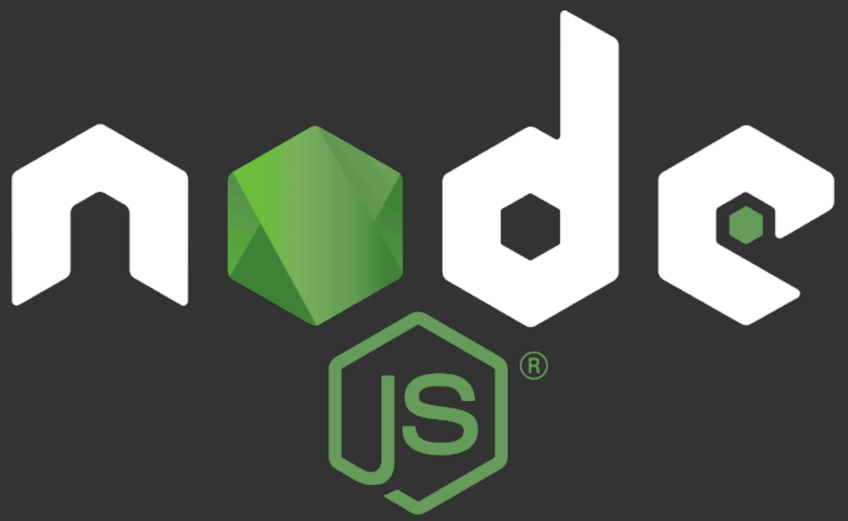
- **JavaScript Beyond Browser**
  Instead of being limited to doing things only on the internet, Node.js lets you use JavaScript to build programs and applications on your computer or server. You get to use the same language you know from web pages.
- **Fast and Efficient**
  Node.js is built to be really 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

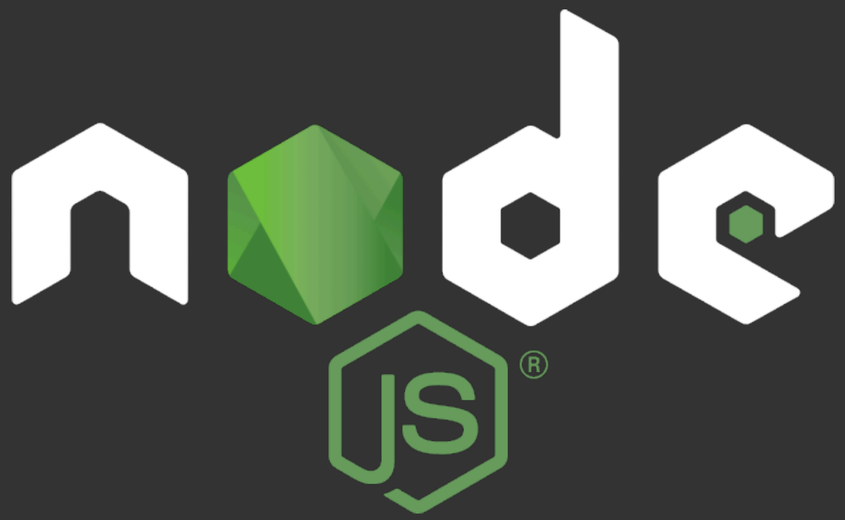
Øvelse

Hello Node.js





Why do we need it?

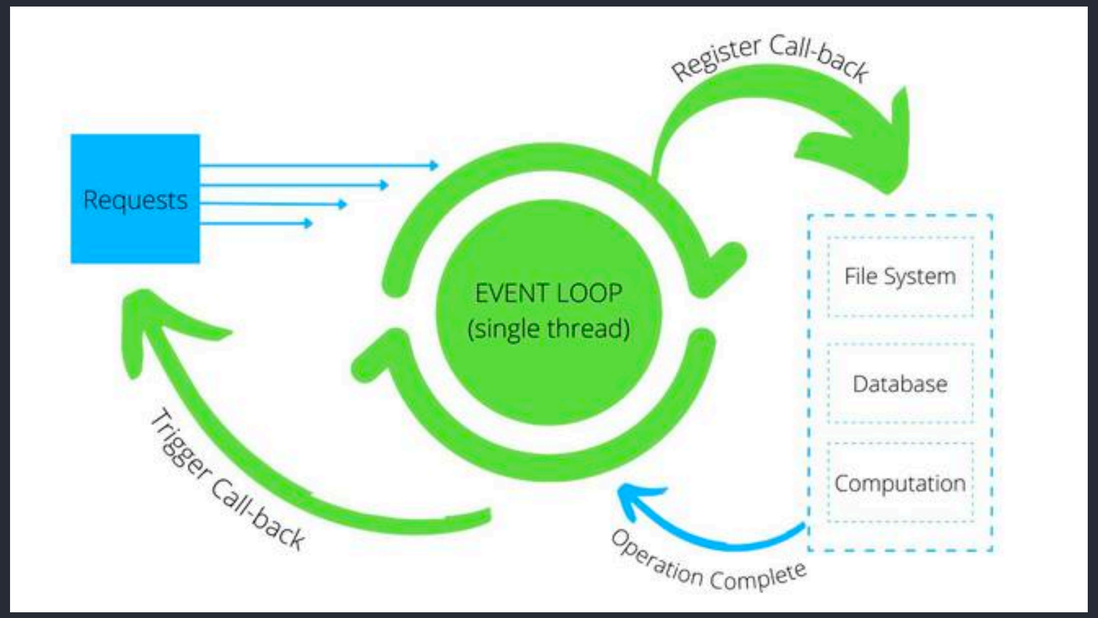


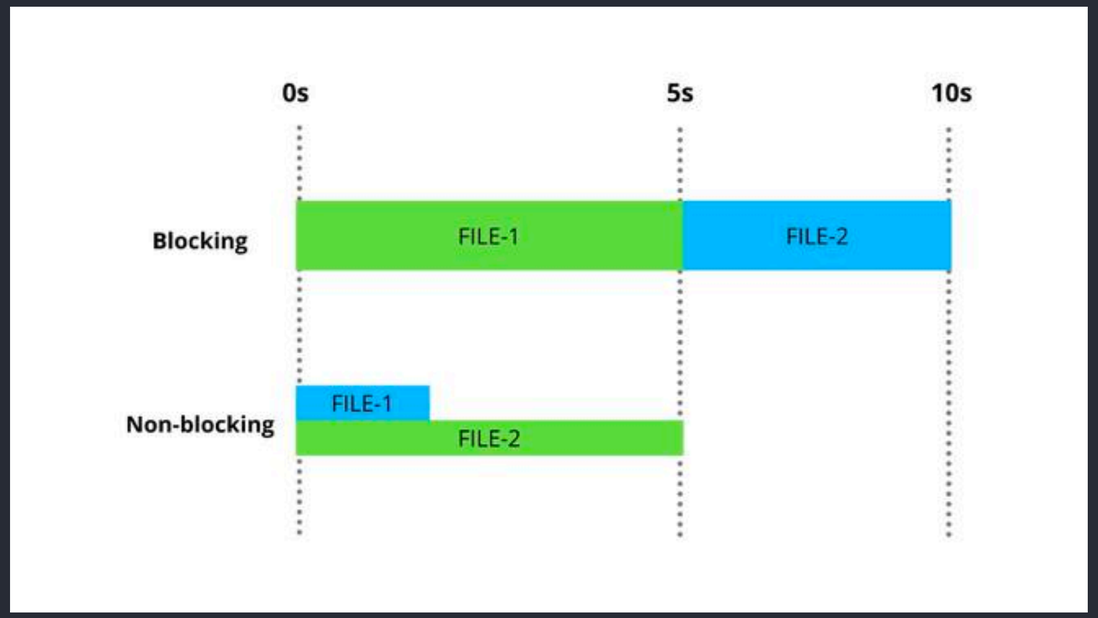
Why do we need it?

- In pairs, discuss and investigate
- Look at code and precious Node projects
- Ask Google and ChatGPT

Runtime Environment

- The Node.js runtime environment is an open-source, cross-platform JavaScript runtime built on Google Chrome's V8 JavaScript engine.
- It allows developers to execute JavaScript code on the server-side, outside of the browser, enabling them to build scalable and high-performance applications.
- Node.js provides a range of features and capabilities that are particularly well-suited for building real-time, networked, and event-driven applications.

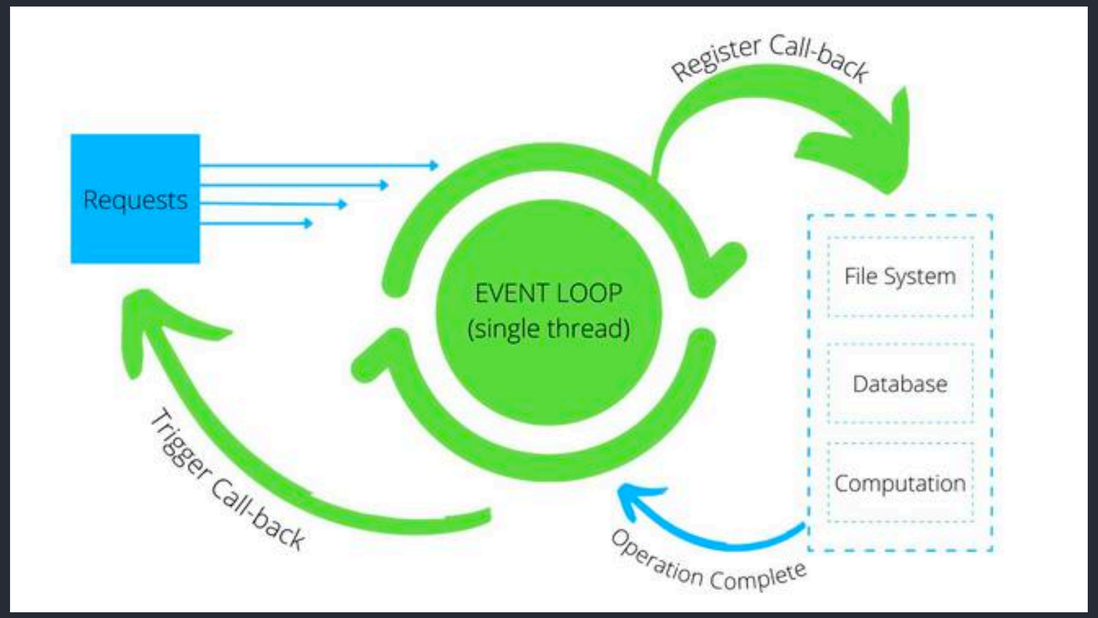


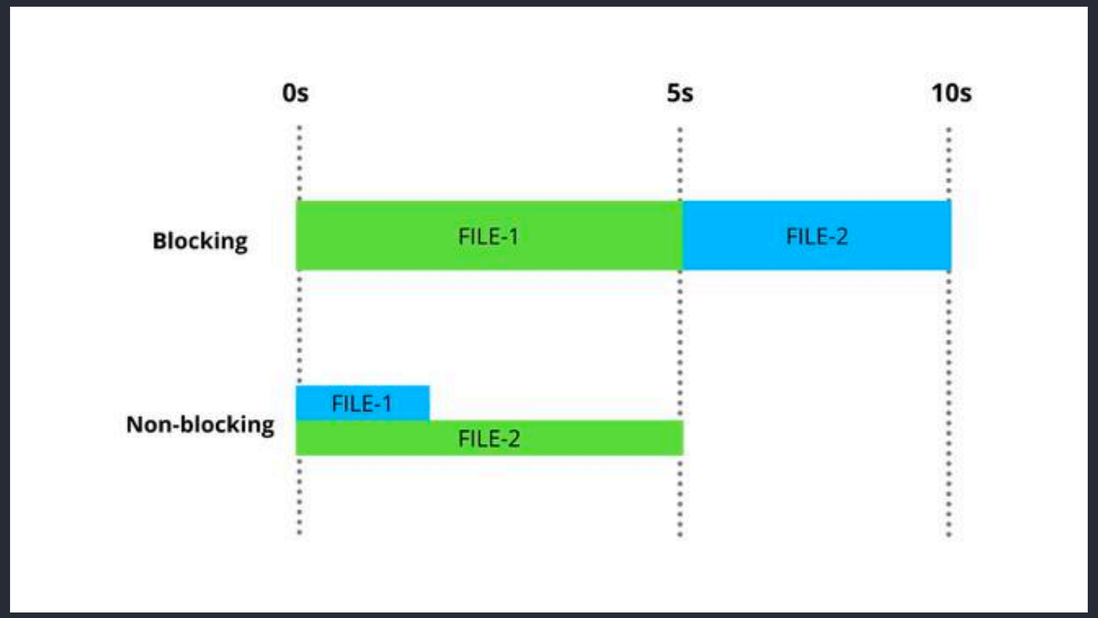


https://www.geeksforgeeks.org/how-node-js-overcome-the-problem-of-blocking-of-i-o-operations/

Runtime Environment

- **Non-blocking I/O**
  Node.js operates using an event-driven, non-blocking I/O model, which means that it can handle multiple concurrent connections without requiring a separate thread for each connection. This design choice enables efficient handling of I/O-bound tasks, making it suitable for applications that involve a lot of network communication.

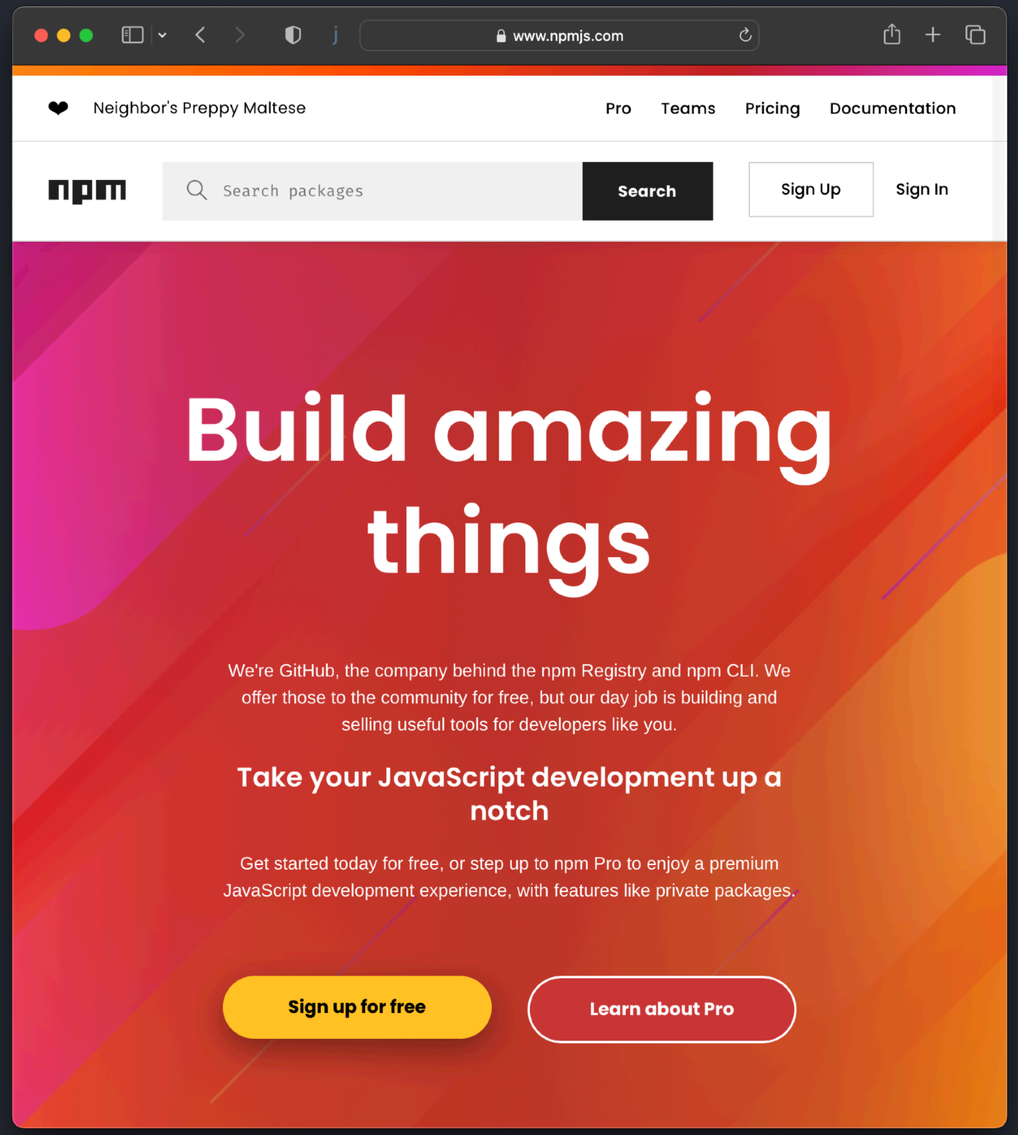




https://www.geeksforgeeks.org/how-node-js-overcome-the-problem-of-blocking-of-i-o-operations/

Runtime Environment

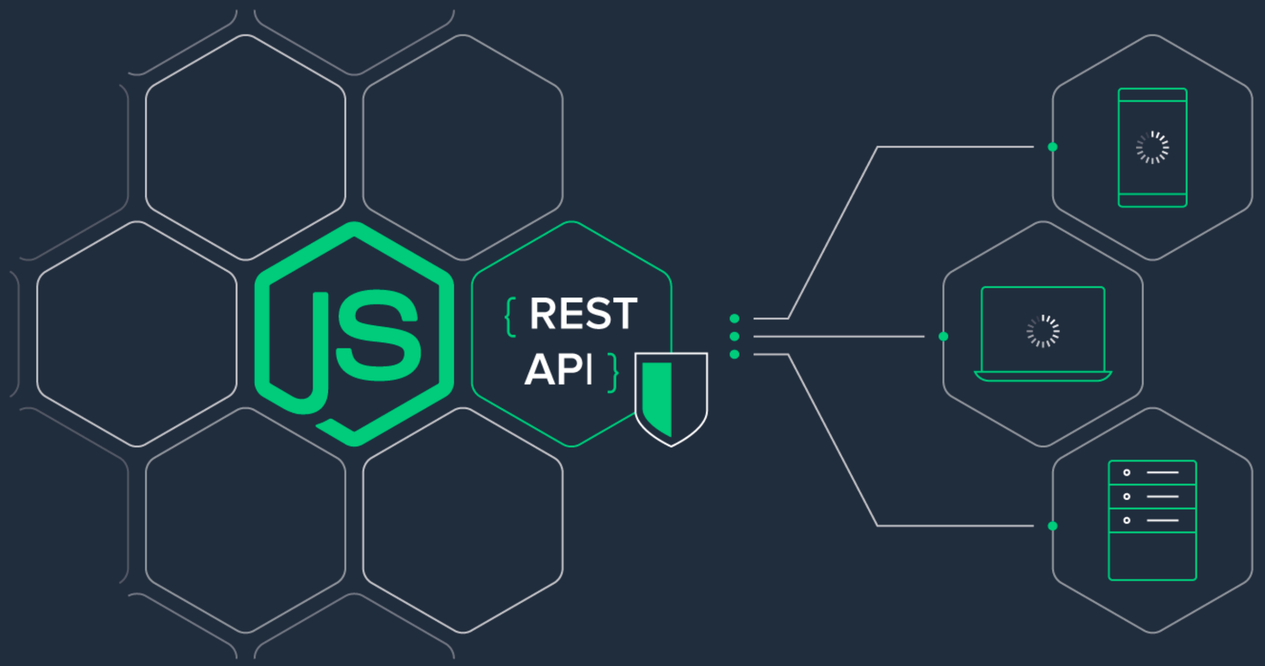
- **Asynchronous Programming**
  Node.js heavily emphasizes asynchronous programming patterns. This allows developers to write code that doesn't block the execution of other tasks while waiting for certain operations to complete. This approach enhances the application's responsiveness and throughput.



https://www.npmjs.com/

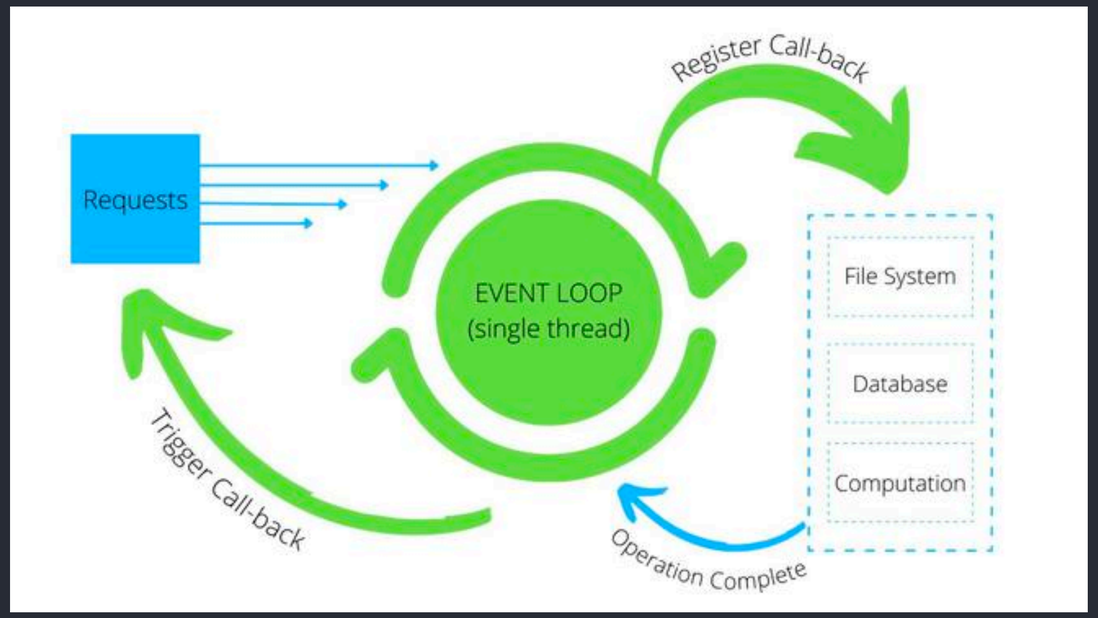
Runtime Environment

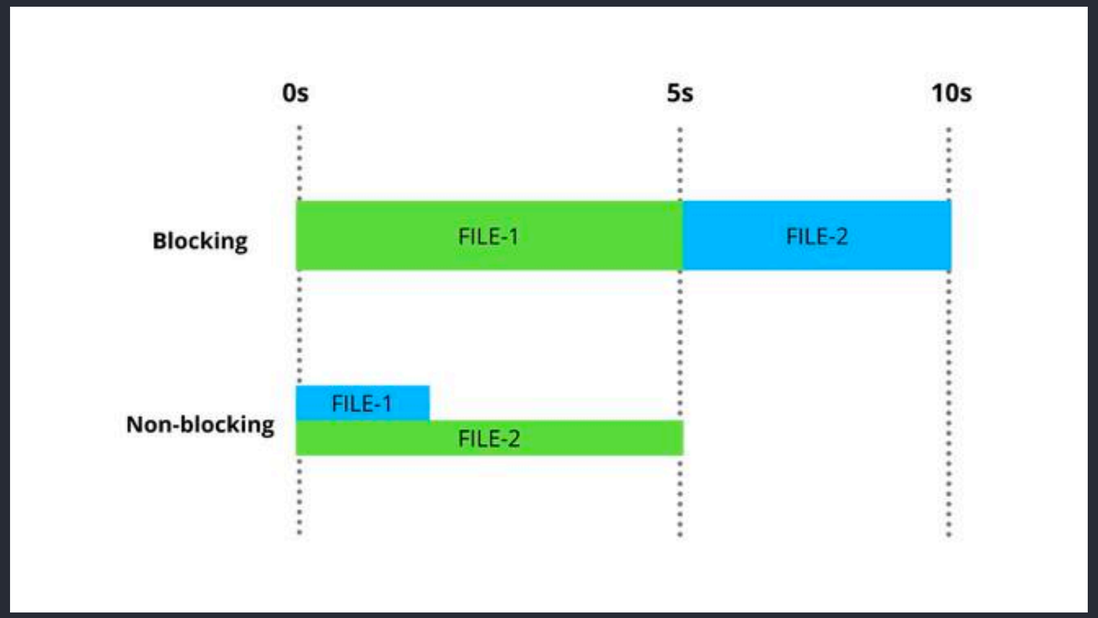
- **NPM (Node Package Manager)**
  NPM is a powerful package manager that comes bundled with Node.js. It provides access to a vast ecosystem of open-source libraries and modules that can be easily integrated into Node.js applications, saving developers time and effort by avoiding the need to reinvent the wheel.



Runtime Environment

- **Server-side Capabilities**
  Node.js can function as a web server, handling incoming HTTP requests and serving responses. Developers can build web applications, APIs, and microservices using the same language for both client and server, which can streamline development and maintenance.

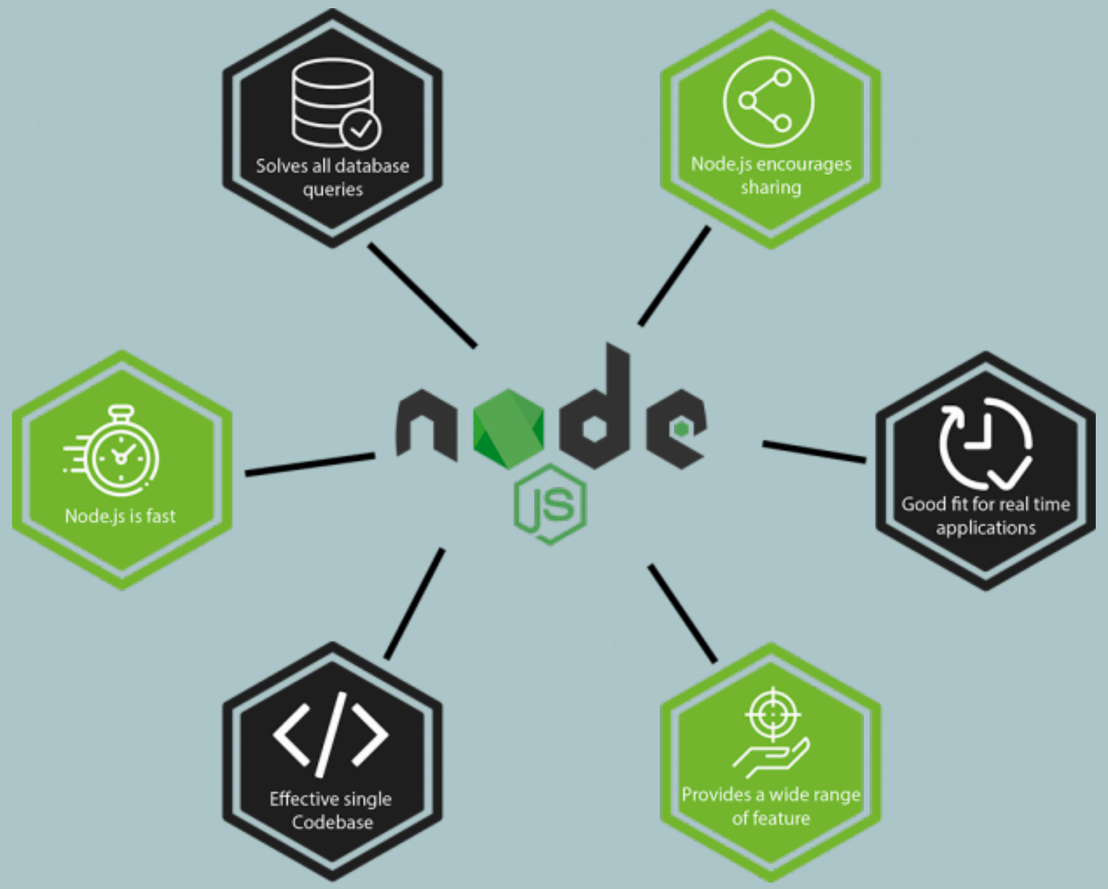




https://www.geeksforgeeks.org/how-node-js-overcome-the-problem-of-blocking-of-i-o-operations/

Runtime Environment

- **Scalability**
  Due to its non-blocking architecture, Node.js is well-suited for building scalable applications that can handle a large number of concurrent connections. This makes it a popular choice for real-time applications like chat applications, online gaming, and streaming services.



Runtime Environment

- **Community and Ecosystem**
  Node.js has a large and active community that continuously contributes to its growth. The ecosystem includes various tools, frameworks, and libraries that extend its capabilities and make development more efficient.

Runtime Environment

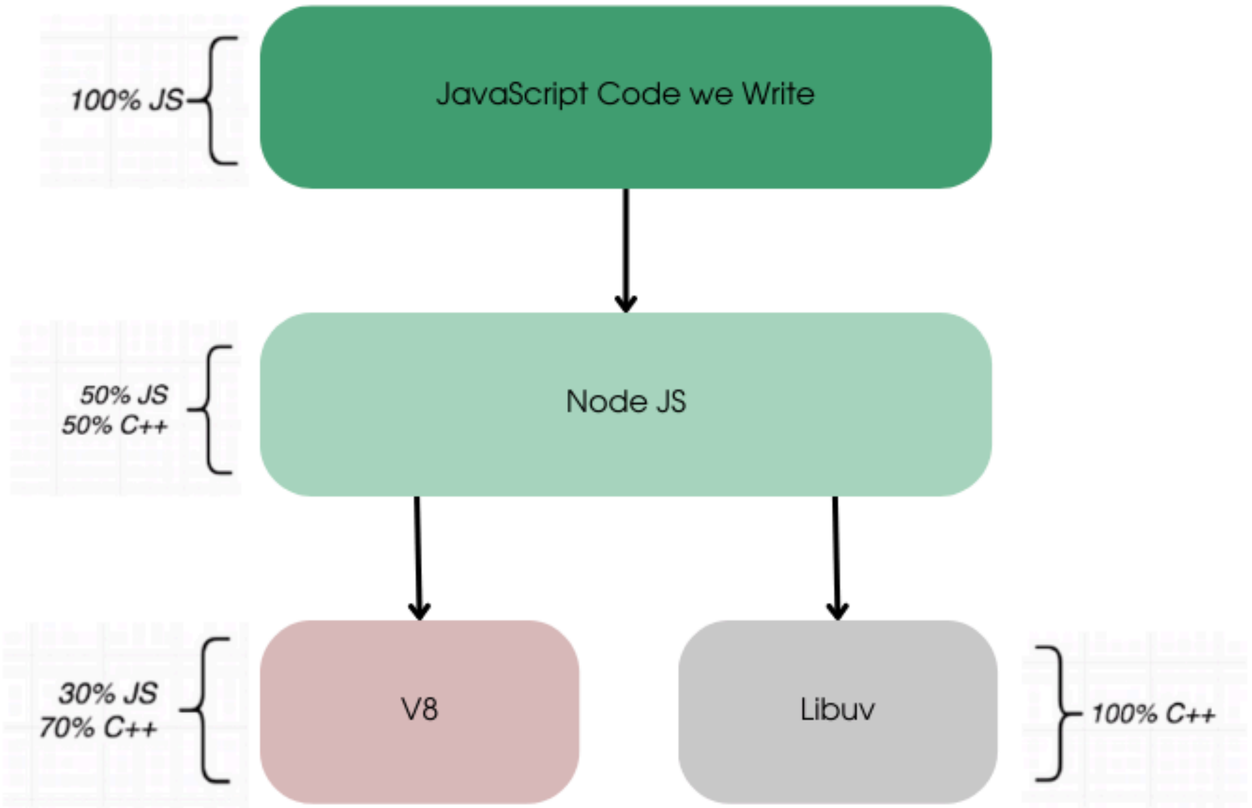
- In summary, Node.js provides a runtime environment that allows developers to use JavaScript beyond the browser, enabling them to create high-performance, scalable, and real-time applications for various use cases.

... we can use our existing **JavaScript expertise** to develop backend applications, servers, REST APIs, etc.

Runtime
Environment

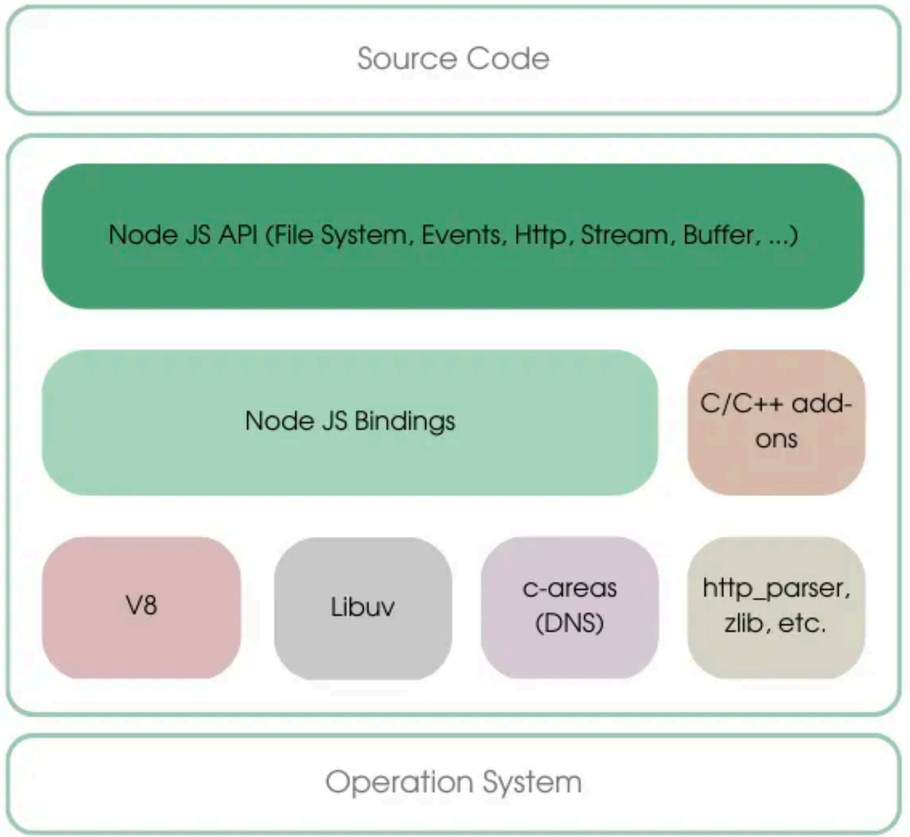
= Runtime Environment + JavaScript Library

Node.js Architecture



https://levelup.gitconnected.com/nodejs-runtime-environment-libuv-library-event-loop-thread-pool-5f5ecadc0318

Node.js Architecture



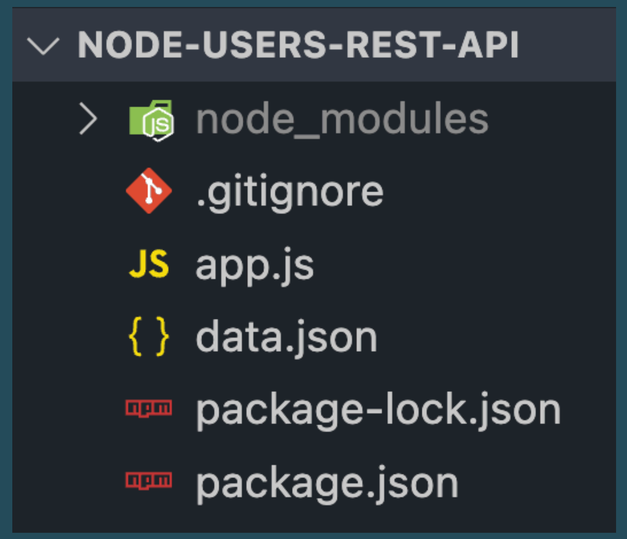
https://levelup.gitconnected.com/nodejs-runtime-environment-libuv-library-event-loop-thread-pool-5f5ecadc0318

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.

My Project



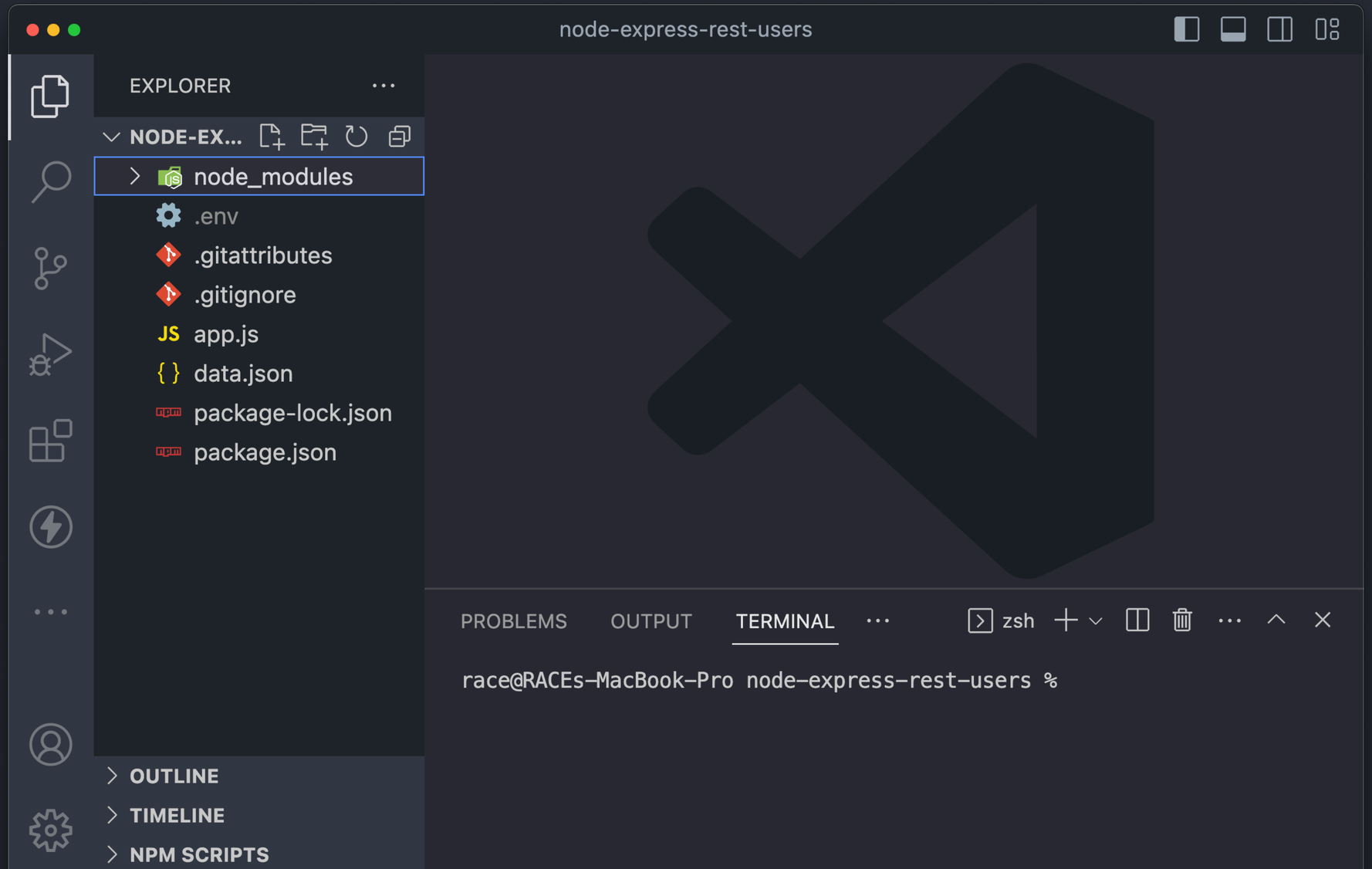
npm

Express JS

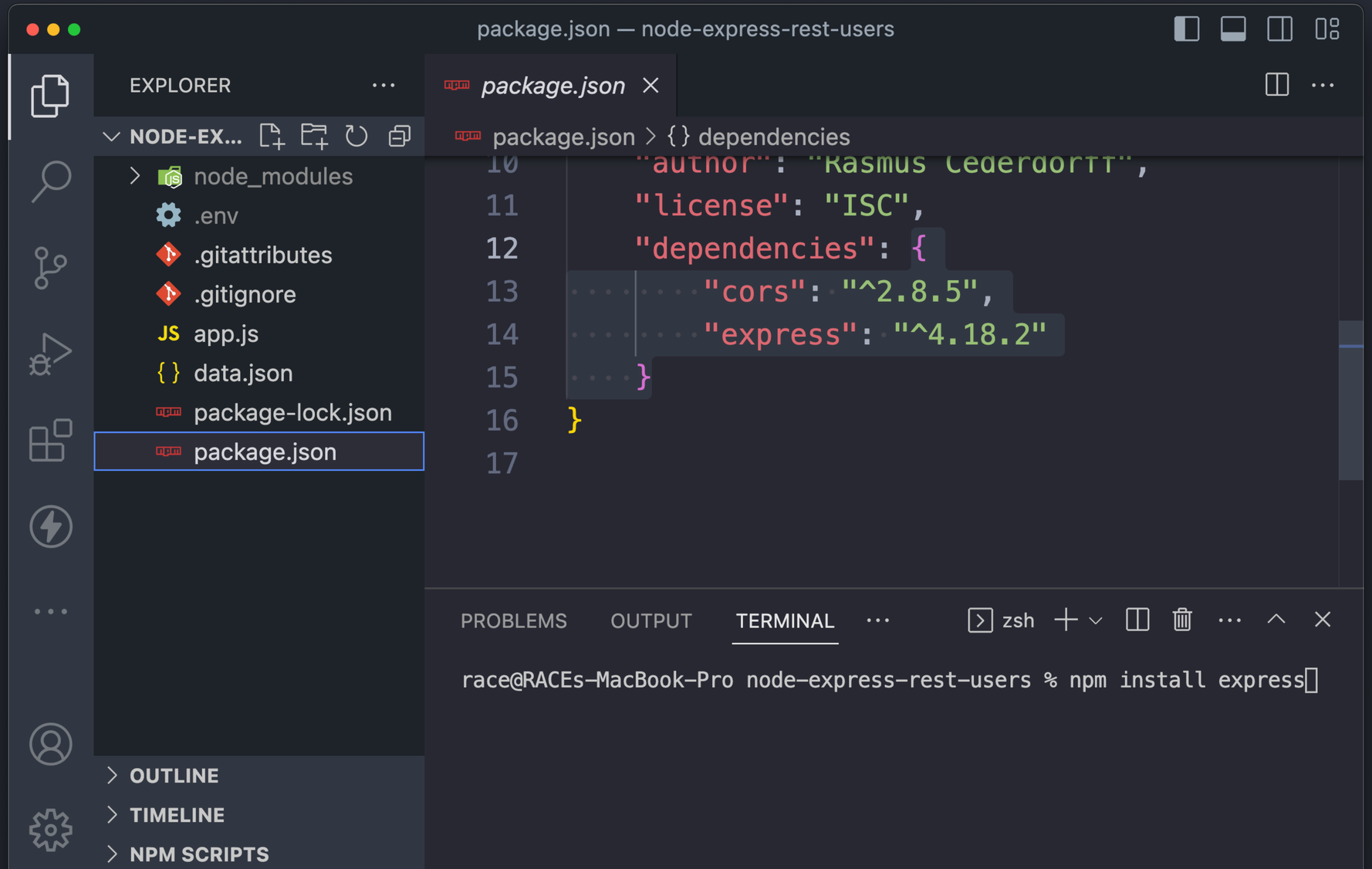
CORS

And many other
JavaScript libraries

node_modules



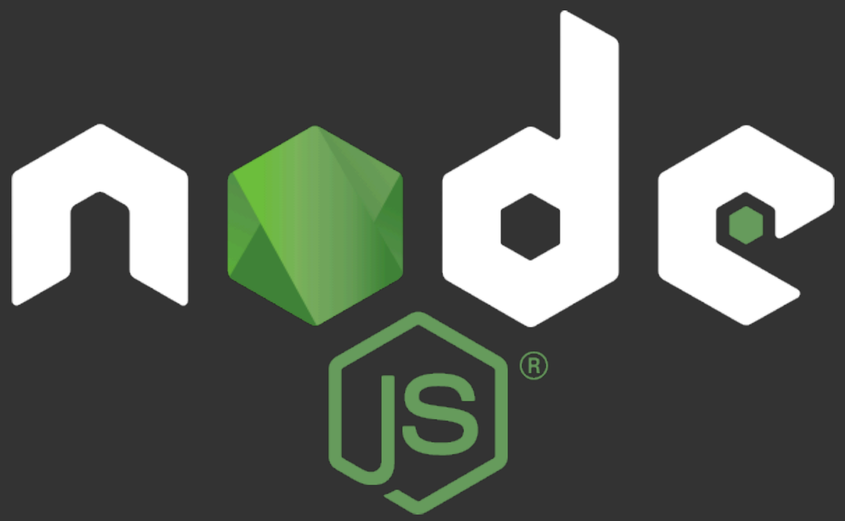
npm install {package-name}



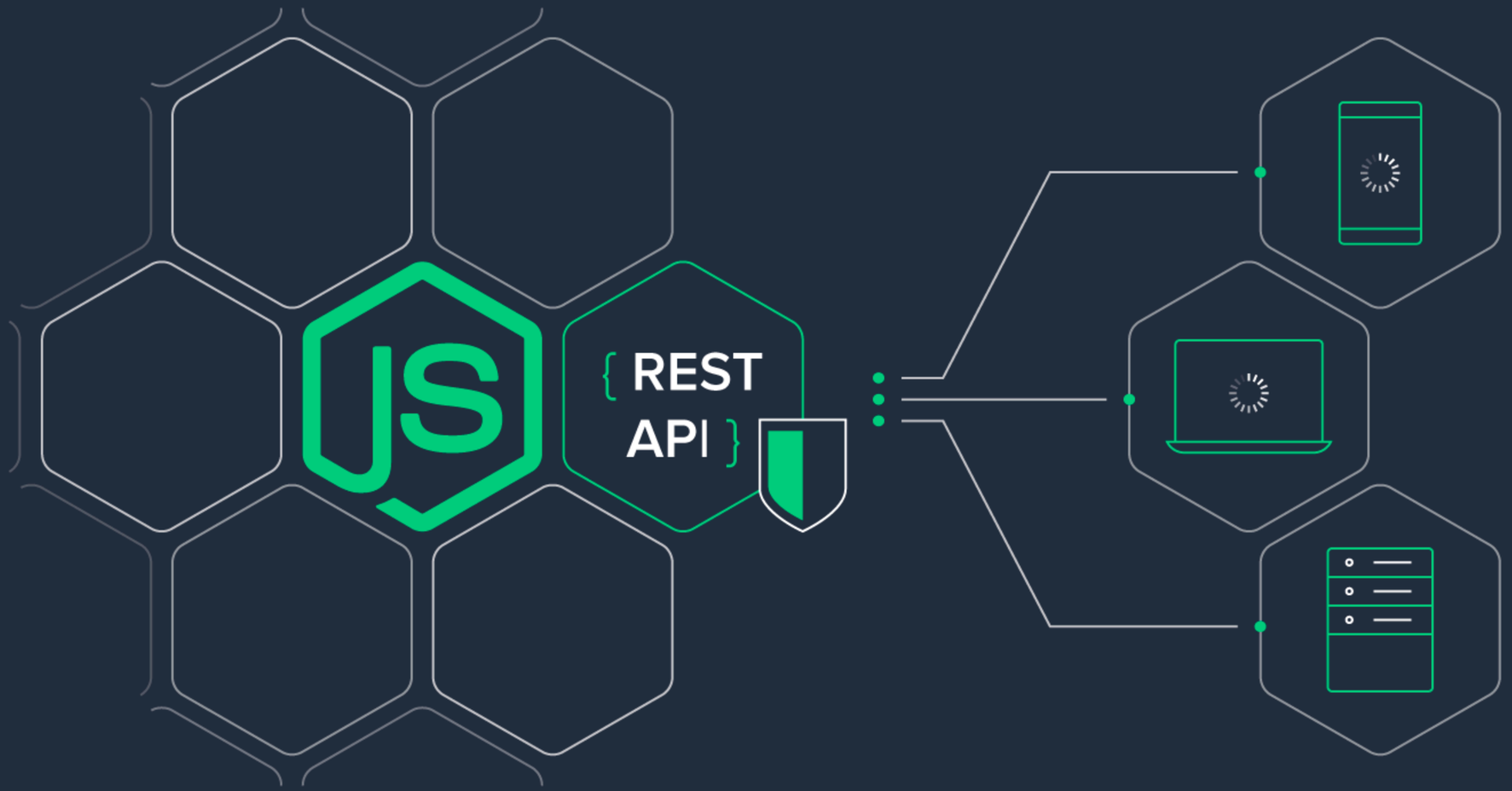
npm consists of

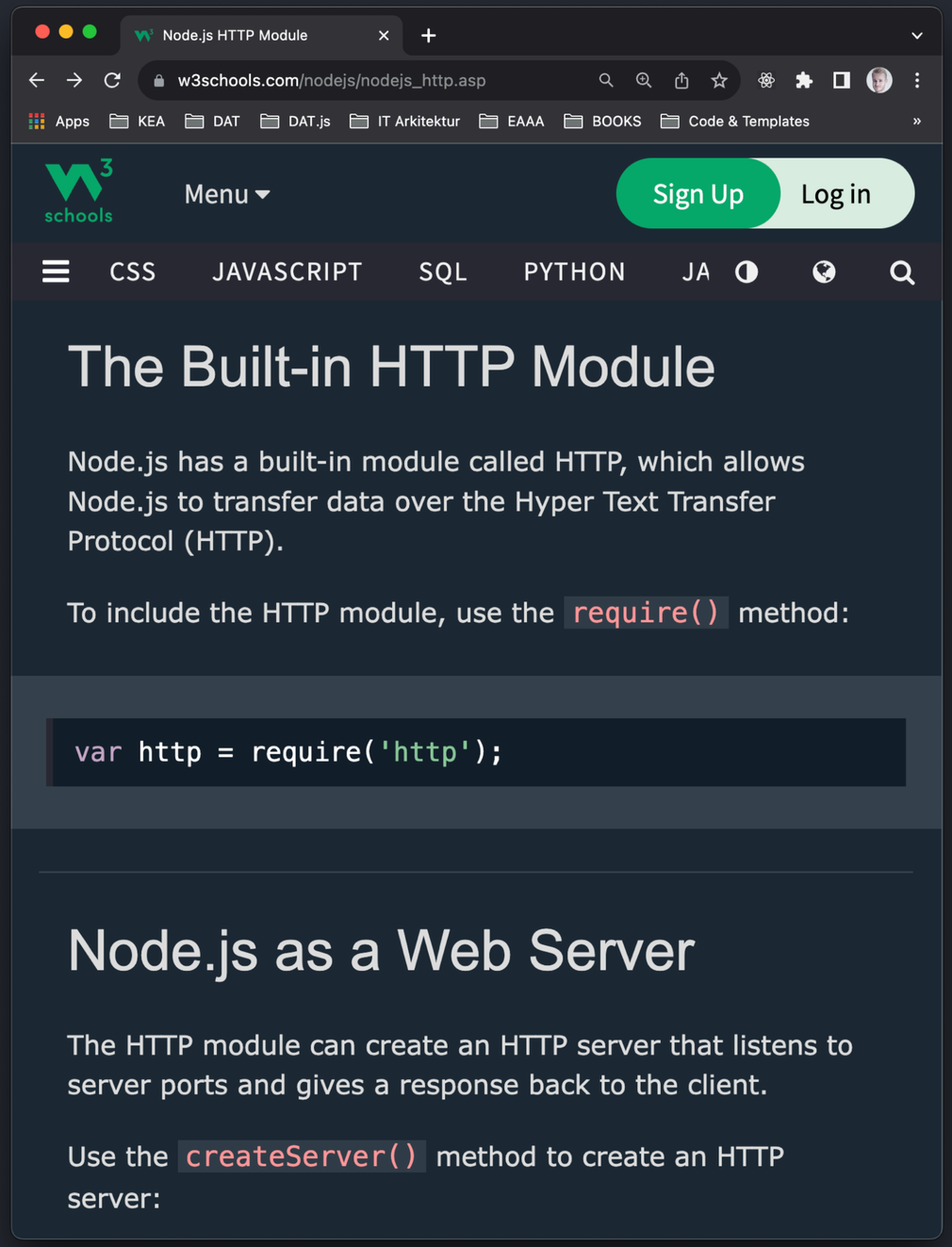
1. Website to discover packages
2. Command line interface (cli)
3. A registry - database with JS software/ libraries/ node modules

https://docs.npmjs.com/about-npm/



Node.js HTTP Module


JS
{ REST
API }

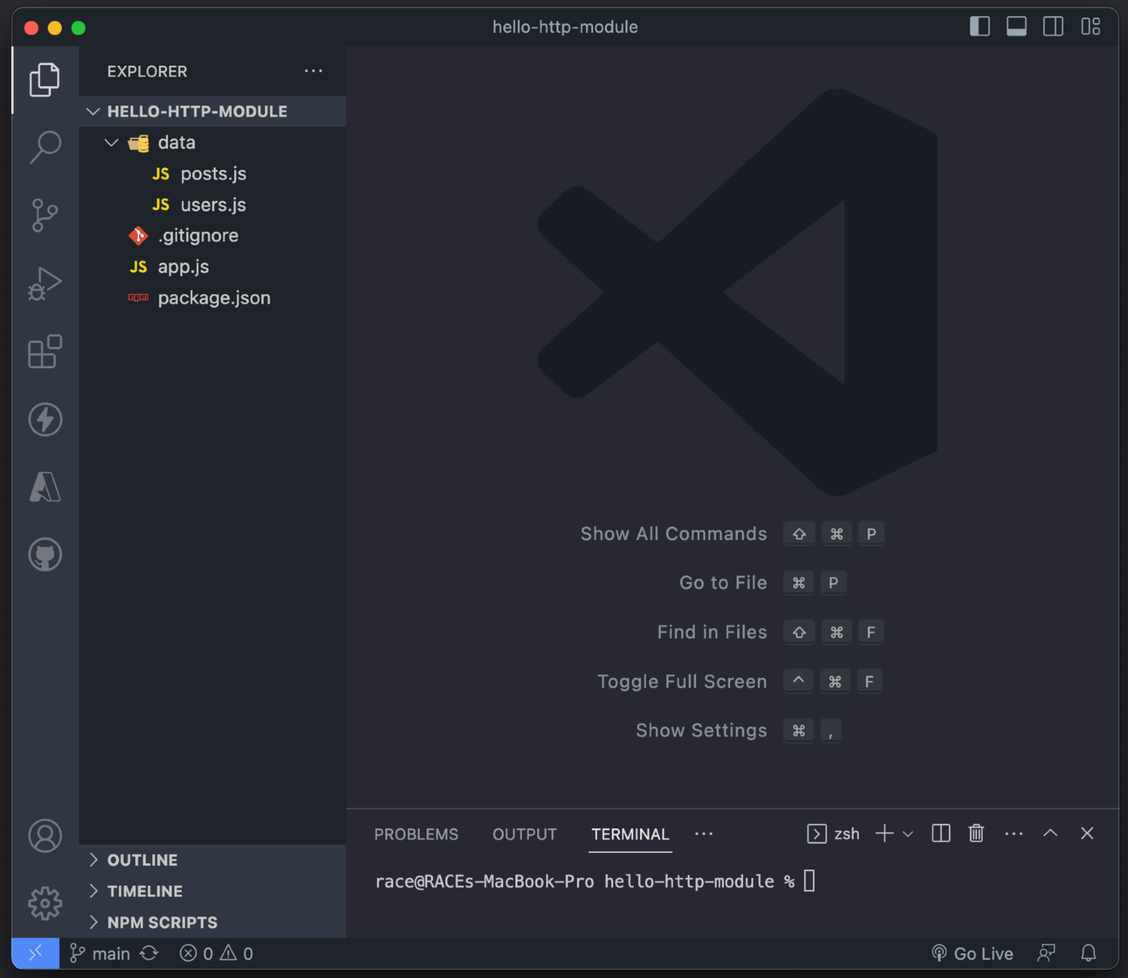


https://www.w3schools.com/nodejs/nodejs_http.asp

```
// Import the built-in Node.js 'http' module.
import http from "http";

// Here, we create an HTTP server using 'createServer' function.
// Takes a callback function that is called when a page is requested.
const app = http.createServer((request, response) => {
    // Set status code and headers for the response.
    response.statusCode = 200;
    response.setHeader("Content-Type", "text/plain");
    // Send a message as the response.
    response.end("Working with HTTP Module and routing");
});

// Here, we define the port number the server should listen on.
const port = 3000;

// Finally, we start the server by calling the 'listen' function.
app.listen(port, () => {
    // Once the server is started, we display a message in the terminal.
    console.log(`Server is running at http://localhost:${port}`);
});
```

Øvelse

Hello HTTP Module



POSTMAN

Getting Started with Postman

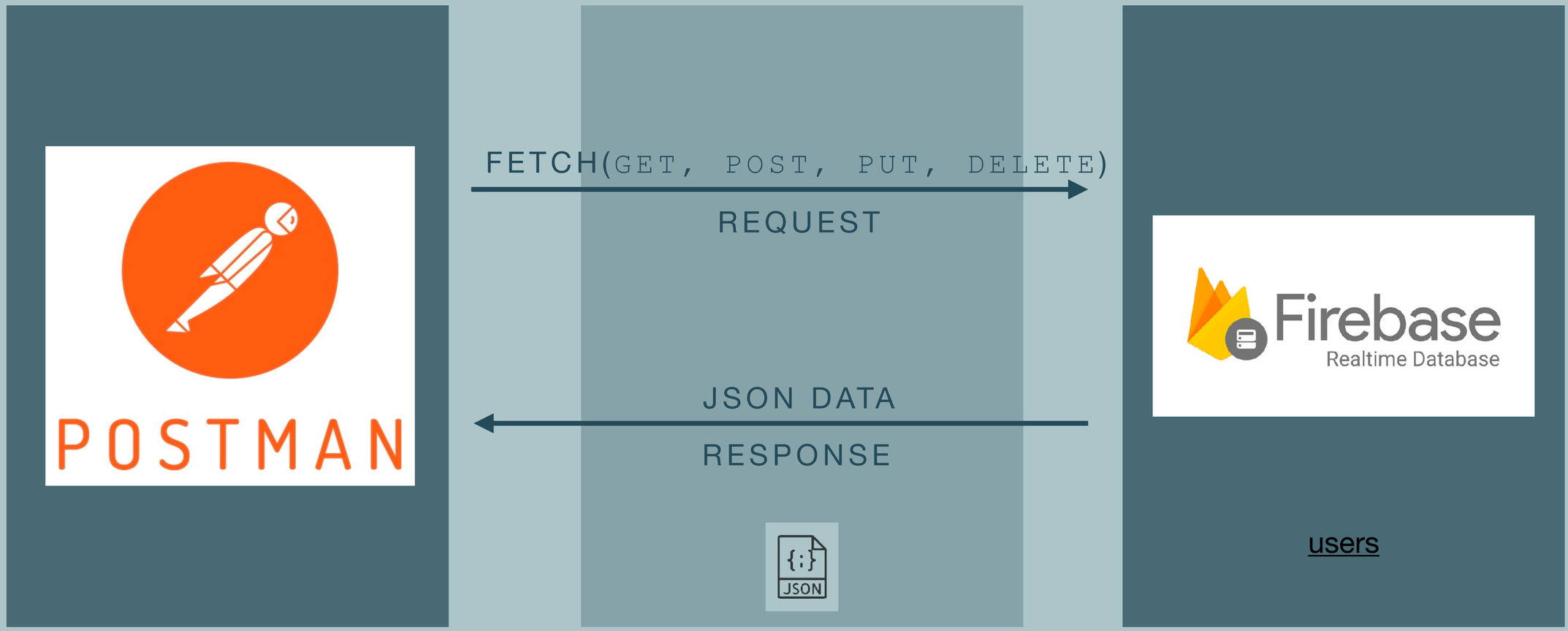
- Postman is a software tool for working with APIs.
- We can use it as client to test our backend API’s and to test APIs in general.
- It helps developers send requests and receive responses from APIs.
- Postman tests API functionality and documents how to use APIs.
- It simplifies API development and testing.
- Developers collaborate using Postman to work together effectively.

Web Development

POSTMAN

FIREBASE REST API

DATA SOURCE



CLIENT

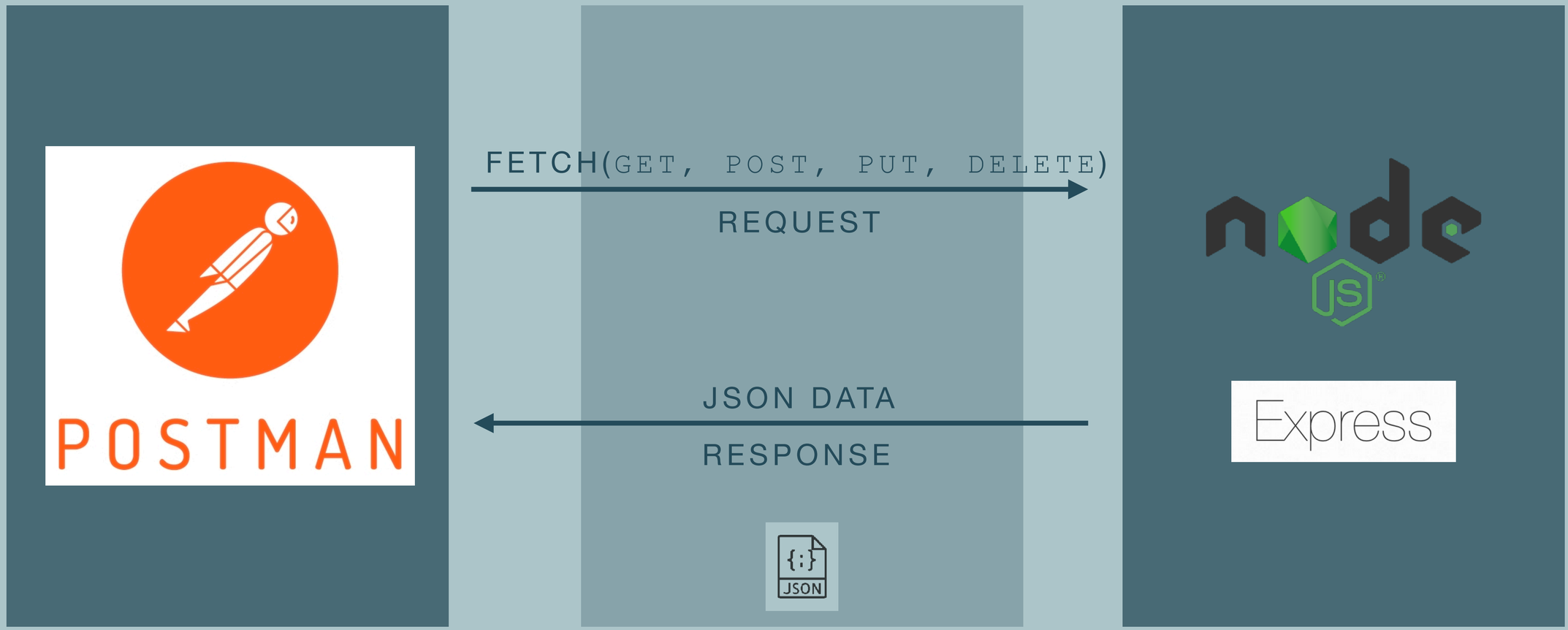
SERVER

Web Development

POSTMAN

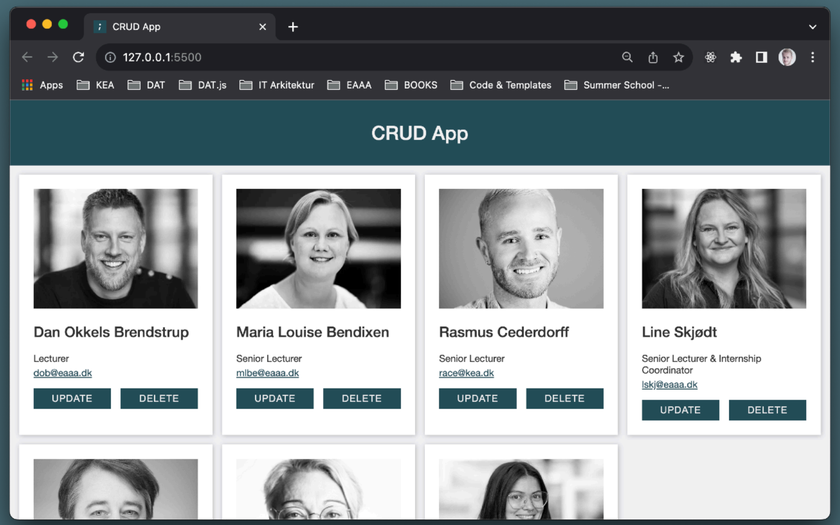
FIREBASE REST API

DATA SOURCE

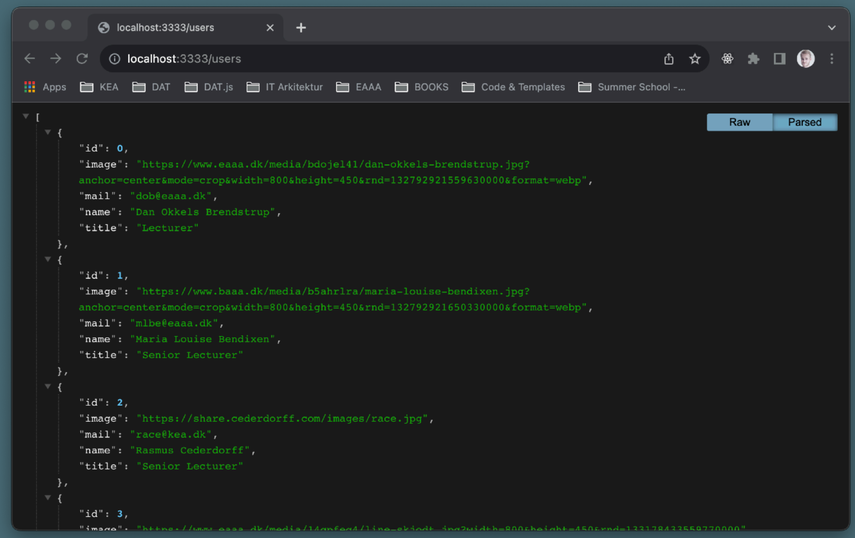


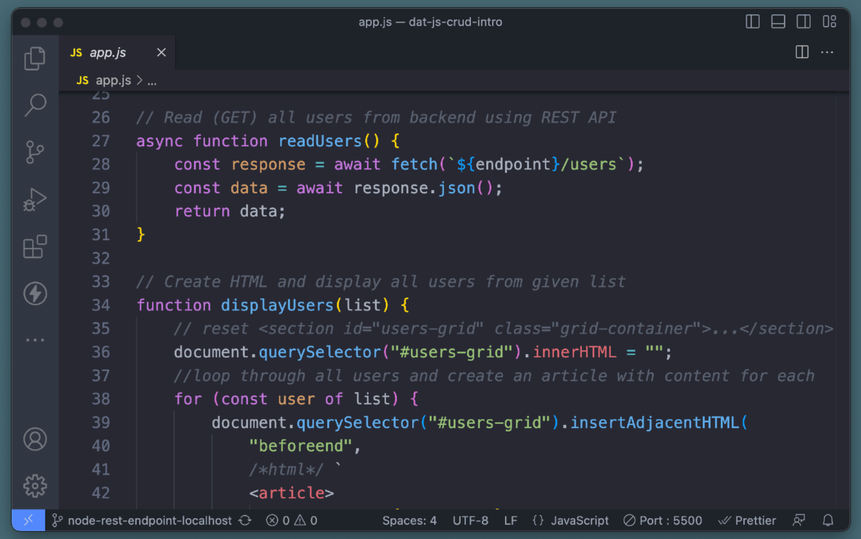
CLIENT

SERVER

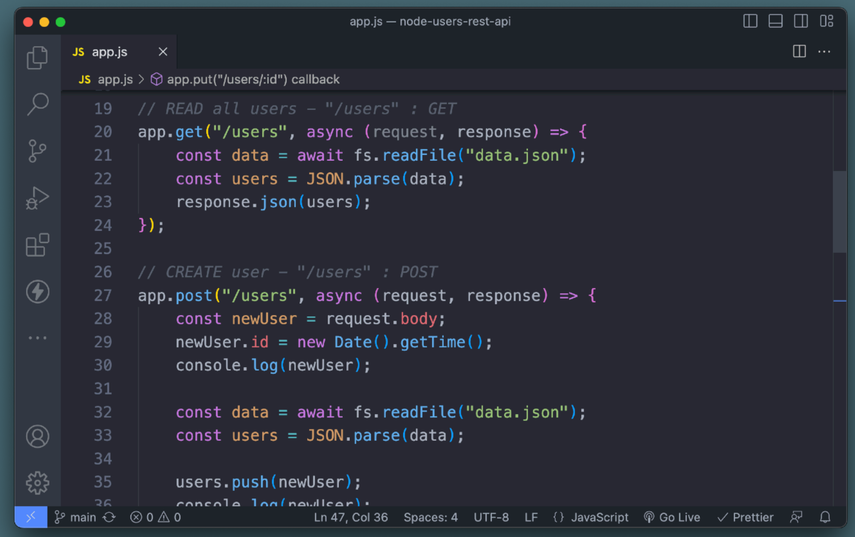


REQUEST



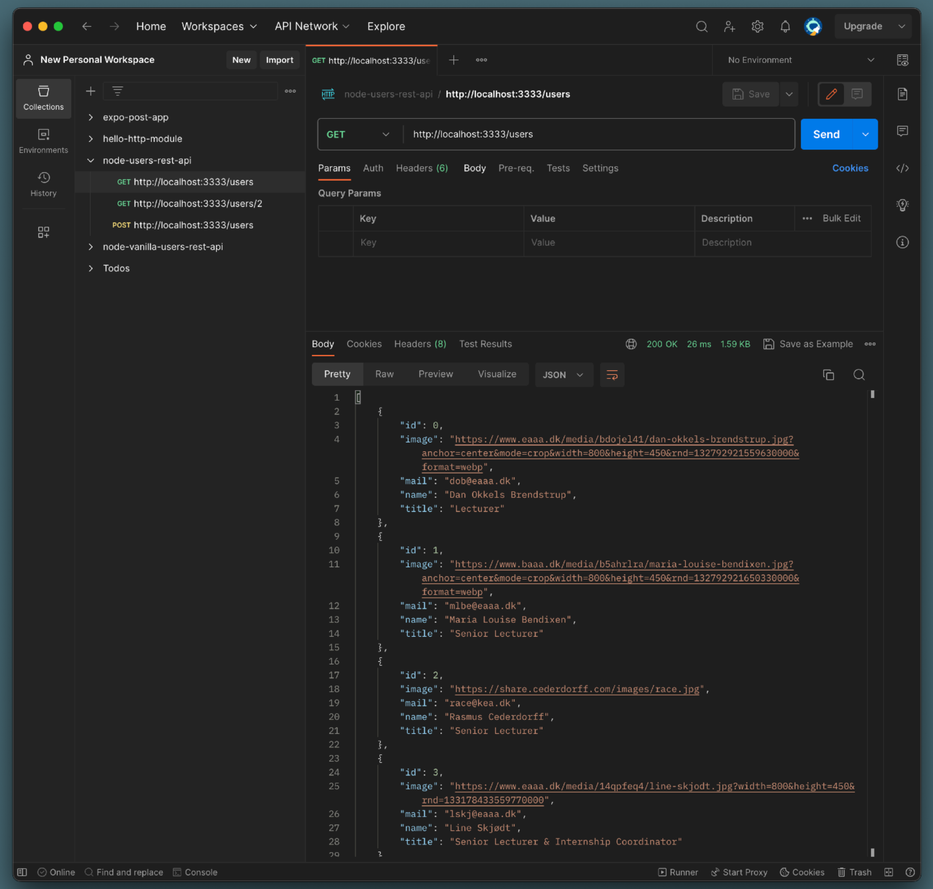


RESPONSE



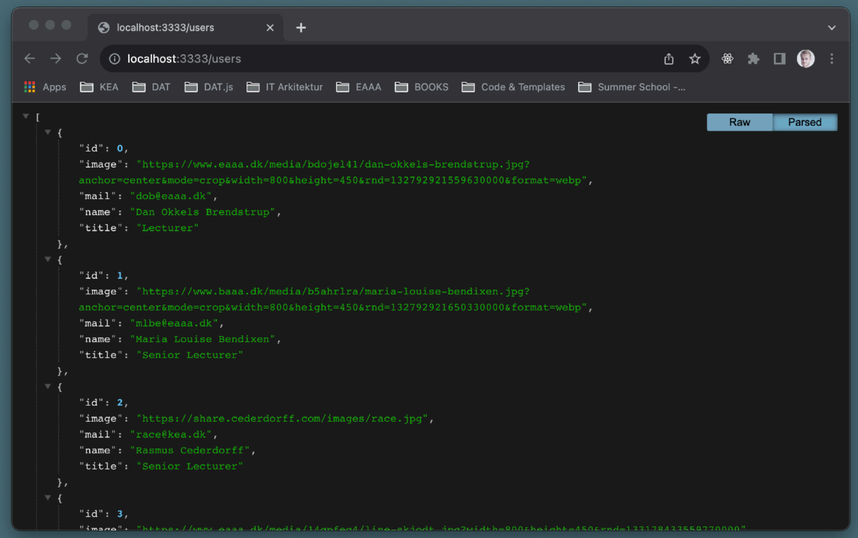
Frontend

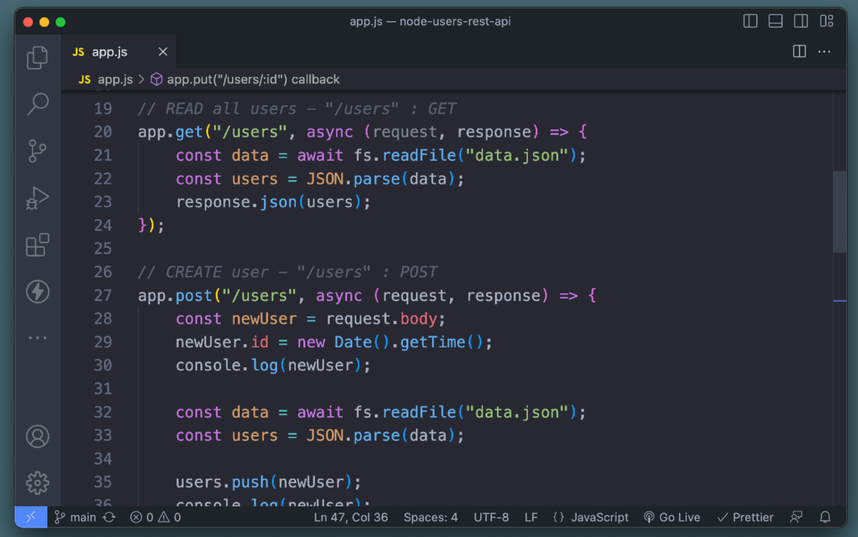
Backend



REQUEST

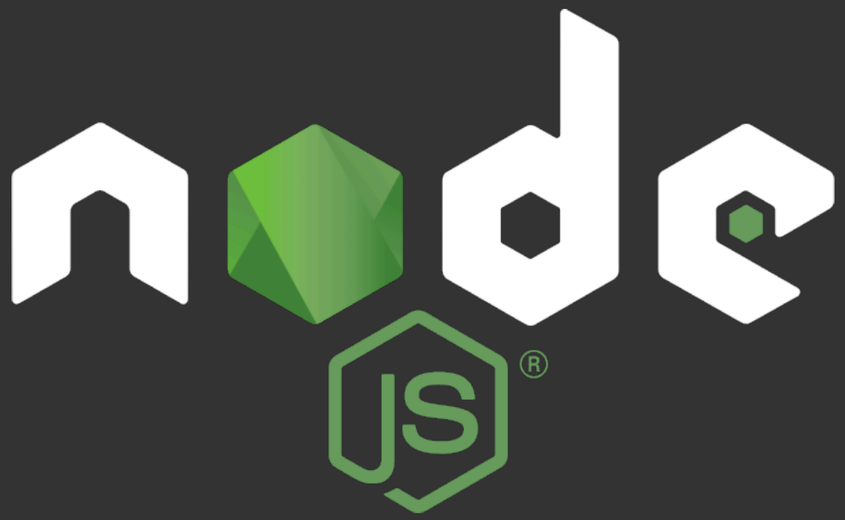
RESPONSE





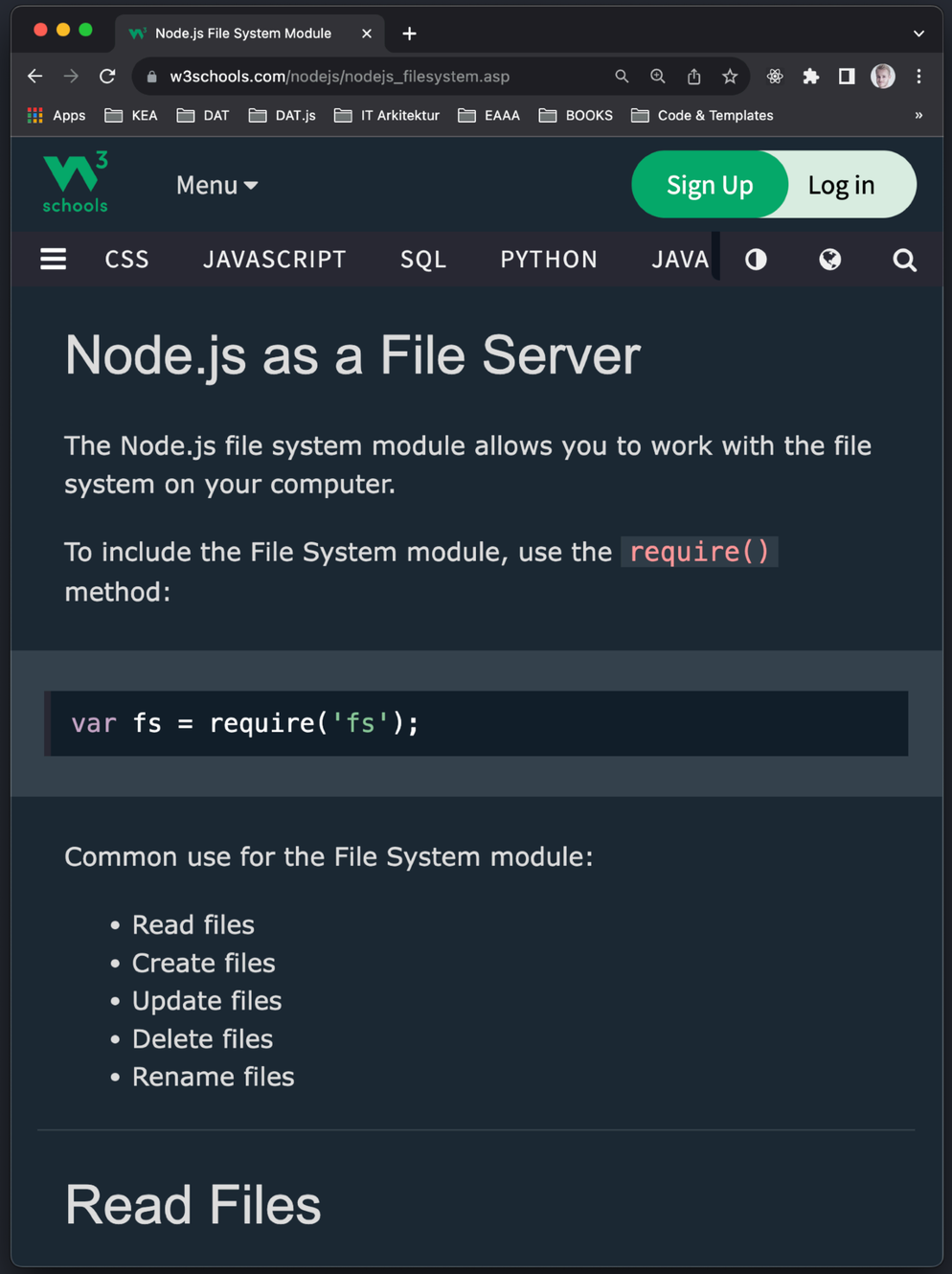
Client

Backend



Node.js File System

Allows us to work with the file system on the computer or server

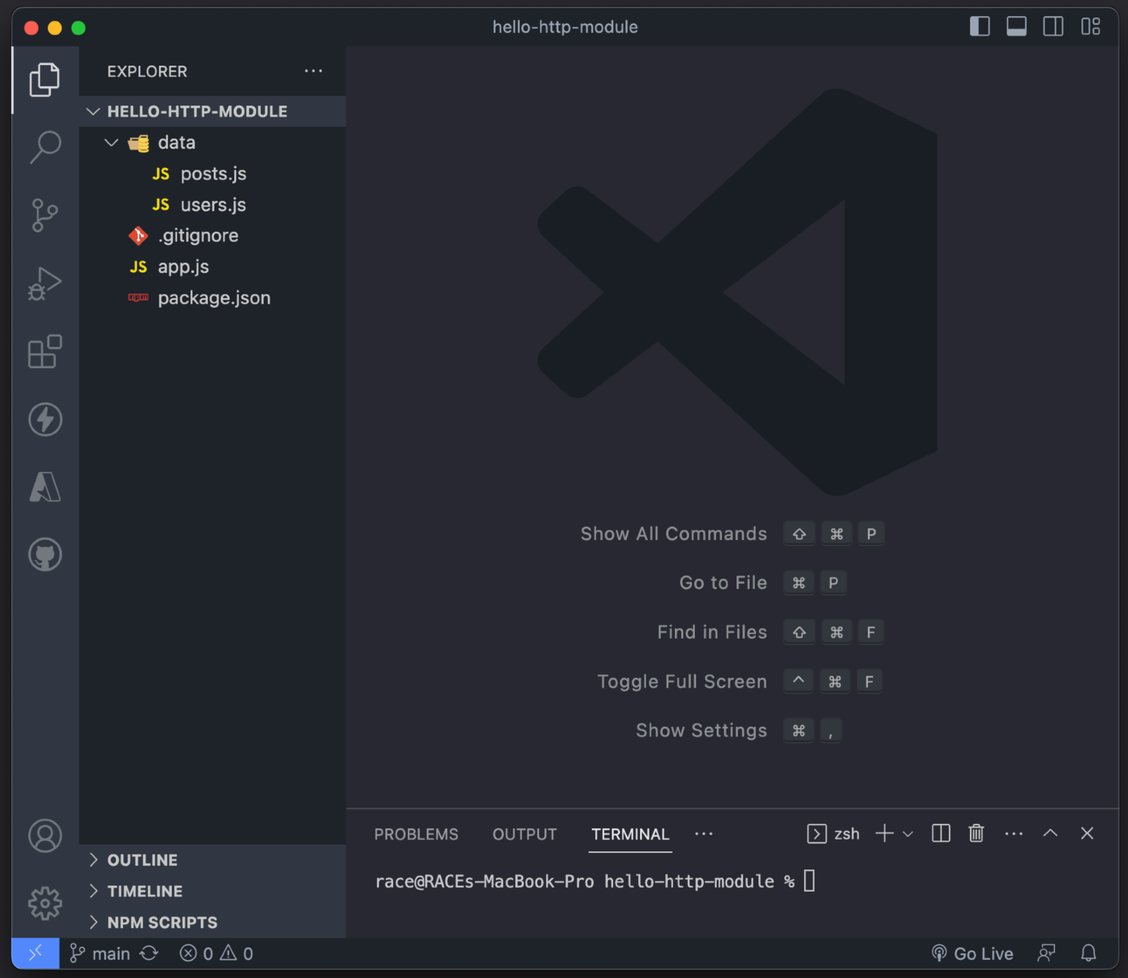


https://www.w3schools.com/nodejs/nodejs_filesystem.asp

```
import fs from "fs/promises";

// Læs fra JSON
const json = await fs.readFile("data/users.json");
console.log(json);
// Parse til JavaScript
const users = JSON.parse(json);
console.log(users);
// Tilføj "user" til "users"
users.push({ name: "Peter Lind", title: "Senior Lecturer" });
// Konverter users til JSON igen
const usersJSON = JSON.stringify(users);
// Skriv til JSON-fil
await fs.writeFile("data/users.json", usersJSON);
```

Øvelse

Node.js File System



Wrap Up

- NPM & JavaScript Libraries
  - Node.js NPM
  - Node.js Upload Files
  - Node.js Send an Email
  - Brug ChatGPT og Google
- Bunden forudsætning: Udvikl REST API med Node.js & Express (Se Fronter)

Code Every Day