

Noteify

Secure Software
Engineering

Wer sind wir?

...

Marcel Kaiser

Informatik
4. Semester

...

Benjamin Wirth

Informatik
4. Semester

...

Jonathan Rech

Informatik
4. Semester



Ablauf

01

Aufbau

Genereller Aufbau der Webapp

02

Funktionen

Funktionen der Anwendung, Dependencies, Umsetzung und Sicherheit

03

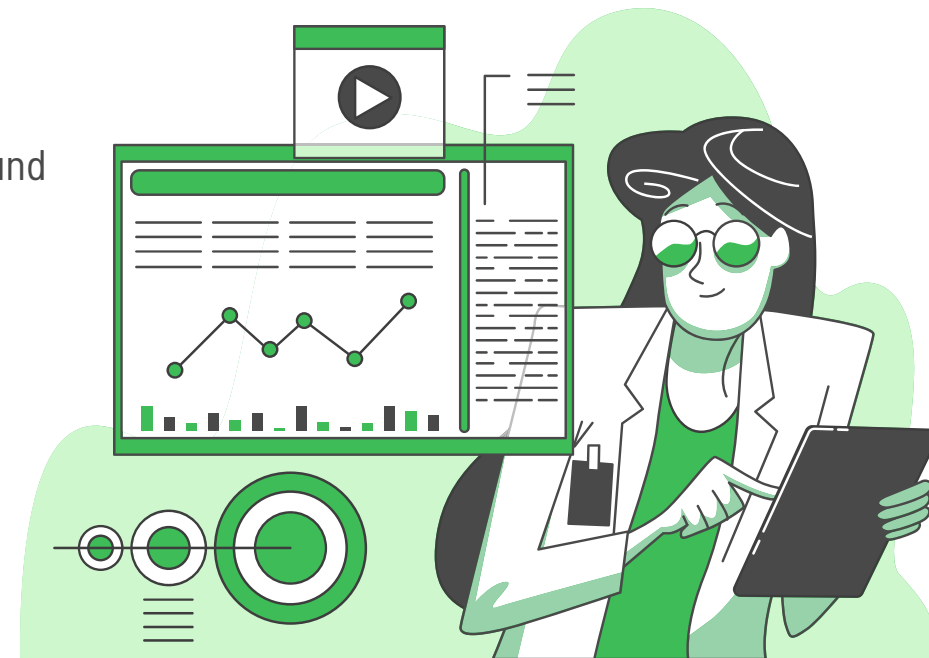
Risiken

Mögliche Sicherheitsrisiken und der Schutz vor diesen

04

Ausblick

Was hätte noch, wenn mehr Zeit und Geld?



01 Aufbau

Genereller Aufbau der Webapp

Aufbau

01

Angular

Frontend

02

Express

Backend

03

MariaDB

Datenbank

04

Docker

Container-Dienst

05

Nginx

Webserver



Angular

Eigenschaften

Frontend-Framework

Typescript

Modular
Wartbar



Komponentenbasiert

Open-Source

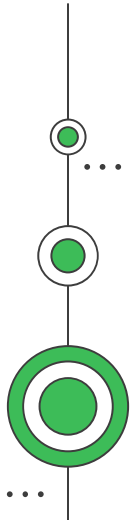
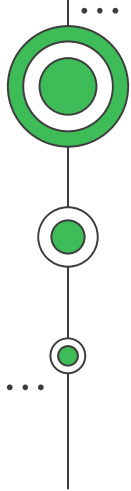
...

Security

Sanitizer

Viele Anwender

Google (Support gegeben)





Express



Express.js

Eigenschaften

Serverseitiges Webframework

Node

Javascript

Lightweight

Open-Source

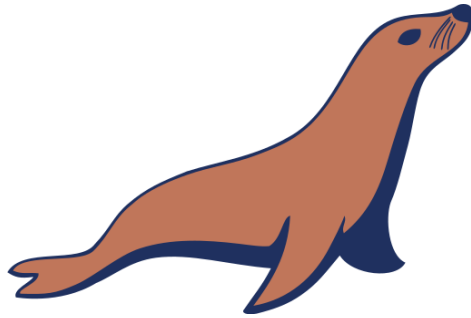
...

Security

Viele Anwender

Teil der Openjs-Foundation





MariaDB

MariaDB

Eigenschaften

Datenbank

Abspaltung von MySQL

Spektakulär unspektakulär

Open Source

...

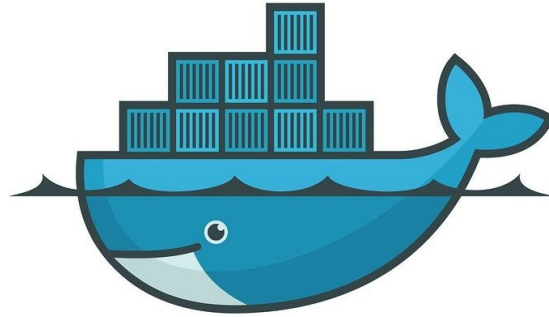
Security

Viele Anwender

Großer Entwickler

State Of The Art

...



Docker

Eigenschaften

Container-Dienst

Plattformübergreifende Nutzbarkeit der Webapp

Open Source

...

Security

Viele Anwender

Virtuelle Umgebung

Am Markt etabliert



Nginx

Eigenschaften

Webserver-Software

Open Source

Security

Detaillierte CVE-Liste

DOS-Protection

Viele weitere (Security-)Features

...

02

Funktionen

Funktionen der Anwendung,
Dependencies, Umsetzung und
Sicherheit

Registrierung neuer Benutzer



Benutzername bereits vorhanden?

Passwort Überprüfung

Verschlüsseltes Abspeichern

SQL-Injection



Benutzername schon vorhanden



```
async function checkIfUserExists(username){
  const checkUsername = 'SELECT COUNT(*) AS count FROM users WHERE username = ?';
  'SELECT COUNT(*) AS count FROM users WHERE username = ?';
  try{
    const conn = await pool.getConnection();
    const result = await conn.query(checkUsername, [username]);
    conn.release();

    const count = result[0].count;
    return count > 0;
  } catch(error) {
    console.error(error);
    throw error;
  }
}
```





Passwort Überprüfung

```
const {username, password} = req.body;  
const zxcvbnResult = zxcvbn(password, [username]);  
const score = zxcvbnResult.score;  
const feedback = zxcvbnResult.feedback;
```

The image shows a 'Register' form with a light yellow background. It contains a text input field with 'JohnDoe', a password input field with masked characters '.....', and a 'Register' button. Below the form, there are two feedback messages in separate boxes. The first is a pink box with the text 'Your password is too weak.' and a close button (X). The second is a light blue box with the text 'This is a frequently used password.' and a close button (X).

Register

JohnDoe

.....

Register

Your password is too weak. X

This is a frequently used password. X



zxcvbn-ts/core

↓ Weekly Downloads

83.955



Contributors 57



+ 46 contributors

- 📖 Readme
- 📄 MIT license
- 🔒 Security policy
- 📈 Activity
- ☆ 544 stars
- 👁 7 watching
- 🔗 48 forks

zxcvbn-ts

Guide

- Introduction
- Getting started
- Installation
 - npm
 - yarn
 - Usage
- Bundler like webpack
- As script tag
- Output
- Change prior to original library
- Migration
- Comparison
- Languages
- Lazy loading
- Matcher
- Options
- User input
- Framework examples
- Examples

Demo

Demo

Getting started

Installation

npm:

```
npm install @zxcvbn-ts/core @zxcvbn-ts/language-common @zxcvbn-ts/language-en ...
```

yarn:

```
yarn add @zxcvbn-ts/core @zxcvbn-ts/language-common @zxcvbn-ts/language-en ...
```

Usage

Bundler like webpack

```
1 import { zxcvbn, zxcvbnOptions } from '@zxcvbn-ts/core'
2 import * as zxcvbnCommonPackage from '@zxcvbn-ts/language-common'
3 import * as zxcvbnLanguageEn from '@zxcvbn-ts/language-en'
4
5 const password = 'somePassword'
6 const options = {
7   translations: zxcvbnCommonPackage.translations,
8   graphs: zxcvbnCommonPackage.adjacencyGraphs,
9   dictionary: {
10     ...zxcvbnCommonPackage.dictionary,
11     ...zxcvbnLanguageEn.dictionary,
12   },
13 }
```

BUNDLE SIZE

24.5kB

MINIFIED

8.8kB

MINIFIED + GZIPPED

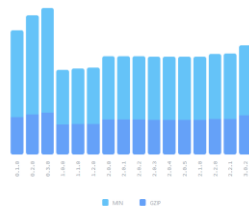
DOWNLOAD TIME

176ms

SLOW 3G

10ms

EMERGING 4G



✓ 0ebc6ff 2 weeks ago 🕒 720 commits



Verschlüsseltes Abspeichern



```
const hashed_password = await argon.hash(password.toString());
```

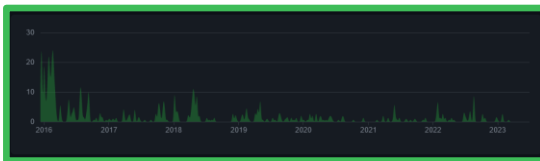
```
async function registerNewUser(username, password){  
  const newUser = 'INSERT INTO users (username, pass) VALUES (?, ?)';  
  
  try{  
    const conn = await pool.getConnection();  
    const result = await conn.query(newUser, [username, password]);  
    conn.release();  
  
    return 1;  
  } catch(error) {  
    return 0;  
  }  
}
```



argon2

Weekly Downloads

153.843



Contributors 39



Readme

MIT license

Code of conduct

Activity

1.6k stars

19 watching

86 forks

Options

Ranieri Althoff edited this page on May 9 · 10 revisions

By default, `argon2.hash` will generate secure hashes according to the security recommendations by the team that develops Argon2.

However, if you need, you can configure a lot of options by passing those as the second parameter of the `hash` function. For example, if you wish to use the Argon2d variant, with 2 ** 16 KB (64 MB) of memory and resulting in a 50 character hash, you would use:

```
const hash = await argon2.hash(password, {
  type: 'argon2d',
  memoryCost: 2 ** 16,
  hashLength: 50,
});
```

Here are the options you can supply and their meaning:

hashLength

The hash length is the length of the hash function output in bytes. Note that the resulting hash is encoded with Base 64, so the digest will be ~1/3 longer.

The default value is 32, which produces raw hashes of 32 bytes or digests of 43 characters.

```
> await argon2.hash('password')
'$argon21$v=19$m=4096,t=3,p=1$5KSc9G0kbvc4uun7EDYRg$3ZlnlCa8+S14tqbM4nRqWvWu3QFHzysPGX7but0ml'

> await argon2.hash('password', {hashLength: 48})
'$argon21$v=19$m=4096,t=3,p=1$71SMD0Qrt6oB7za41+35A$7mf4z59jv5sl3v131B/eJghTTNh0Xg9d7waXzwV/DV8se6J7iU1g'
```

Mind how the second hash is longer.

Pages

Find a page...

Home

Migrating from another hash function

Options

hashLength
timeCost
memoryCost
parallelism
type
secret
raw
version
salt
saltLength
associatedData

+ Add a custom sidebar

Clone this wiki locally

<https://github.com/rnalt/salt/node>

BUNDLE SIZE

330.7 kB

MINIFIED

96.5 kB

MINIFIED + GZIPPED

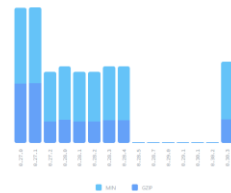
DOWNLOAD TIME

1.93 s

SLOW 3G

110 ms

EMERGING 4G

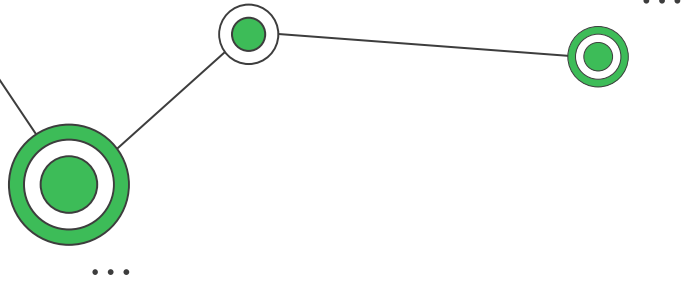


e16fde8 on Mar 5



673 commits

Benutzer Login

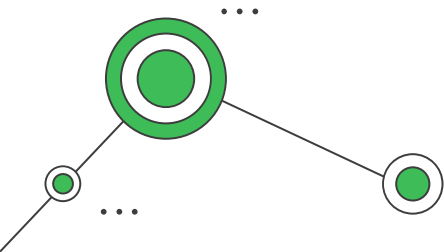


Stimmen Benutzername und Passwort?

Fehlermeldung uneindeutig

JSON Web Token

SQL-Injection





Passwort Überprüfung



```
const sql = `SELECT * FROM users WHERE username = ?`;
const result = await con.query(sql, [username]);
if (result.length > 0) {
```

```
const isMatch = await argon2.verify(hashPassword, password.toString());
```



hashPassword = Aus Datenbank

password = aus HTML-Form (Frontend)



Uneindeutige Fehlermeldung



```
res.send({status:0, error: 'invalid Username or Password', msg:'invalid Username or Password'});
```

Login

JonDa

.....

Login

The given password and username doesn't match.





JSON Web Token



```
let token = jwt.sign({username:username, user_id: userIdQuery[0].user_id.toString()},  
                    jwtSecret,{ expiresIn: '1h' })
```

jwtSecret = Enviroment-Variable

JWT läuft nach einer Stunde ab



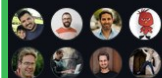
jsonwebtoken

Weekly Downloads

12.460.525



Contributors 100



+ 89 contributors

Readme
MIT license
Security policy
Activity
16.7k stars
264 watching
1.2k forks
Report repository

README.md

jsonwebtoken

Build Dependency
Build Status Dependency Status

An implementation of JSON Web Tokens.

This was developed against [@draft-jetf-oauth-json-web-token-00](#). It makes use of [node-jws](#).

Install

```
$ npm install jsonwebtoken
```

Migration notes

- From v8 to v9
- From v7 to v8

Usage

`jwt.sign(payload, secretOrPrivateKey, [options, callback])`

(Asynchronous) If a callback is supplied, the callback is called with the [JWT](#) or the [JWT](#).

(Synchronous) Returns the [JsonWebToken](#) as string

`payload` could be an object literal, buffer or string representing valid JSON.

Please note that [alg](#) or any other claim is only set if the payload is an object literal. Buffer or string payloads are not checked for JSON validity.

If `payload` is not a buffer or a string, it will be coerced into a string using [JSON.stringify](#).

BUNDLE SIZE

122.9kB

MINIFIED

39.7kB

MINIFIED + GZIPPED

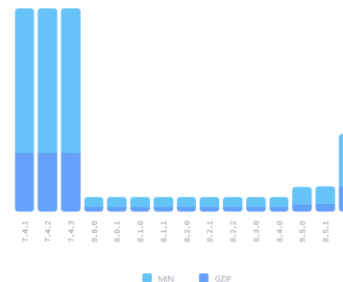
DOWNLOAD TIME

0.79s

SLOW 3G

45ms

EMERGING 4G



✗ a99fd4b on Apr 3

🕒 445 commits

Notiz anlegen



The image shows a 'New Note' form centered within a large green circle. The form has a title 'New Note:' and two input fields: 'Titel' and 'Content'. Below these is a 'Youtube-Video' input field. At the bottom, there is a checkbox labeled 'Note is private:' and a green button labeled 'NewNote'.

Benutzerüberprüfung

Spam-Protection

SQL-Injection



Benutzerüberprüfung



```
router.post('/new', authenticateToken, async function (req, res) {  
  authenticateToken,
```

```
  jwt.verify(req.headers.authorization, jwtSecret, function (err, decoded) {  
    if(decoded){  
      req.user = decoded.user_id;  
      userID = decoded.user_id;  
      if(!search){next();}  
    }else{  
      if(err.name === 'TokenExpiredError') res.send({ message: "Token expired" });  
      else {res.status(401).send({ message: "Unauthorized" }); isAuthenticated = false;}  
    }  
  })  
}
```





Notiz anlegen



```
const query = "INSERT INTO notes(note_id,titel, isPrivate,content, user_id, created, lastChanged, y VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?)"
conn = await pool.getConnection();
const date = new Date().toISOString().slice(0, 19);
const note = await conn.query(query, [uuidv4(), titel, isPrivate, content, authorId, date, date, youtube]);
```

uuidv4 ist garantiert random

State-Of-The-Art



Einmalig für:

- Gerät
- Uhrzeit
- 128-Bit zufällige Zahl

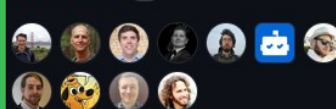
uuid

↓ Weekly Downloads

88.183.798



Contributors 63



Readme
MIT license
Security policy
Activity
13.5k stars
141 watching
903 forks
Report repository

uuid

For the creation of RFC4122 UUIDs

- **Complete** - Support for RFC4122 version 1, 3, 4, and 5 UUIDs
- **Cross-platform** - Support for ...
 - CommonJS, ECMAScript Modules and CDN builds
 - Node 12, 14, 16, 18
 - Chrome, Safari, Firefox, Edge browsers
 - Webpack and rollup.js module bundlers
 - [React Native](#) / [Expo](#)
- **Secure** - Cryptographically-strong random values
- **Small** - Zero-dependency, small footprint, plays nice with "tree shaking" packagers
- **CLI** - Includes the [uuid](#) command line utility

Note Upgrading from [uuid@v1](#)? Your code is probably okay, but check out [Upgrading From uuid@v1](#) for details.

Note Only interested in creating a version 4 UUID? You might be able to use [crypto.randomUUID\(\)](#), eliminating the need to install this library.

Quickstart

To create a random UUID...

1. Install

```
npm install uuid
```

2. Create a UUID (ES6 module syntax)

BUNDLE SIZE

8.3 kB

MINIFIED

3.3 kB

MINIFIED + GZIPPED

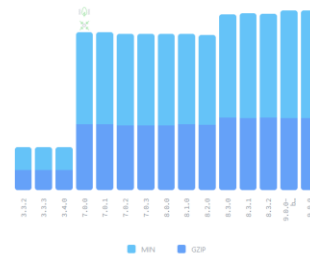
DOWNLOAD TIME

66 ms

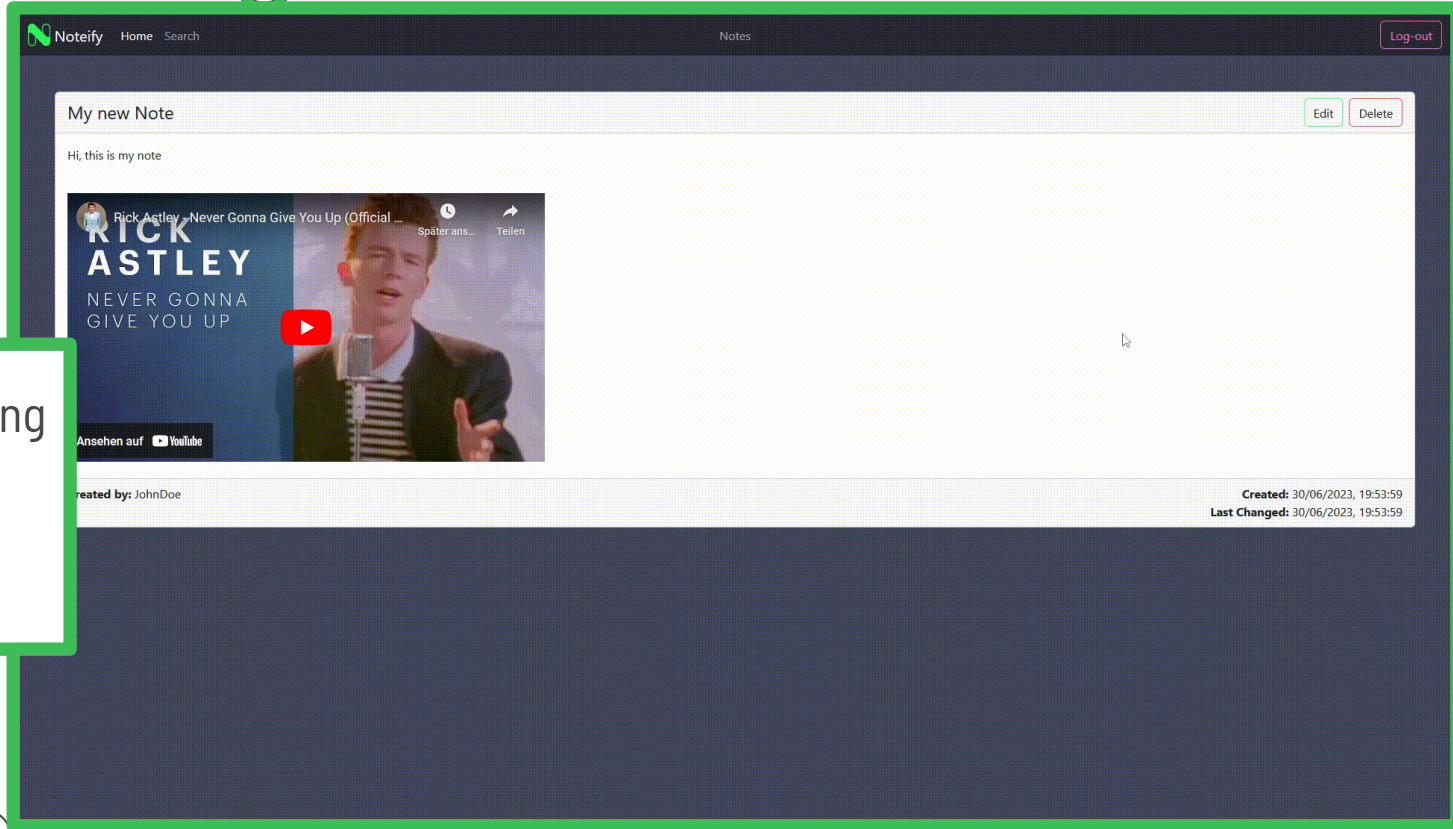
SLOW 3G

4 ms

EMERGING 4G



✓ 4de23a6 on Apr 19 505 commits



Benutzerüberprüfung

Spam-Protection

SQL-Injection



Benutzerüberprüfung



```
router.post('/update/:id',authenticateToken, async function (req, res) {  
    authenticateToken,
```

```
const query2 = "SELECT user_id FROM notes WHERE note_id = ?";  
authorIdQuery = await conn.query(query2,[id]);  
if(authorId == authorIdQuery[0].user_id){
```





Notiz editieren

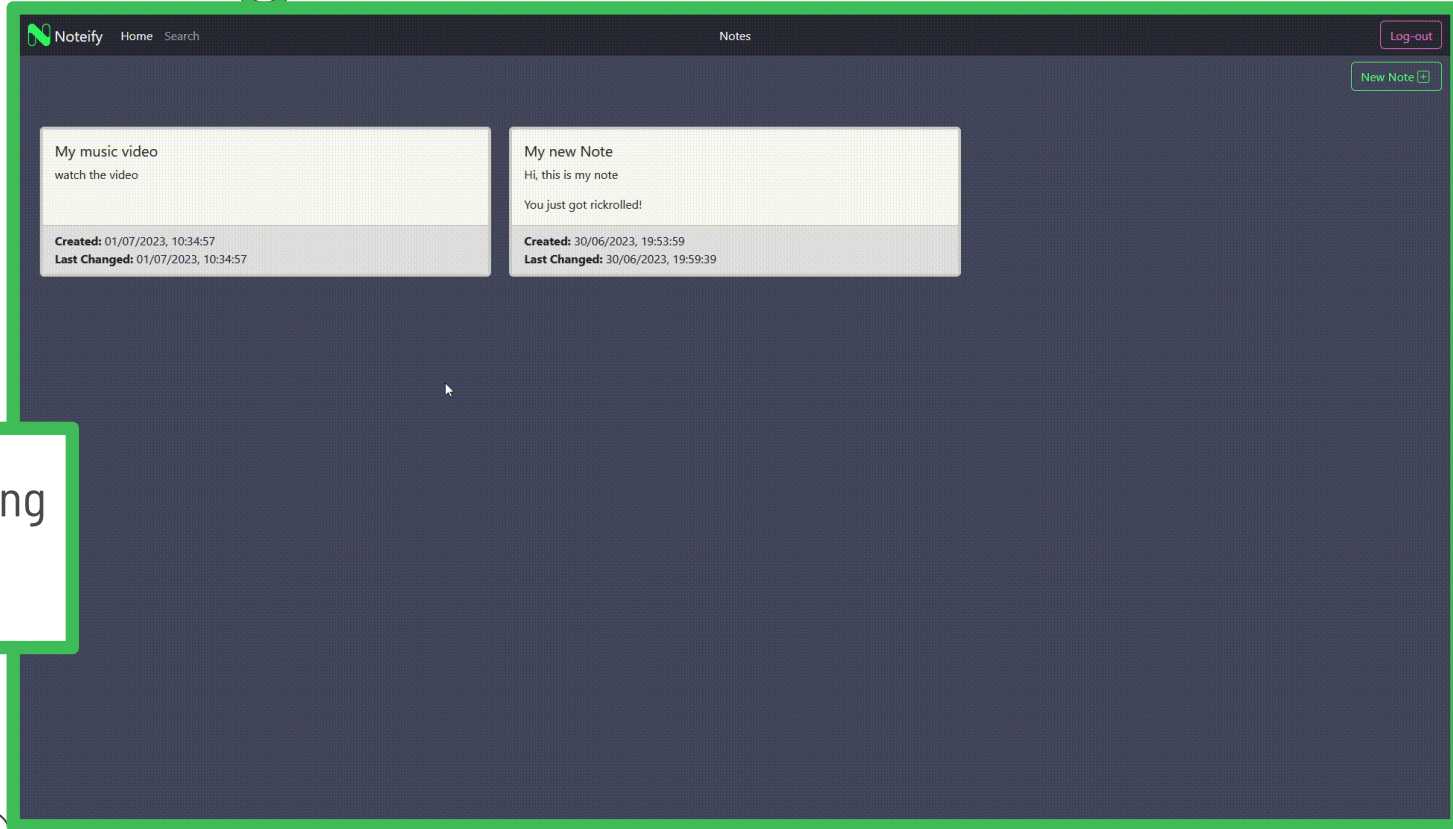


```
"UPDATE notes SET titel = ?, isPrivate = ?, content = ?, lastChanged = ?, youtube = ? WHERE notes.note_id = ?"
```

```
const note = await conn.query(query, [titel, isPrivate, content, date, youtube, id]);
```



Notiz löschen



Benutzerüberprüfung
SQL-Injection



Benutzerüberprüfung + Notiz löschen



```
router.delete('/singlenote/:id', authenticateToken, async function (req, res) {  
  authenticateToken,
```

```
const query = "DELETE FROM notes WHERE notes.note_id = ?"  
conn = await pool.getConnection();  
const query2 = "SELECT user_id FROM notes WHERE note_id = ?";  
authorIdQuery = await conn.query(query2, [id]);  
if(authorId == authorIdQuery[0].user_id){  
  const note = await conn.query(query, [id]);
```

Youtube-Video an Notiz anhängen



The image displays two mobile application screens, each with a green border, representing a 'Notes' app. The left screen, titled 'Notes' at the top, shows a 'New Note:' form. It includes a 'Titel' (Title) input field, a 'Content' text area, and a 'Youtube-Video' section with a checkbox for 'Note is private:'. A green 'NewNote' button is at the bottom right. The right screen, also titled 'Notes', shows an 'Update Note' form. It features a text area containing 'My music video' and 'watch the video', followed by a YouTube URL: 'https://www.youtube.com/watch?v=vfHBOKa_ZG0'. It also has a 'Note is private:' checkbox (which is checked) and a green 'UpdateNote' button at the bottom right.

Schadcode abfangen

URL-Überprüfung



Regex



```
var regExp = /^.*((youtu.be\/)|(v\/)|(\u\/\w\/)|(embed\/)|(watch\?))\??v?=?([^\#&?]*).*;/;
```

<https://m.youtube.com/watch?v=lal0y8Mbfdc>

<https://youtu.be/oTJRivZTMLs&feature=channel>

http://m.youtube.com/v/OzM3nApSvMg?fs=1&hl=en_US&rel=0

<http://youtu.be/-wtIMTCHWul>





Darstellung des iFrames



```
<youtube-player videoId="{{notes[0].youtube}}" suggestedQuality="highres" [height]="360" [width]="640"></youtube-player>
```

```
<youtube-player
```

Eigene Funktion von Angular

Keine Darstellung, wenn fehlerhafte URLs oder Schadcode

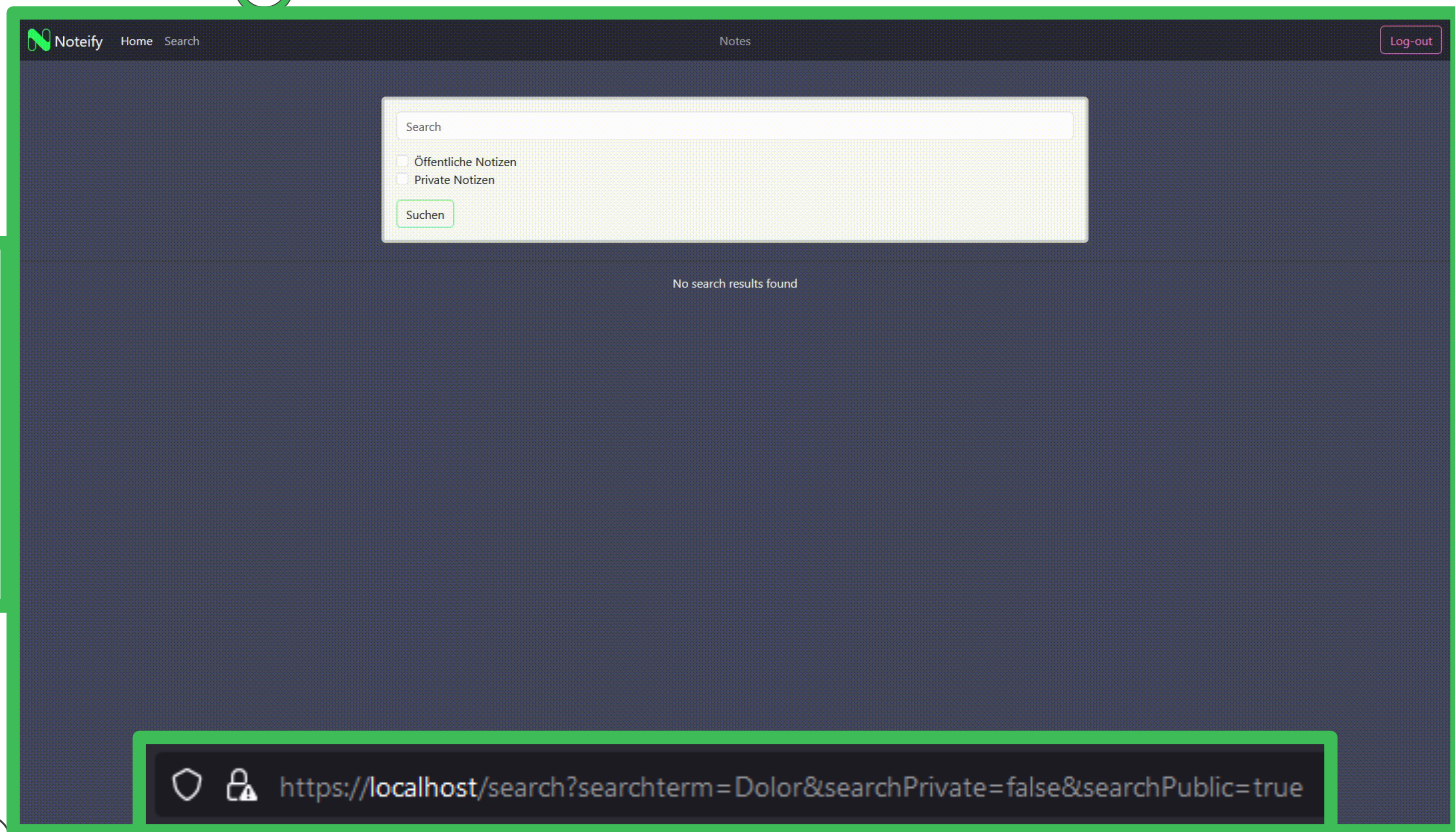


Suchen von Notizen

XSS-Protection

SQL-Injection

Authentifizierung





XSS-Protection



```
<p class="text-muted mt-3" *ngIf="searchterm">Suchbegriff: {{ searchterm }}</p>
```

Eigene Funktion von Angular

Angular übernimmt Sanitization





Suchen von Notizen



```
const query = "SELECT notes.* from notes JOIN users ON notes.user_id = \
users.user_id WHERE ((notes.titel LIKE CONCAT('%', ?, '%') OR \
notes.content LIKE CONCAT('%', ?, '%') OR \
users.username LIKE CONCAT('%', ?, '%')) \
AND notes.isPrivate = 1) AND users.user_id = ?";
```



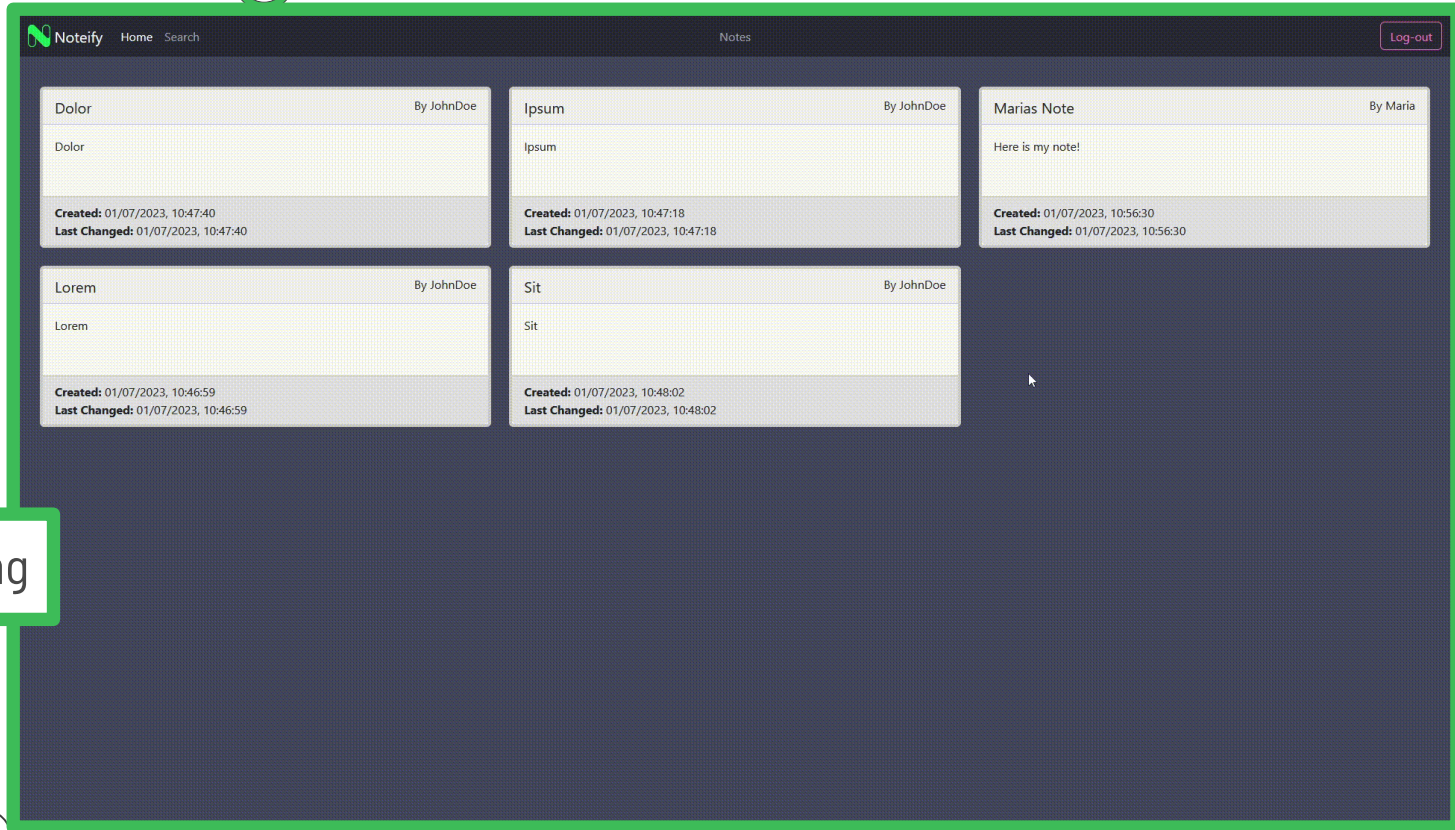
Private Search



```
if (searchPrivate) {  
    authenticateToken(req, res, next, true);  
}
```

Prepared Statement ist äquivalent in „public Search“

... Anzeigen von öffentlichen/eigenen Notizen



Benutzerüberprüfung



Benutzerüberprüfung + eigene Notizen

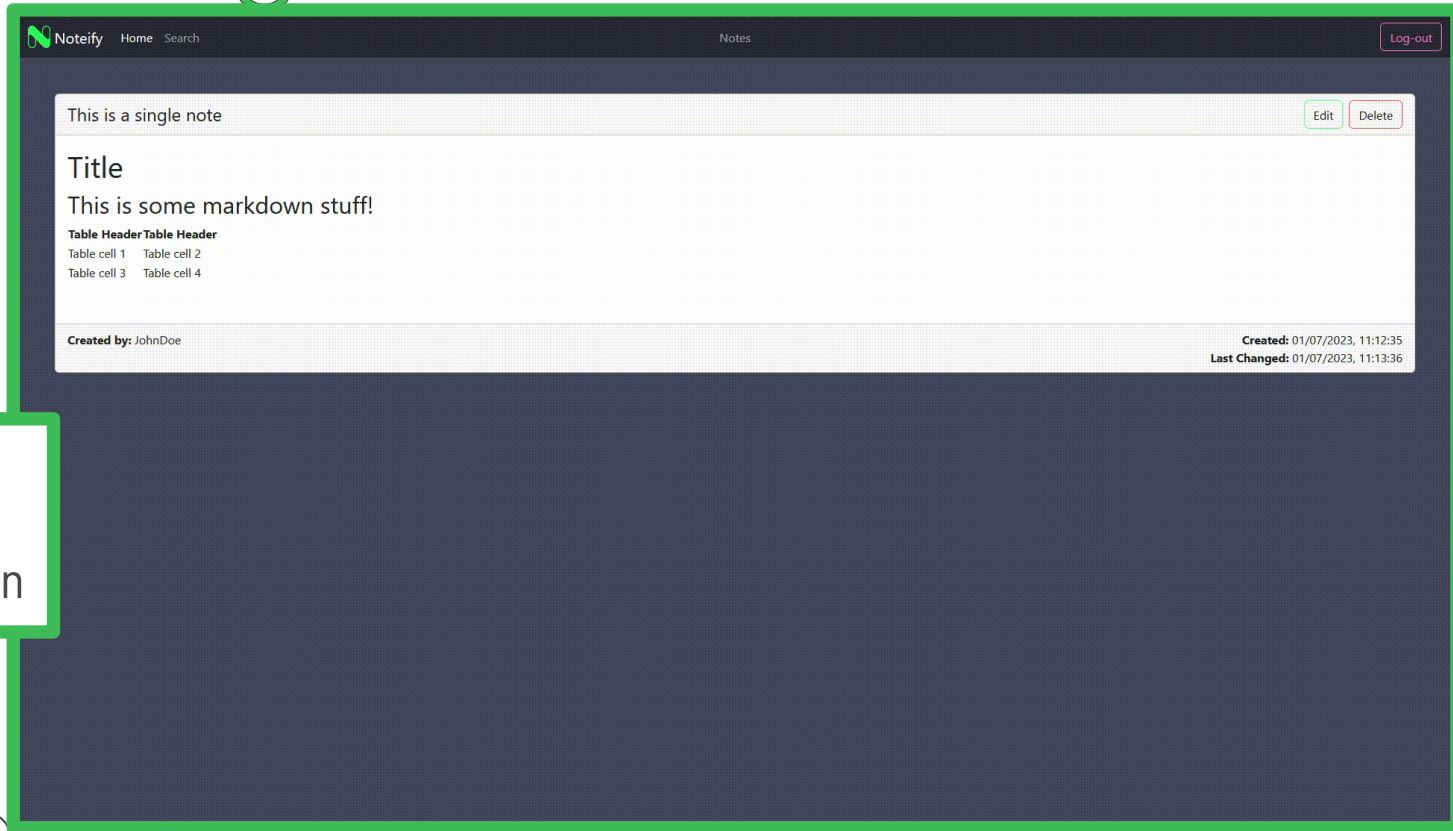
```
router.get('/usernotes', authenticateToken, async function (req, res) {  
  authenticateToken,
```

```
const query = "SELECT notes.*,users.username FROM notes JOIN users \  
ON notes.user_id = users.user_id WHERE notes.user_id = ?";
```



Öffentliche Notizen: Ohne Benutzerüberprüfung und andere Where-Bedingung

Anzeigen einzelner Notizen



XSS-Protection

Private Notizen lesen



XSS-Protection



```
<p class="card-text" [innerHTML]="notes[0].content"></p>
```

Eigene Funktion von Angular

Angular übernimmt Sanitization





Markdown- und HTML-Support



```
elem.content = marked.marked.parse(elem.content.replace(/^[\\u200B\\u200C\\u200D\\u200E\\u200F\\uFEFF]/, "")))
```

`marked.marked.parse`

HTML-Support durch [innerHTML]-Tag gegeben



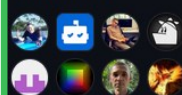
marked

↓ Weekly Downloads

7.614.383



Contributors 161



📖 Readme

⚖️ View license

📄 Code of conduct

⚖️ Security policy

📈 Activity

☆ 30k stars

👁️ 391 watching

🍴 3.3k forks

Marked Documentation

Getting Started

- Demo
- Installation
- Usage
- Specs
- Tools
- Security

Advanced Usage

- Options
- Known Extensions
- Inline Markdown
- Highlighting
- Workers
- CLI Extensions

Extensibility

- marked.use()
- Renderer
- Tokenizer
- Walk Tokens
- Hooks
- Custom Extensions
- Async Marked

Contributing

- Lexer
- Parser
- Design Principles
- Priorities
- Testing
- Code of Conduct

Authors

Publishing

- Versioning

License

Marked is

1. built for speed.**
2. a low-level markdown compiler for parsing markdown without caching or blocking for long periods of time.***
3. light-weight while implementing all markdown features from the supported flavors & specifications.***
4. available as a command line interface (CLI) and running in client- or server-side JavaScript projects.

* Still working on metrics for comparative analysis and definition.

** As few dependencies as possible.

*** Strict compliance could result in slower processing when running comparative benchmarking.

Demo

Checkout the [demo page](#) to see marked in action 🚀

These documentation pages are also rendered using marked 🍷

Installation

CLI: `npm install -g marked`

In-browser:

```
npm install marked
npm install @types/marked # For TypeScript projects
```

Usage

Warning: ⚠️ Marked does not [sanitize](#) the output HTML. If you are processing potentially unsafe strings, it's important to filter for possible XSS attacks. Some filtering options include [DOMPurify](#) (recommended), [is-xss](#), [sanitize-html](#) and

BUNDLE SIZE

52kB

MINIFIED

17.2kB

MINIFIED + GZIPPED

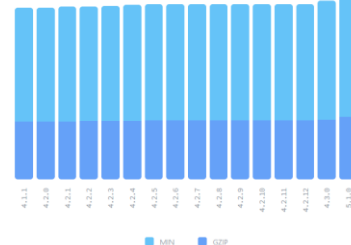
DOWNLOAD TIME

345ms

SLOW 3G 📶

20ms

EMERGING 4G 📶



✓ 884c782 5 days ago 🕒 3,038 commits



Private Notizen lesen

```
+  
  
https://localhost/note/9e9c6dd4-7037-4c85-a400-de95d0c7461a
```

UUIDv4:

- Kaum Guessable
- Bruteforce kaum möglich



03

Risiken

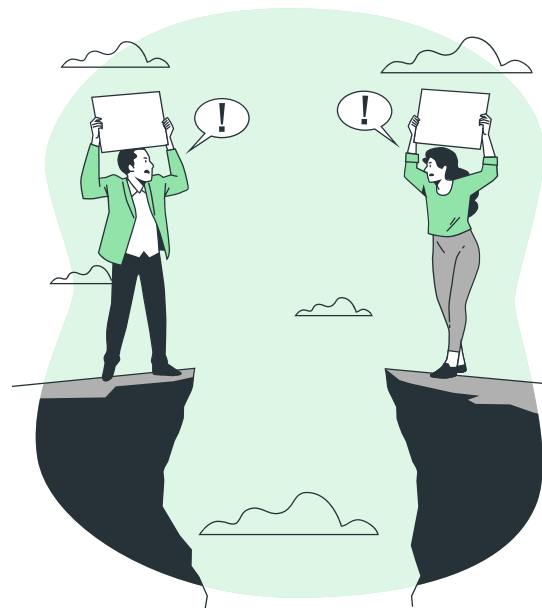
Mögliche Sicherheitsrisiken und der
Schutz vor diesen

Unsichere Kommunikation

- HTTPS zwischen Frontend und Backend
- Erzwingen von HTTPS-Verbindungen des Clients

XSS Angriffe

- Angular Sanitizer





Denial of Service & Brute-Force

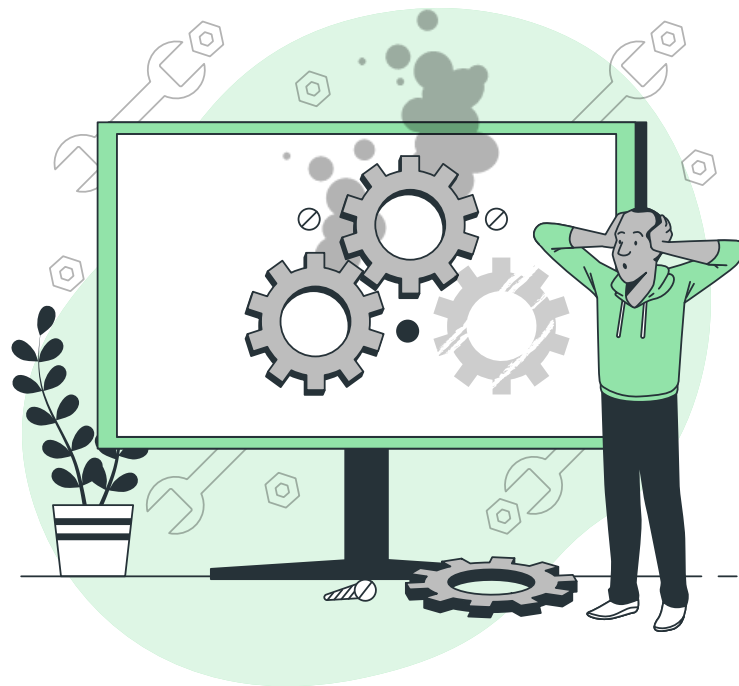


Nginx-Konfiguration

- Rate-Limiting

Express-Konfiguration

- Rate-Limiting

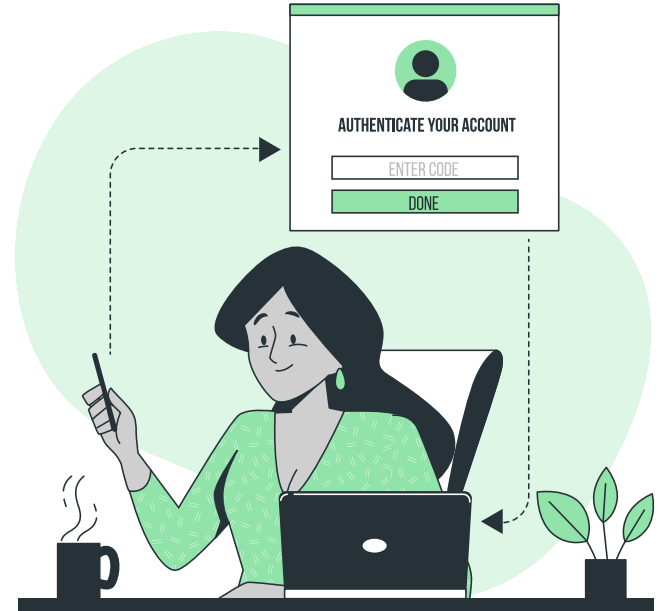


Passwörter

- ZXCVCBN: Enforzen starker Passwörter

Sessions

- JSON Web Tokens
- Laufzeit < 1 Stunde



Datenbank

- Datenbank nur vom Container aus erreichbar

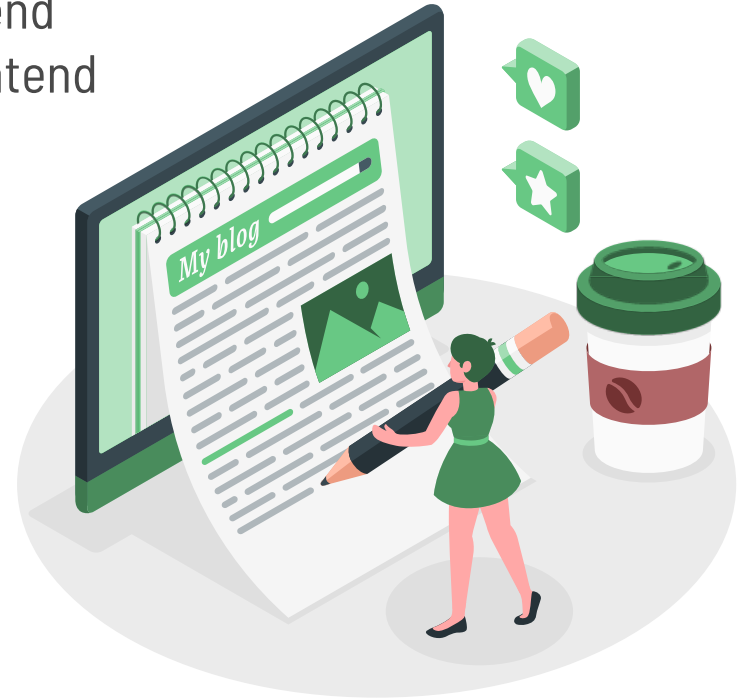
CI/CD

- Dependabot
- **Github-Actions**
- CodeQL
- Frontend-Test
- Backend-Test



Security/Logging/Monitoring-Failures

- Überall uneindeutige Fehlermeldungen
- Keine Fehlermeldungen von Backend an Frontend
- Keine Fehlermeldungen von Datenbank an Frontend



04

Ausblick

Was hätte noch, wenn mehr Zeit und
Geld?

Was muss noch getan werden?

- Impressum und Datenschutzerklärung
 - Web-Application-Firewall
 - DMZ einrichten
 - Verschlüsseln aller Daten in der Datenbank
 - Mehr Pen-Testing (ZAP, Greenbone)
 - Echte signierte Zertifikate
 - Datenbank Backup Strategie
-
- Passwort vergessen
 - OAuth-Funktionalität



Danke!

Zeit für eine Live-Demo 😊

<https://github.com/marcel951/Noteify>

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik** and illustrations by **Stories**

