**Technische Dokumentation**

Autor: Pascal Förber, Luis Hutterli

Datum: 24.01.2025

Version: 2.0

# Inhaltsverzeichnis

[1 Inhaltsverzeichnis 2](#_Toc188640782)

[2 Analyse 4](#_Toc188640783)

[2.1 Projektbeschrieb 4](#_Toc188640784)

[2.2 Anforderungen 4](#_Toc188640785)

[2.3 Realisierungstechnologie 5](#_Toc188640786)

[2.4 Use-Cases 5](#_Toc188640787)

[2.5 ERM 6](#_Toc188640788)

[3 Design 7](#_Toc188640789)

[3.1 Layout 7](#_Toc188640790)

[3.2 Datenbankschema 8](#_Toc188640791)

[3.3 API Endpunkte 8](#_Toc188640792)

[3.4 Testliste 9](#_Toc188640793)

[4 Realisation 11](#_Toc188640794)

[4.1 Backend 11](#_Toc188640795)

[4.2 Frontend 11](#_Toc188640796)

[4.2.1 Routing 11](#_Toc188640797)

[5 Test 12](#_Toc188640798)

[6 Deployment 13](#_Toc188640799)

[6.1 Benutzeranleitung 13](#_Toc188640800)

[6.1.1 Backend 13](#_Toc188640801)

[6.1.2 Frontend 13](#_Toc188640802)

[6.2 Live Deployment 13](#_Toc188640803)

[7 Benutzeranleitung 14](#_Toc188640804)

[7.1 Account 14](#_Toc188640805)

[7.1.1 Erstellen 14](#_Toc188640806)

[7.1.2 Anmelden 14](#_Toc188640807)

[7.1.3 E-Mail-Verifizieren 15](#_Toc188640808)

[7.1.4 Einstellungen 16](#_Toc188640809)

[7.2 Hauptseite – Desktop 16](#_Toc188640810)

[7.2.1 Song Ausleihen 16](#_Toc188640811)

[7.2.2 Song Zurückgeben 17](#_Toc188640812)

[7.2.3 Song Abspielen / Steuern 17](#_Toc188640813)

[7.2.4 Song Steuern über OS-Integration 18](#_Toc188640814)

[7.3 Hauptseite – Mobile 19](#_Toc188640815)

[7.3.1 Song Ausleihen 19](#_Toc188640816)

[7.3.2 Song Zurückgeben 19](#_Toc188640817)

[7.3.3 Song Abspielen / Steuern 20](#_Toc188640818)

[7.3.4 Song Steuern über OS-Integration 20](#_Toc188640819)

# Analyse

## Projektbeschrieb

Es soll eine Web-App entwickelt werden, mit welcher man eine elektronische JukeBox mit NFT-Songs (Non-Fungible Tokens) benutzen und verwalten kann.

## Anforderungen

Ein jukeStack Benutzer (z.B. Steven Hatch) soll sich mit seiner E-Mail Adresse (Steven.Hatch@gmx.net) und einem Passwort (z.B. djfhU78&wR) in der jukeStack-App registrieren können.

Sobald er sich registriert hat, kann er sich mittels E-Mail und Passwort in der jukeStack-App anmelden. Gelingt ihm dies, so sieht er einerseits eine Liste mit NFT-Songs (z.B. Ian Kelly, Speak your mind, 4:08, 2008), die er ausleihen kann und eine zweite Liste mit bereits ausgeliehenen Songs, die aber im Moment noch leer ist.

Nun kann er aus der ersten Liste einen Titel anwählen (z.B. Speak your mind) und ihn ausleihen. Sind die unten angeführten Bedingungen erfüllt, so wird die Ausleihe gewährt und der Song auf die Liste der ausgeliehenen Songs gesetzt. Die Liste der ausgeliehenen Songs enthält nun diesen Song, welcher um das Ausleihdatum ergänzt wurde (z.B. Ian Kelly, Speak your mind, 4:08, 2008, 11.12.2024 14:03) und kann nun abgespielt werden.

Die Bedingungen für eine Ausleihe lauten wie folgt:

- Ein Benutzer darf maximal nur 5 Songs gleichzeitig ausleihen.

- Ein Song kann, da jeder Song in dieser App ein NFT ist, gleichzeitig maximal nur einmal ausgeliehen sein.

- Ein Song kann für maximal n Tage ausgeliehen werden.

Sind die ersten zwei Bedingungen nicht erfüllt, so soll eine entsprechende Warnung ausgegeben werden.

Ist ein Song ausgeliehen, kann er so oft angehört werden, wie der Benutzer es möchte.

Ein ausgeliehener Song, kann, wenn er nicht mehr gewünscht wird, wieder zurückgegeben werden.

Zudem gibt es noch einen Administrator:in (z.B. Liza Suller, Liza.Suller@tokentec.ch, kljht564xc). Logt die sich ein, sieht sie statt der Songs, eine Liste mit allen Benutzern. Wählt sie einen Benutzer an, sieht sie wiederum dessen Ausleihen und kann diese auch rückgängig machen. Zudem kann der Admin die maximale Ausleihdauer n (siehe Bedingung 3) festlegen (default = 1).

Die Passwörter müssen unbedingt gehasht und gesalzen (kein MD5!) in der DB abgelegt werden.

## Realisierungstechnologie

Die Web-App wird folgende mit folgenden Technologien realisiert:

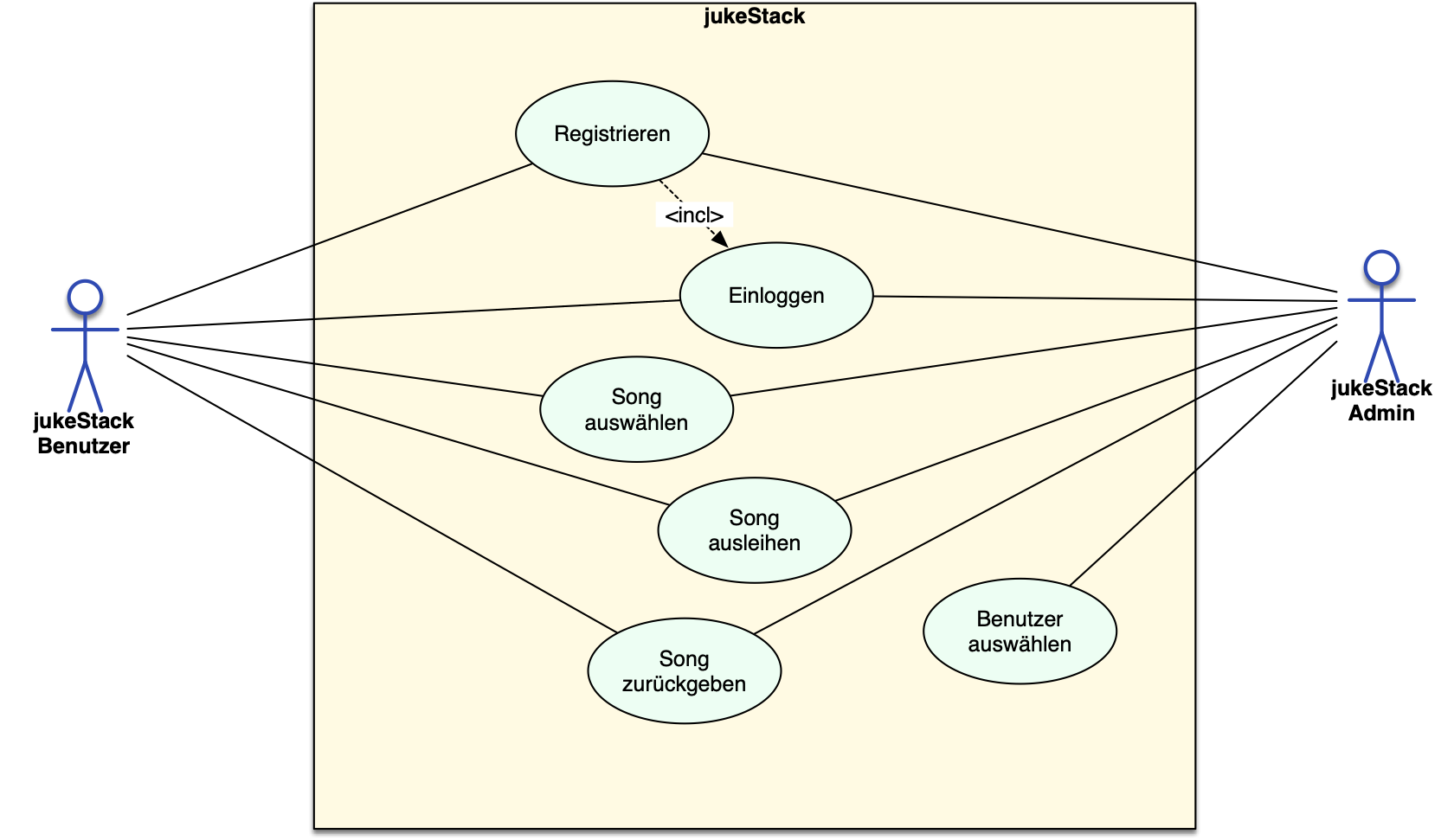
Front-End: React (vite, TypeScript), (UI und weitere Util Library) *laufend auf Google Firebase*

Back-End: MySQL, Java Vert.X Server *laufend auf Google Cloud Run*

Hinzu kommen diese Anbieter:

* Google Firebase (Frontend Deployment)
* Google Cloud Run (Backend Deployment)
* Cloudflare DNS (DNS, Proxy & Cache für die Webseite)
* Cloudflare R2 (Object Storage für Bilder und Song Dateien)
* Zoho Mail (Für automatische Mails und Test Accounts)

## Use-Cases

Aus dem folgenden Diagramm sind die gewünschten Anwendungsfälle erkennbar.

## ERM

A diagram of a block diagram

Description automatically generated

# Design

## Layout

Login/Signup:  
A screenshot of a login screen

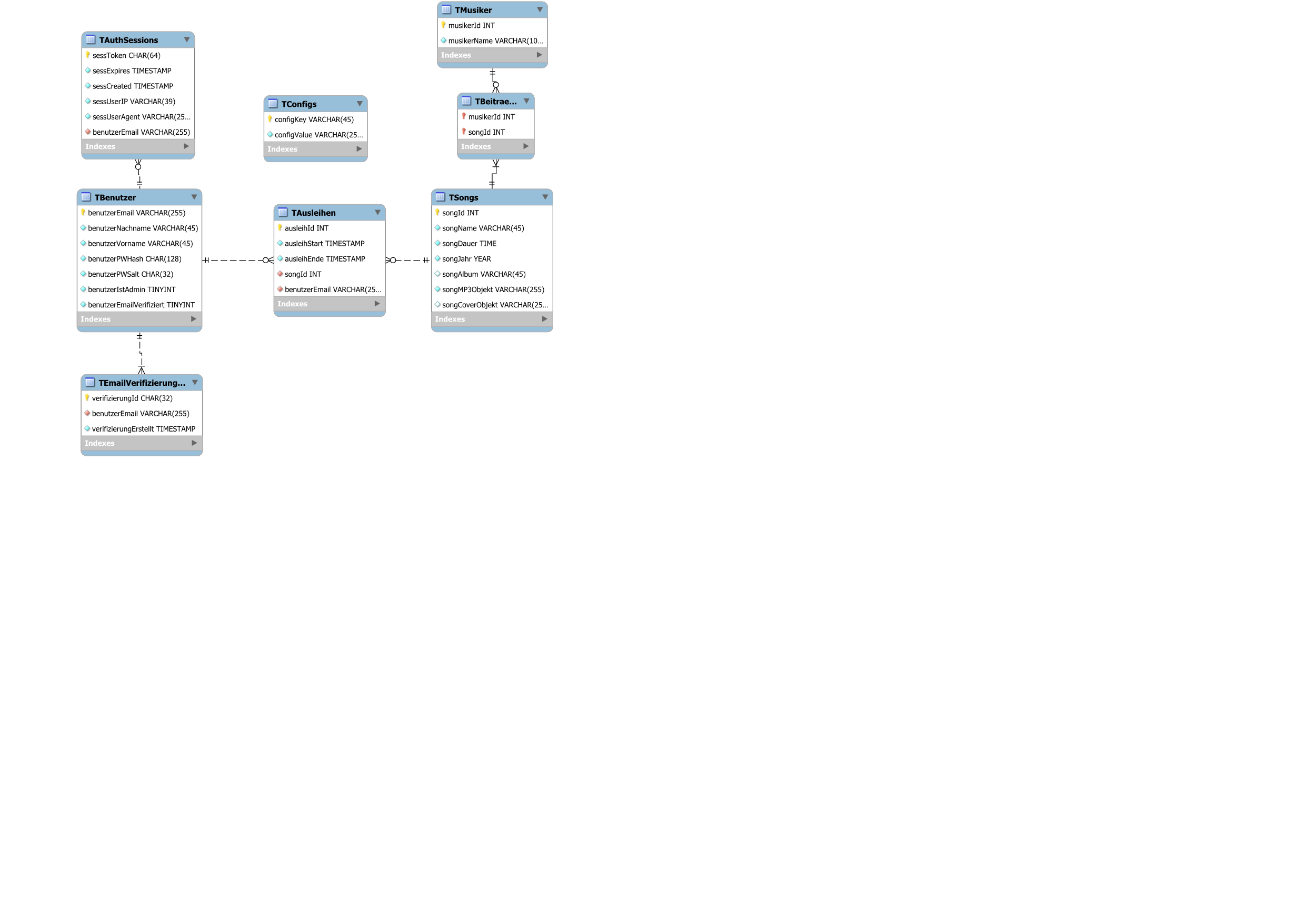
Description automatically generated

Main:

A screenshot of a video

Description automatically generated

## Datenbankschema



## API Endpunkte

Basis: jukestack.ch/api, *(Nach ch.lsh.ims.jukestack.MainVerticle.java)*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Route** | **Method** | **Description** | **Response** | **Implemented** |
| /user | POST | Create user | 201 | Fully Implemented |
| /user | GET | Get user info | 200 | Fully Implemented |
| /user | PUT | Update user info | 200 | Fully Implemented |
| /user | DELETE | Delete user | - | Not Implemented |
| /auth/login | POST | Login | 201 | Fully Implemented |
| /auth/logout | POST | Logout | 200 | Fully Implemented |
| /auth/verify | POST | Check Token validility | 201 | Fully Implemented |
| /auth/refresh | POST | Refresh Token | 201 | Fully Implemented |
| /auth/sendVerify | POST | Send verification email | 200 | Fully Implemented |
| /auth/verifyEmail | GET | Verify email via tokenized link | 302 | Fully Implemented |
| /songs | GET | List all songs | 200 | Fully Implemented |
| /lend | GET | List all borrowed songs | 200 | Fully Implemented |
| /lend/{id} | POST | Lend a song | 200 | Fully Implemented |
| /lend/{id} | DELETE | Return a song | 200 | Fully Implemented |
| /lend/{id}/listen | GET | Generate a link to song file | 200 | Fully Implemented |
| /admin/users | GET | List all users | 200 | Fully Implemented |
| /admin/users/{email}/lend | GET | List a users borrowed songs | 200 | Fully Implemented |
| /admin/lend/{id} | PUT | Update a users lend by lendId | 200 | Fully Implemented |
| /admin/lend/{id} | DELETE | Return a users lend by lendId | 200 | Fully Implemented |

## Testliste

|  |  |  |  |
| --- | --- | --- | --- |
| **Funktionale Testfälle** | | | |
| **Test** | **Erwartetes Ergebnis** | **Bestanden** | **Bemerkung** |
| Layout Registrierung: Benutzer erstellen | Der Benutzer wird erfolgreich in der Datenbank gespeichert. |  |  |
| Layout Anmelden: Benutzer Anmelden | Ein Session-Cookie wird erstellt, der Benutzer wird eingeloggt und zur App weitergeleitet. |  |  |
| Layout Einstellungen: Benutzer Informationen werden Angezeigt | Benutzer Informationen werden in die vorgefertigten Felder eingesetzt |  |  |
| Layout Einstellungen: Benutzer Informationen ändern | Die gewählten Benutzerinformationen werden aktualisiert. Bei Änderung der Email muss die Verifikation erneut durchgeführt werden. |  |  |
| Alle Layouts: Im Header: Abmelden | Der Benutzer wird abgemolden und auf die Start Seite geleitet. Die Session wird Serverseitig deaktiviert. |  |  |
| Layout Main: Neuen Account erstellt oder Email geändert | Der Benutzer kann sich eine Verifizierungsmail schicken lassen. |  |  |
| Verifizierungslink in der erhaltenen Email anklicken | Die Email des Benutzers wird verifiziert und er kann die App jetzt normal Benutzen. |  |  |
| Layout Main: Songs anzeigen | Alle verfügbaren Songs werden korrekt angezeigt. |  |  |
| Layout Main: Ausgeliehene Songs anzeigen | Alle ausgeliehenen Songs werden korrekt angezeigt. |  |  |
| Layout Main: Song ausleihen (auf verfügbaren Song klicken) | Der Song wird in die persönliche Bibliothek hinzugefügt. |  |  |
| Layout Main: Song zurückgeben (auf ausgeliehenen Song klicken) | Der Song wird aus der persönlichen Bibliothek entfernt. |  |  |
| Layout Main: Song abspielen | Der Song wird abgespielt, und Ton ist hörbar. |  |  |
| Layout Main: Maximal 5 Songs ausleihen | Eine Fehlermeldung wird angezeigt, und der Song wird nicht hinzugefügt. |  |  |
| Admin - Alle Benutzer anzeigen | Alle Benutzer (außer andere Admins) werden angezeigt. |  |  |
| Admin - Ausleihen eines Benutzers anzeigen | Alle ausgeliehenen Songs des Benutzers werden korrekt angezeigt. |  |  |
| Admin - Ausleihe eines Benutzers zurückgeben | Der Song wird aus der Bibliothek des Benutzers entfernt. |  |  |
| Admin - Ausleihe eines Benutzers ändern | Die neuen Ausleihtage werden korrekt gespeichert. |  |  |
|  |  |  |  |
|  | |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Destruktive Testfälle** | | | |
| **Test** | **Erwartetes Ergebnis** | **Bestanden** | **Bemerkung** |
| Benutzer erstellen - Unvollständige Eingaben | Eine Fehlermeldung wird angezeigt, und der Benutzer wird nicht erstellt. |  |  |
| Anmelden - Falsche Zugangsdaten | Eine Fehlermeldung wird angezeigt, und der Benutzer wird nicht eingeloggt. |  |  |
| Song ausleihen - Nicht vorhandener Song | Eine Fehlermeldung wird angezeigt. |  |  |
| Song abspielen - Kein Zugriff | Der Song wird nicht abgespielt, und eine Fehlermeldung wird angezeigt. |  |  |
| Admin - Ungültige Änderungen an Ausleihen | Eine Fehlermeldung wird angezeigt, und die Änderung wird nicht übernommen. |  |  |
| Generell - Zugriff auf API Endpunkte mit Falscher oder Fehlender Authentifizierung | Der Zugriff sollte nicht erlaubt werden und ein Status 401 Unauthorized soll zurückgegeben werden. |  |  |
| Generell - Zugriff auf bestimmte API Endpunkte ohne die Email bestätigt zu haben | Der Zugriff sollte nicht erlaubt werden und der Benutzer sollte zu einer entsprechenden Verifizierungsseite weitergeleitet werden. |  |  |
|  |  |  |  |
|  | |  | |

# Realisation

## Backend

Ich habe mich für ein Java-Backend entschieden, da ich auf der suche nach einer guten Serverless Cloud Platform gesucht habe. Hierzu habe ich diese Webseite verwendet: <https://www.srvrlss.io/>

Ursprünglich wollte ich das Backend in Rust schreiben und auf Cloudflare Workers deployen, jedoch war die Integration mit dritten SQL-Datenbank zu aufwändig und ich musste eine andere Plattform suchen.

Ich habe dann schliesslich Google Cloud Run gefunden und ausgewählt, da es sehr flexibel ist, ich ein wenig Erfahrung mit der Google Cloud Plattform habe und es für meinen Anwendungsfall gratis ist. GCR (Google Cloud Run) kann direkt mit Java verwendet werden, jedoch mit SpringBoot, zum glück kann man jedoch sein eigenes Deployment mit Docker erstellen und jede Sprache und Framework deployen. Ich habe mich also für einen Vert.X Server entschieden, da er nach meiner Recherche als einer der schnellsten und flexibelsten herausging.

## Frontend

Das Frontend ist eine React Webapplikation, statt JavaScript habe ich TypeScrypt verwendet.

Als Packager habe ich die Vite Framework verwendet, da sie das Entwickeln und Deployen einfach etwas schneller macht.

Um das Frontend zu Deployen habe ich Google Firebase Hosting verwendet, da ich Firebase schon sehr oft verwendet habe und gute Erfahrungen damit gemacht habe.

### Routing

Die Applikation verwendet ReactRouter und die BrowserRouter Komponente, dass bedeutet, dass Routes bzw. Pfade gleich funktionieren wie bei einer herkömmlichen Webseite (z.B. <https://jukestack.ch/app/login>). Die andere Variante wäre die HashRouter komponente gewesen, jedoch wird hier vor dem Pfad auch immer ein # Platziert (z.B. [https://jukestack.ch/**#**/app/login](https://jukestack.ch/#/app/login)). Jedoch funktioniert der direkte Zugriff auf einen Unterpfad mit dem Browserrouter nicht direkt, da zuerst alle Anfragen auf alle Unterpfäde zum Hauptpfad und dem index.hmtl weitergeleitet werden müssen.

Das erziele ich mit einer speziellen Konfiguration in [firebase.json](https://gitlab.com/KREATIV-LSH/jukestack_luis/-/blob/main/firebase.json?ref_type=heads);

Dort «rewrite» ich alle Anfragen auf allen Pfäden zu index.html.

Auch kann ich durch die direkte Integration von Google Firebase in die Google Cloud Platform den GCR Container (Backend) auf den Pfad /api binden, da ich alle Anfragen auf diesen Pfad an den GCR Container weiterleite.

# Test

|  |  |  |  |
| --- | --- | --- | --- |
| **Funktionale Testfälle** | | | |
| **Test** | **Erwartetes Ergebnis** | **Bestanden** | **Bemerkung** |
| Layout Registrierung: Benutzer erstellen | Der Benutzer wird erfolgreich in der Datenbank gespeichert. | Ja | - |
| Layout Anmelden: Benutzer Anmelden | Ein Session-Cookie wird erstellt, der Benutzer wird eingeloggt und zur App weitergeleitet. | Ja | - |
| Layout Einstellungen: Benutzer Informationen werden Angezeigt | Benutzer Informationen werden in die vorgefertigten Felder eingesetzt | Ja | - |
| Layout Einstellungen: Benutzer Informationen ändern | Die gewählten Benutzerinformationen werden aktualisiert. Bei Änderung der Email muss die Verifikation erneut durchgeführt werden. | Ja | - |
| Alle Layouts: Im Header: Abmelden | Der Benutzer wird abgemolden und auf die Start Seite geleitet. Die Session wird Serverseitig deaktiviert. | Ja | - |
| Layout Main: Neuen Account erstellt oder Email geändert | Der Benutzer kann sich eine Verifizierungsmail schicken lassen. | Ja | - |
| Verifizierungslink in der erhaltenen Email anklicken | Die Email des Benutzers wird verifiziert und er kann die App jetzt normal Benutzen. | Ja | - |
| Layout Main: Songs anzeigen | Alle verfügbaren Songs werden korrekt angezeigt. | Ja | - |
| Layout Main: Ausgeliehene Songs anzeigen | Alle ausgeliehenen Songs werden korrekt angezeigt. | Ja | - |
| Layout Main: Song ausleihen (auf verfügbaren Song klicken) | Der Song wird in die persönliche Bibliothek hinzugefügt. | Ja | - |
| Layout Main: Song zurückgeben (auf ausgeliehenen Song klicken) | Der Song wird aus der persönlichen Bibliothek entfernt. | Ja | - |
| Layout Main: Song abspielen | Der Song wird abgespielt, und Ton ist hörbar. | Ja | - |
| Layout Main: Maximal 5 Songs ausleihen | Eine Fehlermeldung wird angezeigt, und der Song wird nicht hinzugefügt. | Ja | - |
| Admin - Alle Benutzer anzeigen | Alle Benutzer (außer andere Admins) werden angezeigt. | Ja | - |
| Admin - Ausleihen eines Benutzers anzeigen | Alle ausgeliehenen Songs des Benutzers werden korrekt angezeigt. | Ja | - |
| Admin - Ausleihe eines Benutzers zurückgeben | Der Song wird aus der Bibliothek des Benutzers entfernt. | Ja | - |
| Admin - Ausleihe eines Benutzers ändern | Die neuen Ausleihtage werden korrekt gespeichert. | Ja | - |
|  |  |  |  |
| ***Tests durchgeführt am 24.01.2025 von Luis Hutterli*** | |  | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Destruktive Testfälle** | | | |
| **Test** | **Erwartetes Ergebnis** | **Bestanden** | **Bemerkung** |
| Benutzer erstellen - Unvollständige Eingaben | Eine Fehlermeldung wird angezeigt, und der Benutzer wird nicht erstellt. | Ja | - |
| Anmelden - Falsche Zugangsdaten | Eine Fehlermeldung wird angezeigt, und der Benutzer wird nicht eingeloggt. | Ja | - |
| Song ausleihen - Nicht vorhandener Song | Eine Fehlermeldung wird angezeigt. | Ja | - |
| Song abspielen - Kein Zugriff | Der Song wird nicht abgespielt, und eine Fehlermeldung wird angezeigt. | Ja | - |
| Admin - Ungültige Änderungen an Ausleihen | Eine Fehlermeldung wird angezeigt, und die Änderung wird nicht übernommen. | Ja | - |
| Generell - Zugriff auf API Endpunkte mit Falscher oder Fehlender Authentifizierung | Der Zugriff sollte nicht erlaubt werden und ein Status 401 Unauthorized soll zurückgegeben werden. | Ja | - |
| Generell - Zugriff auf bestimmte API Endpunkte ohne die Email bestätigt zu haben | Der Zugriff sollte nicht erlaubt werden und der Benutzer sollte zu einer entsprechenden Verifizierungsseite weitergeleitet werden. | Ja | - |
|  |  |  |  |
| ***Tests durchgeführt am 24.01.2025 von Luis Hutterli*** | |  | |

# Deployment

## Benutzeranleitung

Mehr über dieses Thema findet man im README.

### Backend

Das Backend kann mit dem deploy.sh Script in ./backend/ direkt zu Google Cloud Run Deployed werden. Hierzu braucht es lediglich ein Google Cloud Projekt mit den richtigen Berechtigungen und die entsprechenden Umgebungsvariablen im Google Secrets Manager.

Lokal kann das Backend mit den zwei Scipts build-docker.sh und run-docker.sh in ./backend/ ausgeführt werden. Hierzu braucht man auch keine jdk, lediglich nur Docker.

### Frontend

Das Frontend wird mit dem Befehl «npm run build» gebundelt und dann mit dem Befehl  
 «firebase deploy» auf eine Google Firebase Seite deployed.

Lokal kann man die webseite wie folgt ausführen: «npm run dev»

(für beides müssen zuerst die Dependencys installiert werden: «npm i»)

## Live Deployment

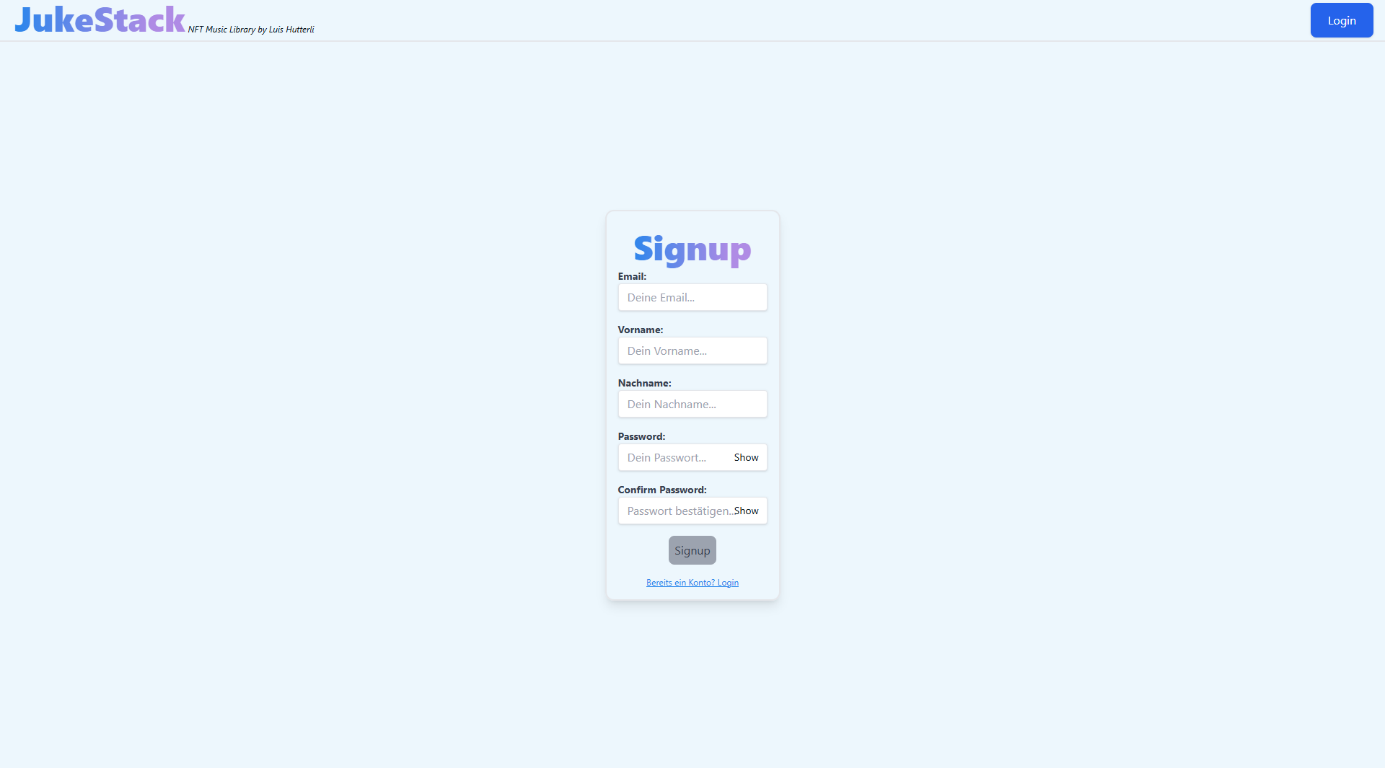
Auf <https://jukestack.ch/>

Login auf <https://jukestack.ch/app/login>

# Benutzeranleitung

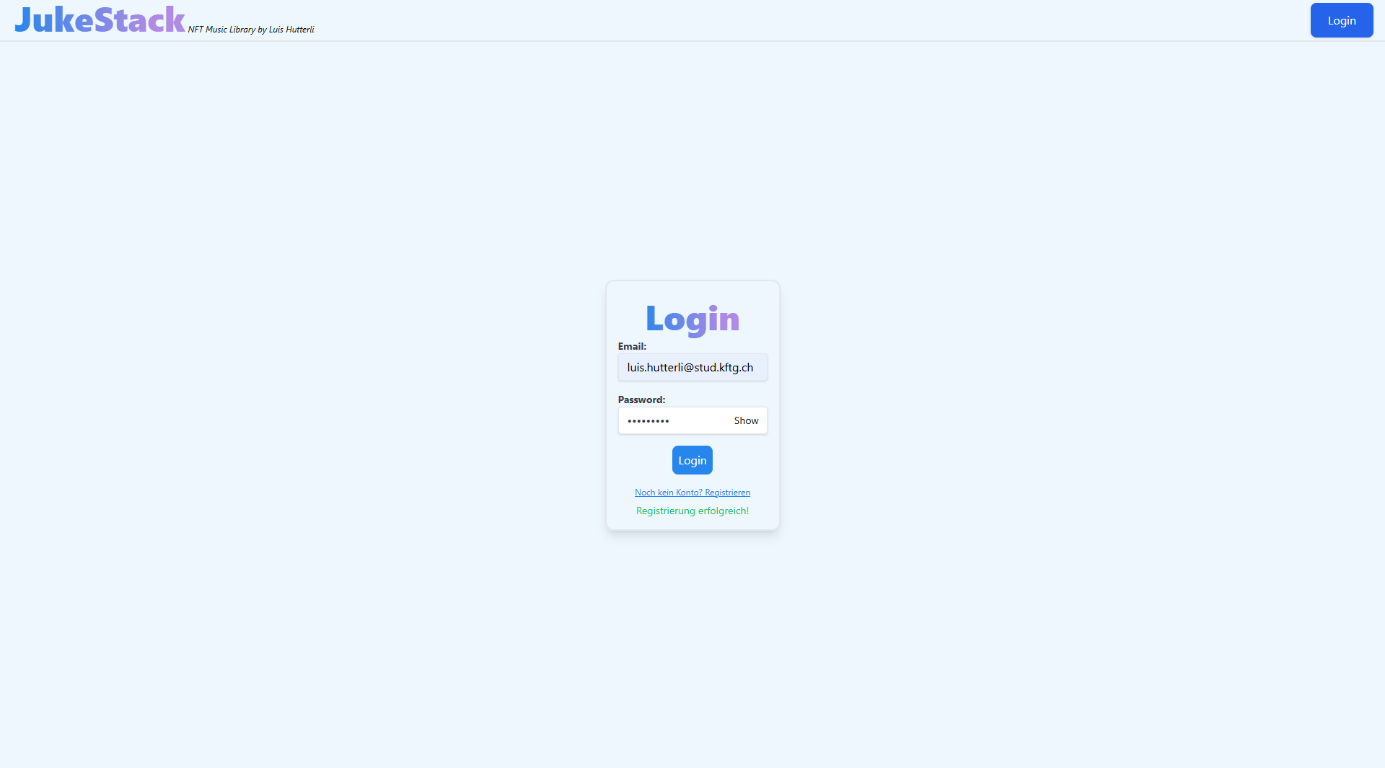
## Account

### Erstellen



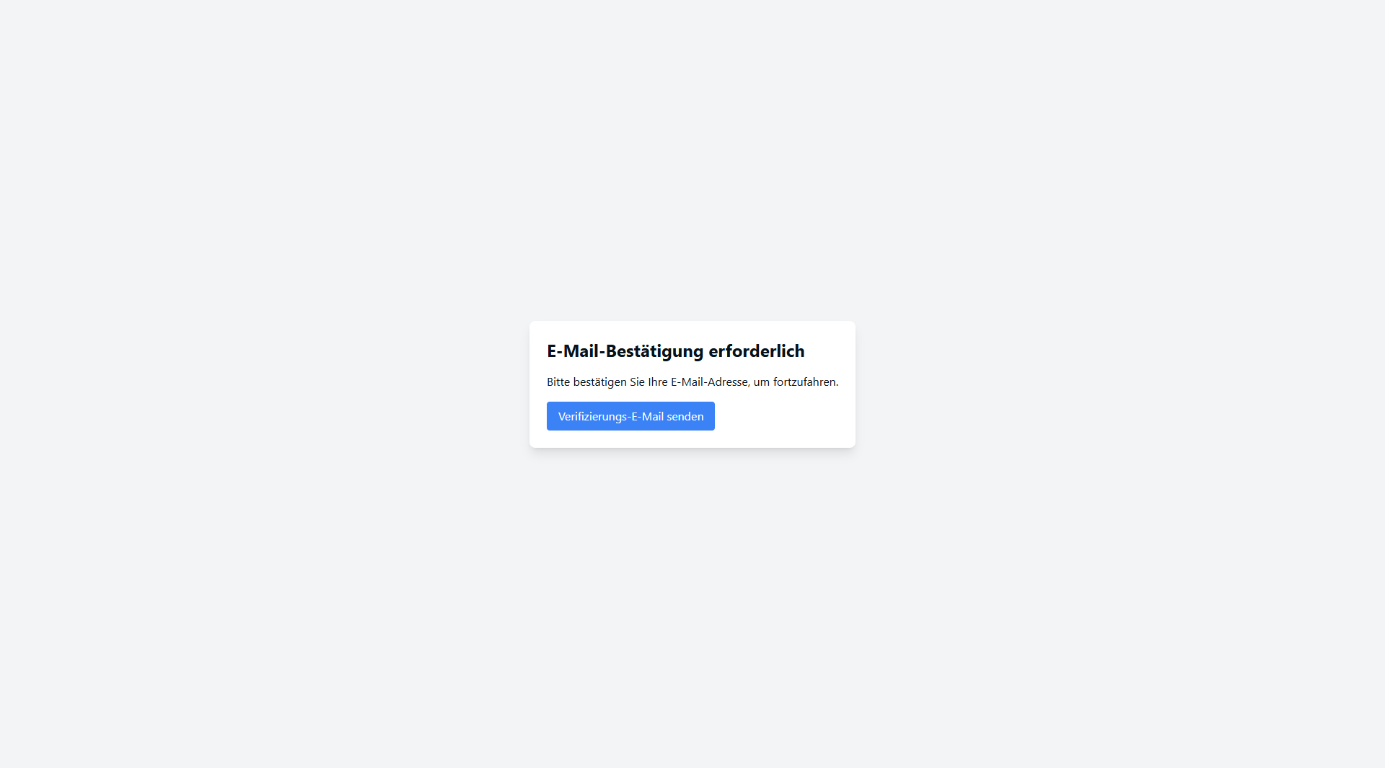
Hier kann der Benutzer ein neues Konto unter Angabe seiner E-Mail, seines Namens und eines Passwortes erstellen. Achtung die E-Mail-Adresse muss gültig sein.

### Anmelden



Hier kann der Benutzer sich mit seinem eben erstellten Konto anmelden.

### E-Mail-Verifizieren



Durch Klicken auf den Knopf kann der Benutzer sich eine Verifizierungs-E-Mail schicken lassen.

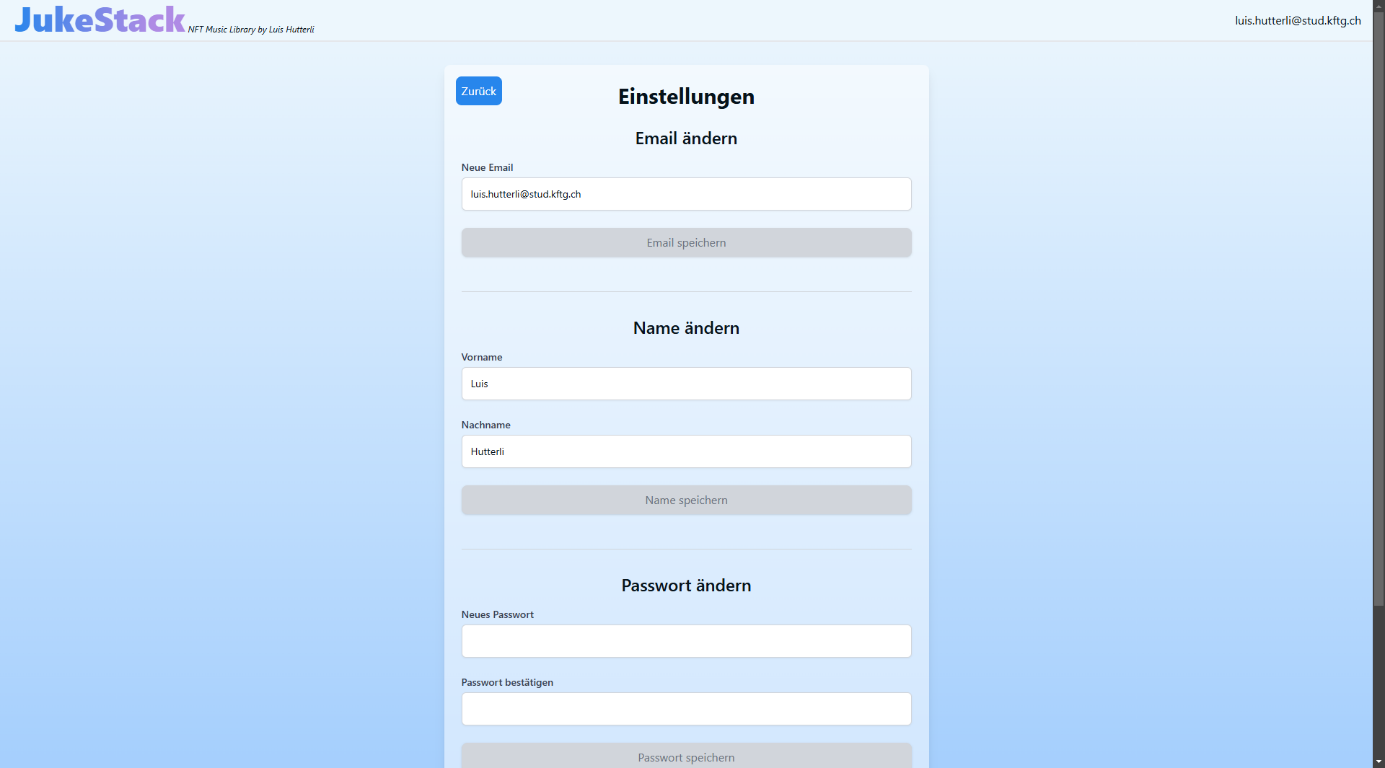
A screenshot of a computer

Description automatically generated

Der Benutzer erhält eine ähnliche E-Mail und kann sich durch Drücken auf den Knopf verifizieren und wird automatisch auf die Hauptseite weitergeleitet.

### Einstellungen

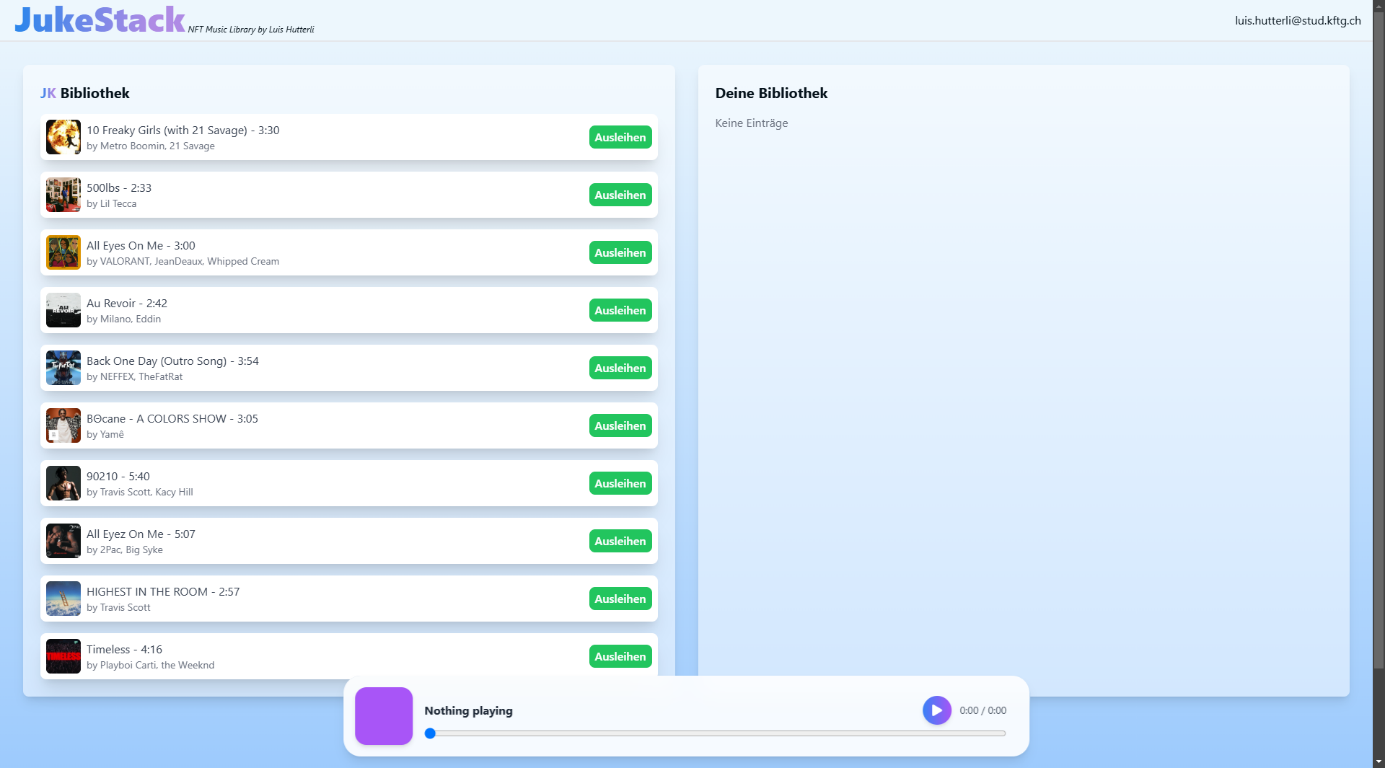
Durch Klicken auf die E-Mail-Adresse in sämtlichen Header kann der Benutzer auf seine Einstellungen zugreifen.



Hier kann der Benutzer die Einstellungen von seinem Konto einsehen und bearbeiten. Achtung bei einer Änderung der E-Mail-Adresse muss der Benutzer diese auch Verifizieren.

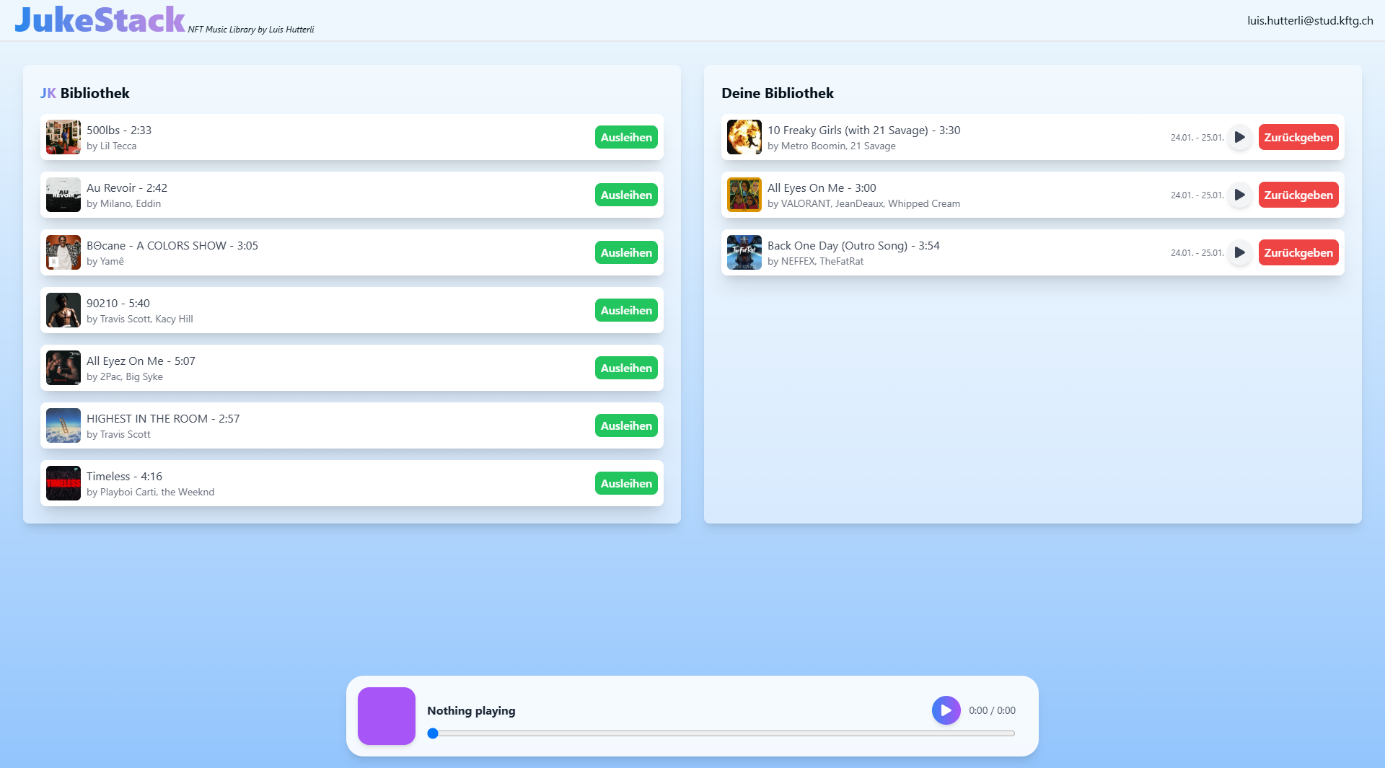
## Hauptseite – Desktop

### Song Ausleihen



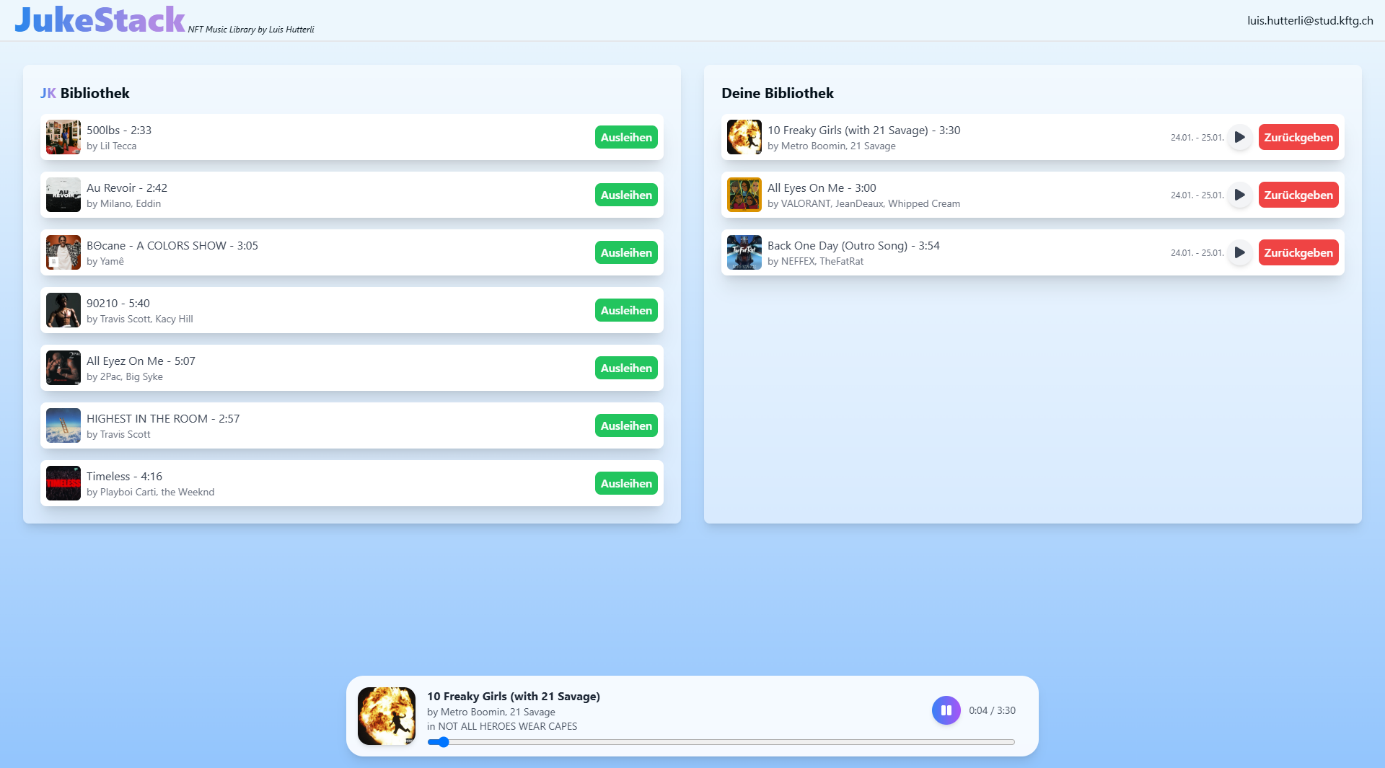
Durch Klicken auf den Knopf «Ausleihen» kann der Benutzer den gewünschten Song ausleihen und in seine Bibliothek hinzufügen.

### Song Zurückgeben



Durch Klicken auf den Roten Knopf «Zurückgeben» kann der Benutzer eine beliebige Ausleihe wieder zurückgeben.

### Song Abspielen / Steuern



Durch das Klicken auf den Abspiel Knopf (Dreieck neben dem Zurückgeben Knopf) kann der Benutzer den gewünschten Song abspielen und er wird in den unteren Musik-Player eingefügt.

Im Musik Player kann der Song beliebig pausiert und wieder abgespielt werden und durch das Benutzen des unteren Schiebereglers kann auch gespult werden.

### Song Steuern über OS-Integration

Wenn der Benutzer wie oben erklärt einen Song anwählt und abspielt, dann wird dieser Song auch einer «mediaSession» beigefügt, wodurch die Steuerung auch durch Betriebssystem oder Browser Komponenten möglich ist.

A screenshot of a video player

Description automatically generated  
Hier ein Beispiel der Chrome Komponente.

A screenshot of a video player

Description automatically generated  
Hier ein Beispiel der Windows Komponente.

## Hauptseite – Mobile

### Song Ausleihen

A screenshot of a music player

Description automatically generatedA screenshot of a music player

Description automatically generatedDurch das Klicken auf das Menu-Knopf (linkes Bild) kann der Benutzer das Menu des Songs öffnen und durch den Unterpunkt «Ausleihen» den Song ausleihen und zu seiner Bibliothek hinzufügen.

### Song Zurückgeben

A screenshot of a music player

Description automatically generatedEin beliebiger Song aus der eigenen Bibliothek kann durch das Öffnen des Song-Menüs (oben beschrieben) und durch Anwählen des Unterpunkts «Zurückgeben» zurückgegeben werden.

### A screenshot of a phone Description automatically generatedSong Abspielen / Steuern

Ein beliebiger Song aus der eigenen Bibliothek kann durch das Öffnen des Song-Menüs (oben beschrieben) und durch Anwählen des Unterpunkts «Abspielen» abgespielt werden und wird in den untenstehenden Musik-Player eingefügt.

Im Musik Player kann der Song beliebig pausiert und wieder abgespielt werden und durch das Benutzen des unteren Schiebereglers kann auch gespult werden.

### Song Steuern über OS-Integration

Wenn der Benutzer wie oben erklärt einen Song anwählt und abspielt, dann wird dieser Song auch einer «mediaSession» beigefügt, wodurch die Steuerung auch durch Betriebssystem oder Browser Komponenten möglich ist.

A screenshot of a phone

Description automatically generatedHier ein Beispiel des Samsung Musik-Players (OneUI 6.1)

Hinweis: So kann ein Song auch abgespielt und gesteuert werden, wenn das Handy gesperrt oder im Standby ist.

## Admin

### Startseite – Liste aller Benutzer