|  |
| --- |
| TH Köln Campus Gummersbach |
| Social Events |
| Praxisprojekt Dokumentation |

|  |
| --- |
| Kevin Röttel (Matr. Nr. 11139133) |

Inhaltsverzeichnis

[Abbildungsverzeichnis 2](#_Toc129723389)

[1. Projektidee 3](#_Toc129723390)

[2. Projektziel 3](#_Toc129723391)

[3. Genutzte Technologien 3](#_Toc129723392)

[3.1 Backend 3](#_Toc129723393)

[3.2 Frontend 5](#_Toc129723394)

[4. Domain Model 6](#_Toc129723395)

[5. Funktionalitäten 7](#_Toc129723396)

[5.1 Login 7](#_Toc129723397)

[5.2 Account Einstellungen 7](#_Toc129723398)

[5.3 Event Erstellen 8](#_Toc129723399)

[5.3.1 Venue Erstellen 8](#_Toc129723400)

[5.3.2 Künstler Erstellen 9](#_Toc129723401)

[5.4 Event Ansicht 9](#_Toc129723402)

[5.4.1 Entfernungsrechnung 10](#_Toc129723403)

[5.5 Event Bearbeiten 10](#_Toc129723404)

[5.6 Locations Page 10](#_Toc129723405)

[5.7 Artist Page 11](#_Toc129723406)

[5.8 Freunde System 11](#_Toc129723407)

[5.9 Search Bar 11](#_Toc129723408)

[5.10 Dashboard 12](#_Toc129723409)

[6. Fazit 14](#_Toc129723410)

[7. Ausblick auf Bachelorarbeit 14](#_Toc129723411)

[Glossar 15](#_Toc129723412)

[Quellenverzeichnis 15](#_Toc129723413)

[Eigenständigkeitserklärung 16](#_Toc129723414)

# Abbildungsverzeichnis

[Abbildung 1: Domain Model 6](#_Toc129717423)

[Abbildung 2: Login Seite 7](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717424)

[Abbildung 3: Account Einstellungs Seite 7](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717425)

[Abbildung 4: Event Erstellungs Formular mit Vorschau 8](#_Toc129717426)

[Abbildung 5: Venue Erstellungs Modal 8](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717427)

[Abbildung 6: Künstler Erstellung Dropdown 9](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717428)

[Abbildung 7: Künstler Erstellung Ähnlichkeits Information 9](#_Toc129717429)

[Abbildung 8: Event Ansicht mit einem Beitrag und Kommentar 9](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717430)

[Abbildung 9: Event Bearbeitung Line-Up ändern 10](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717431)

[Abbildung 10: Venue Seite 10](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717432)

[Abbildung 11: Künstler Seite 11](file:///C:\laragon\www\social-events\Praxisprojekt%20Dokumentation.docx#_Toc129717433)

[Abbildung 12: Freunde Seite 11](#_Toc129717434)

[Abbildung 13: Suchanfrage mit Ergebnissen 11](#_Toc129717435)

[Abbildung 14: Dashboard Seite 12](#_Toc129717436)

# Projektidee

Durch den Abfall von insbesondere jüngeren Nutzern bei Facebook, werden dort ebenfalls Einträge für diverse Veranstaltungen immer weniger. Mit der Entwicklung einer webbasierten Anwendung, soll eine Alternative geschaffen werden, um sich mit anderen Besuchern besser zu vernetzen und um ggf. Fahrgemeinschaften oder Treffen zu organisieren. Die Anwendung soll ein allgemeines Freunde-System beherbergen, sowie das Erstellen von Veranstaltungen möglich machen. Nutzer sollen ihr Interesse bekunden können, zu Veranstaltungen zusagen können sowie mit anderen Nutzern über dedizierte Bereiche kommunizieren können. Ebenfalls soll ein geeigneter Vorschlagsalgorithmus zum Einsatz kommen, um neue Veranstaltungen an die Nutzer heranzutragen. Ebenfalls soll nach einer Möglichkeit gesucht werden, neue Veranstaltungen automatisch von verschiedenen Quellen abzufragen und in der Anwendung einzuspeisen.

# Projektziel

Das Ziel ist eine Plattform über welche Nutzer sich über Veranstaltungen informieren, sowie mit anderen Nutzern über diese Veranstaltungen kommunizieren können. Die Anwendung soll gegen Ende des Projektes ein Minimal Viable Product sein mit Explorativen Features wie dem Vorschlagealgorithmus.

# Genutzte Technologien

## Backend

Als Grundgerüst und Backend für die Webplattform wird Laravel genutzt. Laravel ist ein PHP-Framework welches das MVC-Muster[[1]](#footnote-1) nutzt. Es ist in dieser Applikation für die Datenstruktur, sowie für alle Datenbank-Manipulationen verantwortlich. Ebenfalls stellt Laravel den Einstiegspunkt für die Frontend-Frameworks zur Verfügung, sowie die Login- und Register-Seiten. Auch werden über Laravel alle internen Web-Routen definiert.

Ein Model in Laravel ist im Code eine Klasse, welche hauptsächlich die Datenbeziehungen mit anderen Models festlegt. So wird z.B. im Model Artist, die n:m-Beziehung namentlich als *lineup* zu Events festgelegt:

class Artist extends Model

{

    public function events() {

        return $this->belongsToMany(Event::class)->as('lineup');

    }

}

Ein Model kann allerdings nicht allein die Datenbank erzeugen. Hierfür werden Migration-Dateien benötigt, in welchen die Datenstruktur festgelegt wird, welche dann durch den Shell-Befehl „php artisan migrate:fresh“ erstellt wird. Dort wird der Tabellenname festgelegt, sowie alle Attribute mit Namen und Typen sowie Fremdschlüsseln und Null-Fähigkeit.

public function up()

{

    Schema::create('events', function (Blueprint $table) {

        $table->id();

        $table->string('name');

        $table->text('description')->nullable();

        $table->longText('flyer')->nullable();

        $table->date('date');

        $table->string('doors')->nullable();

        $table->string('begin');

        $table->string('ticketLink')->nullable();

        $table->unsignedBigInteger('location\_id');

        $table->foreign('location\_id')->references('id')->on('locations');

        $table->timestamps();

    });

}

In den Controllern der Models werden die üblichen CRUD-Operationen implementiert. Innerhalb der jeweiligen Funktionen findet auch eine Validierung der Nutzergenerierten Daten statt. Hier ein Beispiel für das Erstellen eines Künstlers durch eine Web-Route:

public function createArtist(Request $request) {

    $request = $request->validate([

        'name' => 'required|unique:artists|string'

    ]);

    $artist = Artist::create([

        'name' => $request['name']

    ]);

    return $artist;

}

Web-Routen sind in Laravel sind die Routen, welche intern von Usern genutzt werden. Damit eine inhaltliche Trennung zu möglichen API-Routen besteht, werden diese zudem immer in zwei verschiedenen Dateien gespeichert.[[2]](#footnote-2)

Eine Web-Route besteht aus einer Route-Methode wie *post* oder *get*, der URL und möglichen URL-Parametern, sowie dem Namen der Funktion und einem Controller-Verweiß, welche die Aktion durchführen soll.

Route::post('/artists', [ArtistController::class, 'createArtist']);

## Frontend

Für den Großteil an Frontend Interaktionen ist Vue verantwortlich. Vue ist ein JavaScript Framework für Single-Page-Webanwendungen[[3]](#footnote-3). Eine Vue-Applikation besteht aus verschiedenen Components, welche mehrfach getrennt voneinander wiederverwendet werden können. Jeder Component besteht aus einem Template- und einem Script-Teil.

Das Template wird zu großem Teil aus den Standard-HTML-Elementen geschrieben. Mit Vue kommen allerdings auch verschiedene Kontrollstrukturen wie Schleifen und Bedingungen hinzu. Dadurch lässt sich der Inhalt der Seite dynamisch erstellen, und kann auch Client-seitig direkt erweitert werden, ohne die Seite neuzuladen. Neue Components können zudem einfach als HTML-Tag im Template eingebunden werden.

Im Script-Teil des Components werden die benötigten Daten bearbeitet und geladen, welche nötig sind um dem Nutzer den korrekten Content anzuzeigen. Bspw. werden hier über Web-Routen aus der Datenbank abgefragt oder auch das Speichern von neuen Datensätzen angestoßen.

Um auf der Single-Page-Webanwendung auch Browserfunktionalitäten zur Verfügung zu stellen, wurde Vue-Router genutzt. Vue-Router ist ein Add-on um verschiedene Components je nach URL zu laden. Dafür muss ein Router-Objekt erstellt werden, welches Components zu einer URL zuweist und eine Browser-History erstellt. Somit kann auch die Zurück-Funktion des Browsers genutzt werden, und beim neuladen der Seite wird nicht automatisch auf die Startseite zurückgeleitet.

Da mehrere Vue-Components die gleichen Daten nutzen müssen, wird Pinia verwendet. Pinia ist ebenfalls ein Add-on, um seitenübergreifend Daten in einem virtuellen Storage zu speichern. Hierdurch müssen nicht bei jedem Seitenwechsel alle Daten neugeladen werden. Somit werden nur bei einem Page-Refresh alle Daten aus der Datenbank abgefragt.

Damit alles etwas angenehmer aussieht, wird Bootstrap verwendet, um mit Hilfe von CSS-Klassen Elemente besser hervorzuheben. Zudem können durch Bootstrap Elemente wie z.B. Modale wesentlich schneller erstellt werden.

# Domain Model

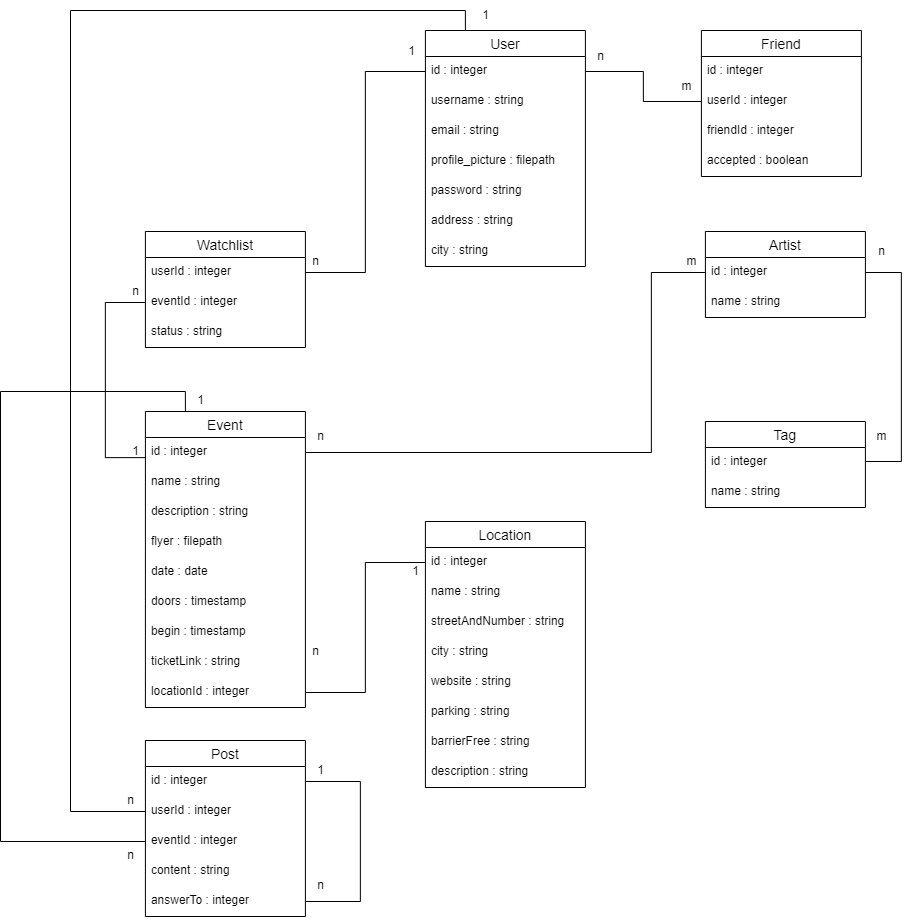


Abbildung : Domain Model

# Funktionalitäten

## Login

Die Login-Seite ist die allgemeine Startseite für nicht angemeldete Benutzer. Hierrüber können sich neue Nutzer registrieren sowie bestehende einloggen.

Abbildung : Login Seite

Bei fehlerhaften Login-Versuchen wird dem Nutzer mitgeteilt das seine Anmeldedaten nicht korrekt sind. Ebenfalls werden bei der Registrierung Informationen angezeigt, sollte eine angegebene E-Mail oder ein Username bereits im System vorhanden sein.



## Account Einstellungen

Abbildung : Account Einstellungs Seite

Durch die Account Seite lassen sich die Informationen des Nutzers bearbeiten. Dazu zählen das Ändern des Usernames, der E-Mail sowie das Passwort und das Hochladen eines Profilbildes.

Außerdem können Nutzer ihren Wohnort mit Adresse festlegen. Diese Angaben sind freiwillig, und dienen rein für die Berechnung von Entfernungen zu Veranstaltungsorten.



## Event Erstellen

Über die Navbar[[4]](#footnote-4) lässt sich die Seite für die Eventerstellung öffnen. Diese besteht aus dem Formular und einer Vorschau der Eventansicht.

Das Formular besteht aus neun Eingabefeldern, welche die wichtigsten Informationen für eine Veranstaltung abdecken sollen. Verpflichtende Angaben sind der Name der Veranstaltung, das Datum mit einer beginnenden Uhrzeit, sowie einen Veranstaltungsort und Künstler.

Zusätzliche Angaben sind eine Beschreibung, ein Flyer, sowie einen Einlasszeitpunkt und einen Ticket-Link.

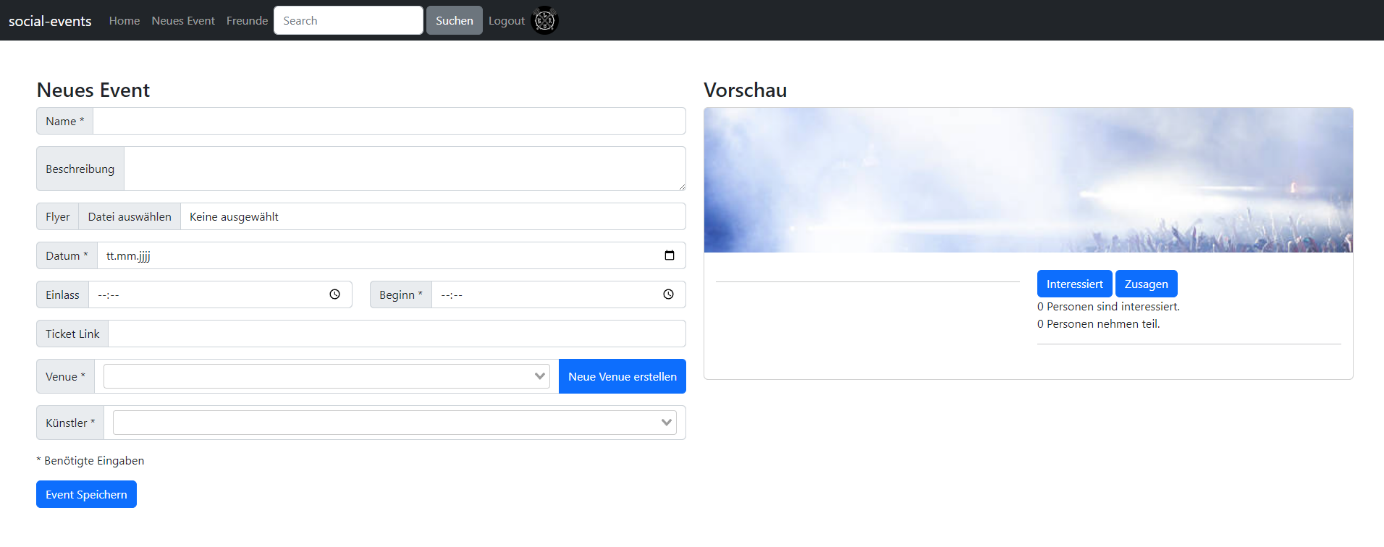


Abbildung : Event Erstellungs Formular mit Vorschau

### Venue Erstellen

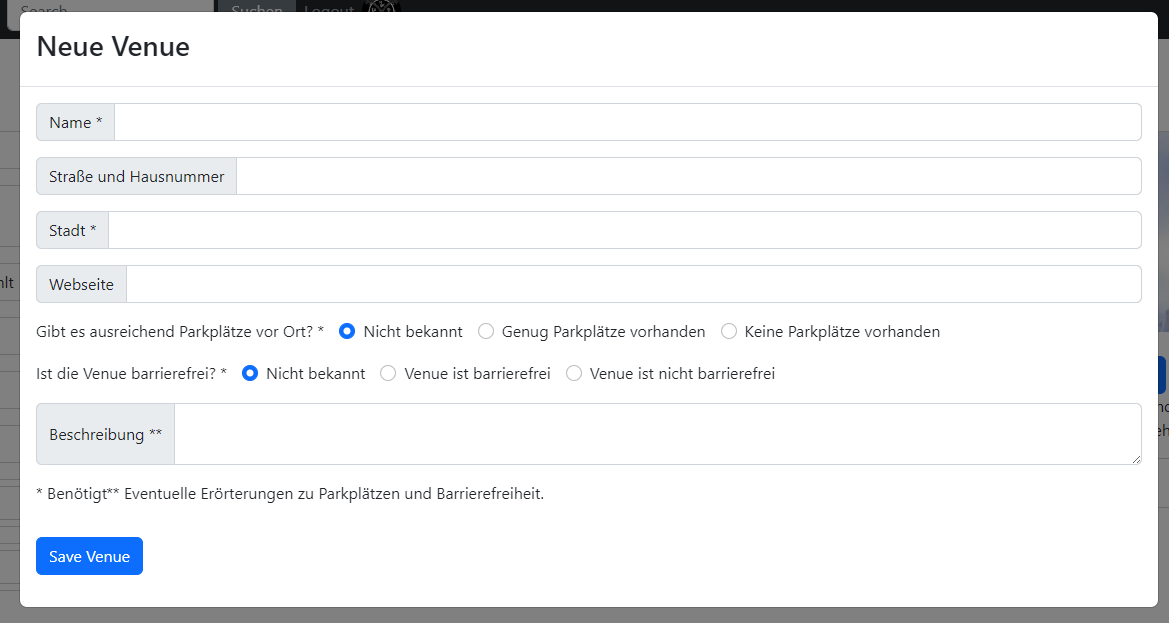
Innerhalb des Formulars lassen sich zudem neue Veranstaltungsorte anlegen. Dies war zuerst ein losgelöster Teil der Eventerstellung, wurde dann aber so integriert das eine Eventerstellung nicht unterbrochen werden muss, um einen neuen Veranstaltungsort hinzuzufügen.

Abbildung : Venue Erstellungs Modal

Ein Veranstaltungsort (ff. Venue), besteht aus dem Namen, einer Adresse, dem Ort und einer optionalen Webseite. Zudem wird abgefragt ob es ausreichend Parkplätze vor Ort gibt, sowie der Zustand der Barrierefreiheit innerhalb der Venue. Diese können dann auch über eine Beschreibung weiter erläutert werden.

### Künstler Erstellen

Abbildung : Künstler Erstellung Dropdown

Für eine Veranstaltung müssen auch Künstler ausgewählt werden. Dies funktioniert über ein Text-Eingabefeld, welches mehrere Werte gleichzeitig tragen kann. Die ausgewählten Künstler können dann auch per Klick wieder entfernt werden. Gleichzeitig funktioniert dieses Eingabefeld auch als Suche, um vorhandene Künstler zu finden, und direkt auszuwählen.

Sollte ein Künstler noch nicht in der Datenbank vorhanden sein, so muss der Nutzer den vollständigen Namen eingeben und per Eingabe hinzufügen. Hier wird automatisch eine Benachrichtigung angezeigt, dass die Korrektheit des Künstlers überprüft werden soll. Zudem wird auch eine weitere Information angezeigt, sollte der Name sehr ähnlich zu einem anderen Künstler sein.

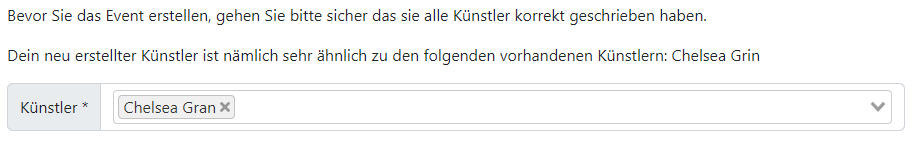


Abbildung : Künstler Erstellung Ähnlichkeits Information

## Event Ansicht

Durch das Anklicken eines kleinen Event-Teasers auf dem Dashboard, oder nach der Erstellung, wird das Event als eigenständige Seite geöffnet. Dort sind alle hinterlegten Informationen abzurufen, inklusive der ungefähren Entfernung vom Nutzer zur Venue, sowie der Anzahl an Interessens-bekundungen sowie Zusagen. Ebenfalls sind Verlinkungen der Venue und den einzelnen Künstlern vorhanden, welche zu den jeweiligen Seiten führen.

Abbildung : Event Ansicht mit einem Beitrag und Kommentar

Es gibt auch die Möglichkeit Beiträge zu einem Event zu posten. Dies ist am unteren Ende der Eventinformationen möglich. Dies kann genutzt werden, um eventuelle Fahrgemeinschaften zu bilden, oder weitere Informationen zum Event hinzuzufügen (z.B. Zeitpläne), ohne das Event selbst Bearbeiten zu müssen. Auch können zu jedem Beitrag Kommentare verfasst werden.

### Entfernungsrechnung

Für den Fall, dass der Nutzer seine Adresse im System hinterlegt hat, wird innerhalb des Events die ungefähr kürzeste Route zur Venue berechnet. Dies setzt ebenfalls voraus, dass die Venue namentlich auf den genutzten Karten hinterlegt ist, oder ebenfalls eine vollständige Adresse hinterlegt hat.

Zunächst müssen für die korrekte Berechnung die Längen- und Breitengrade des Nutzers und der Venue herausgefunden werden. Dies geschieht über die Nominatim-API, was grundsätzlich eine Search Engine für OpenStreetMap[[5]](#footnote-5) ist.

Mit diesen Daten wird dann über die API von OSRM[[6]](#footnote-6) eine Route abgefragt, wovon nur die Distanz benötigt wird, um sie schließlich dem Nutzer anzuzeigen.

Dies soll allgemein dazu dienen, dem Nutzer eine direkte Information über die Entfernung zu einer ggf. unbekannten Venue zu geben, ohne noch extra über Google Maps die Route zu überprüfen.

## Ein Bild, das Text enthält. Automatisch generierte BeschreibungEvent Bearbeiten

Abbildung : Event Bearbeitung Line-Up ändern

Die Eventbearbeitung ist fast identisch mit dem Aufbau der Erstellung. Nur das Ändern des Line-Ups ist in ein Modal ausgelagert worden, um schnelle Fehlklicke zu vermeiden.

## Locations Page

Jede Location kann auch als eigene Seite angezeigt werden, um weitere Infos zu erhalten. Dort können ebenfalls Details bearbeitet werden, um z.B. auf neue Parkmöglichkeiten hinzuweisen.

Ebenfalls werden hier alle bevorstehenden und vergangene Veranstaltungen in einer Liste angezeigt.

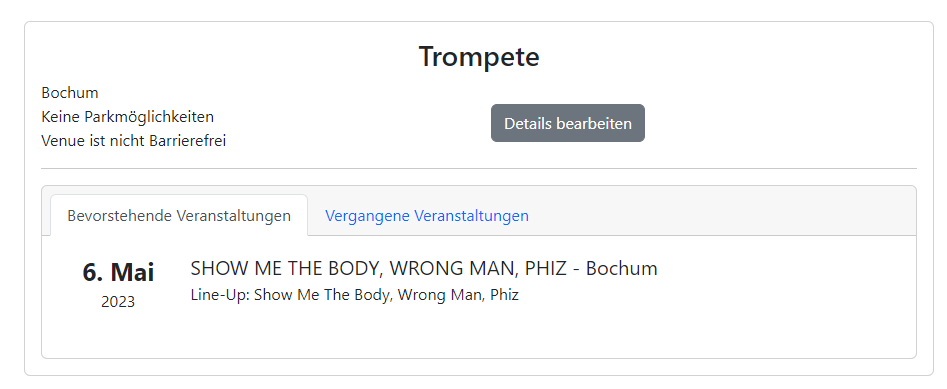


Abbildung : Venue Seite

## Artist Page

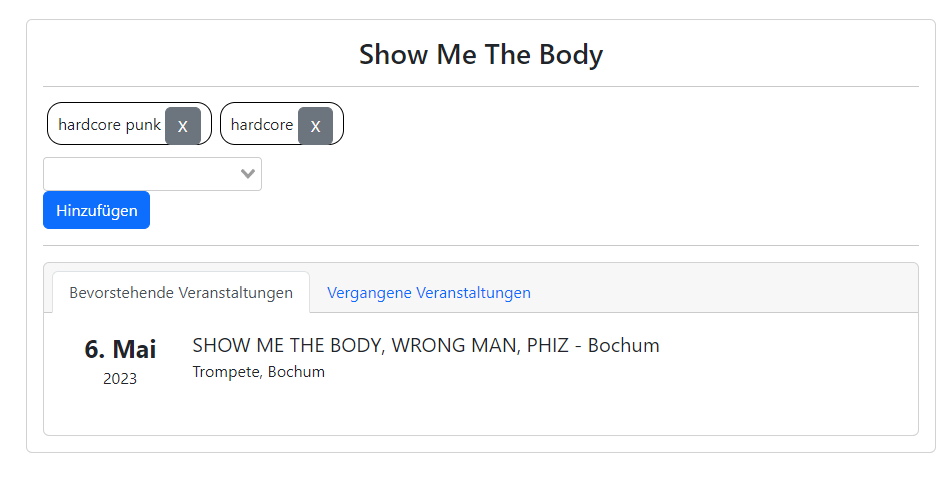
Auch jeder Künstler wird auf einer eigenen Seite angezeigt. Hier sind ebenfalls alle bevorstehenden und vergangenen Veranstaltungen hinterlegt.

Abbildung : Künstler Seite

Zudem besitzen Künstler ein Tag-System. Hier können beliebig beschreibende Tags hinzugefügt und entfernt werden. Diese Tags dienen den Vorschlagssystemen auf dem Dashboard, und um Nutzern ein Gefühl zu geben, was sie bei dem Künstler erwartet.

## Freunde System

Ein rudimentäres System für Nutzerfreundschaften ist ebenfalls vorhanden. Dieses beschränkt sich zurzeit auf das Senden, Annehmen und Löschen von Freundschaftsanfragen.

Ebenfalls ist auf dem Dashboard ein Abschnitt vorhanden für Events an denen Freunde interessiert sind oder zugesagt haben.

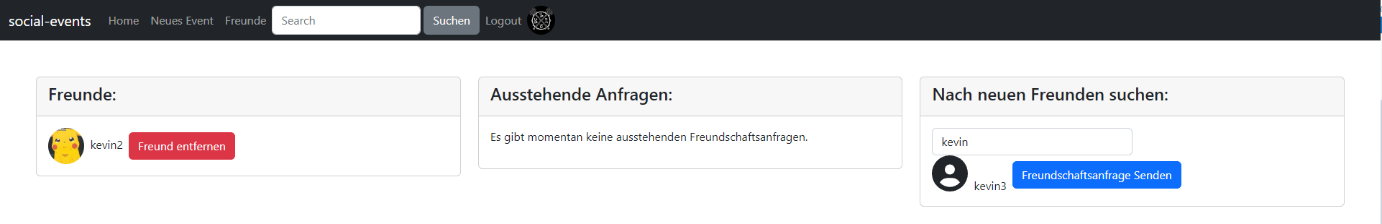


Abbildung : Freunde Seite

## Search Bar

Über die Navbar lassen sich Suchanfragen starten. Diese Suchanfragen vergleichen den String mit dem Namen von Künstlern, Events und Venues. Übereinstimmungen werden dann als Liste auf einer Ergebnis-Seite angezeigt, und verlinken auf die jeweils eigenständigen Seiten der Künstler, Events und Venues.

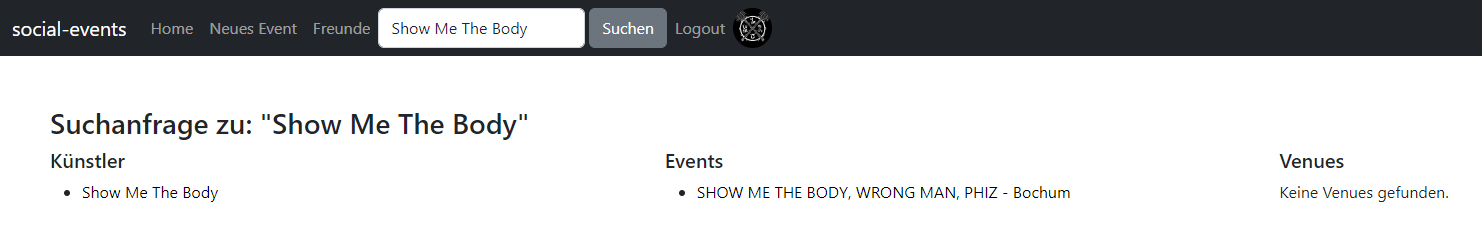


Abbildung : Suchanfrage mit Ergebnissen

## Dashboard

Das Dashboard ist die allgemeine Startseite für eingeloggte Nutzer. Hier sind die Events der Nutzer-Watchlist zu finden, sowie Vorschläge, die darauf basieren.

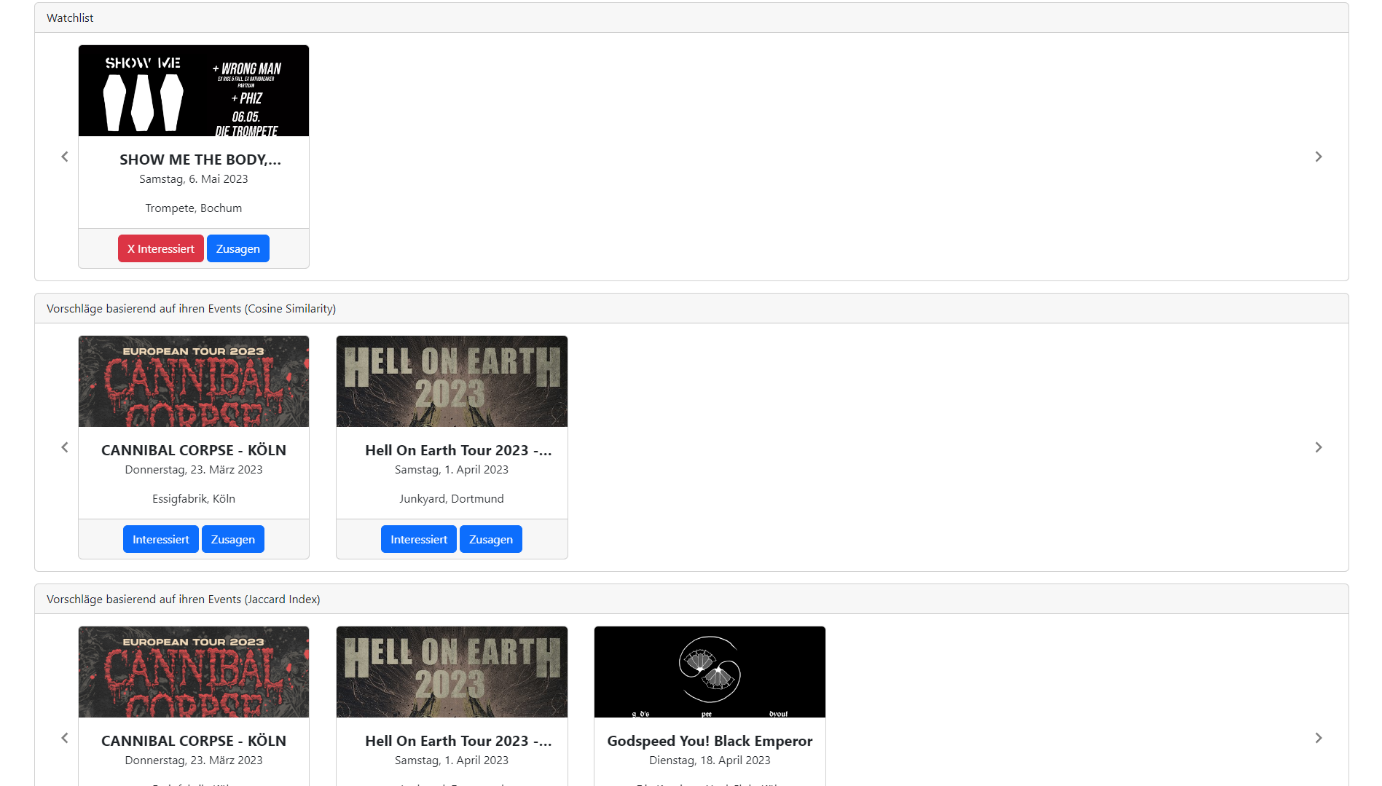


Abbildung : Dashboard Seite

Die Vorschläge werden zurzeit mit Hilfe der Cosine Similarityoder des Jaccard Index generiert. Dafür werden zu jedem Event, alle Tags der Künstler in ein gemeinsames Array gesteckt. Dieses Array wird dann von den Algorithmen genutzt um Vorschläge zu generieren.

Die Kosinus-Ähnlichkeit berechnet die Ähnlichkeit zweier Vektoren. Dies geschieht durch die Berechnung des Kosinus-Winkels zwischen den beiden Vektoren. Damit können wir auch berechnen wie ähnlich sich zwei Dokumente oder auch Texte sind anhand von Häufigkeits-Vektoren. [[7]](#footnote-7)

Dies könnte folgendermaßen in Code aussehen[[8]](#footnote-8):  
Die Funktion *cosineSimilarity* ist in zwei weitere Funktionen gegliedert. Einmal das Skalarprodukt (*dotProduct*) als Zähler und zweimal die Größe (*magnitude*) der Vektoren als Nenner.

const dotProduct = (a, b) => {

    let dotProduct = 0;

    for (let i = 0; i < a.length; i++) {

        dotProduct += a[i] \* b[i];

    }

    return dotProduct;

}

const magnitude = (vector) => {

    let sum = 0;

    for (let i = 0; i < vector.length; i++) {

        sum += vector[i] \* vector[i];

    }

    return Math.sqrt(sum);

}

const cosineSimilarity = (a, b) => {

    return dotProduct(a, b) / (magnitude(a) \* magnitude(b));

}

Der Jaccard Index beschreibt die Ähnlichkeit von zwei Gruppen. Definiert wird dies durch die Größe der Schnittmenge geteilt durch die Größe der Vereinigung der jeweiligen Gruppen. Je näher der Index an 1 liegt, desto höher ist die Ähnlichkeit der Mengen. [[9]](#footnote-9)

Würde man dies nun in JavaScript umsetzen, könnte dies z.B. so aussehen:  
Zunächst werden die Überschneidungen gefiltert, und danach die Berechnung aus der obigen Formel angewendet.

const jaccardIndex = (s1, s2) => {  
 let intersection = new Set([...s1].filter(i => s2.has(i)));  
 return intersection.size / (s1.size + s2.size - intersection.size);  
}

# Fazit

Das Projekt ist nun zum Ende ein Minimal Viable Product mit den wichtigsten Funktionalitäten realisiert. Dies umfasst die Eventerstellung und -bearbeitung, inklusive der Erstellung von Venues und Künstlern. Ebenfalls wurden explorative Features implementiert, wie Vorschlagsalgorithmen.

Die Plattform ist somit in einem test- und erweiterbaren Zustand. Erweiterungen würde unter anderem folgende Punkte enthalten:

* Mobile Optimierung und interessanteres Design
* Web Accessibility/Behindertengerechtes Internet
* Freunde-System präsenter machen (z.B. Privat-Chat, Posts von Freunden hervorheben)
* Benachrichtigungen bei neuen Event-Zusagen von Freunden oder Posts zu Interessierten Events
* Dateneffizienteres Speichern und Laden von Flyern und Profilbildern
* Events absagen
* Mehrtägige Events
* Event-Grabbing von z.B. Tour Flyern oder Social-Media Posts
* Bewertung von Künstler-Tags

# Ausblick auf Bachelorarbeit

Für die Bachelorarbeit soll der Fokus auf die Vorschlagssysteme gelegt werden. Bislang wurden auf der Plattform nur Content-basierte Algorithmen genutzt. Eine Möglichkeit wäre z.B. ein Nutzer-basiertes oder Hybrides Vorschlagssystem zu implementieren. Diese Systeme könnten miteinander verglichen werden, wie komplex die Implementierung ist und wie gut jeweils Vorschläge generiert werden. Dies könnte auch mit freiwilligen Nutzern gemacht werden, um somit die Eingabe von vielen Sample-Daten zu vermeiden. Durch die Testung mit mehrere Nutzern, können ebenfalls noch weitere Fehler gefunden werden.

# Glossar

|  |  |
| --- | --- |
| MVC-Muster | Model-View-Controller-Muster  Entwurfsmuster, erleichtert Erweiterungen und Änderungen |
| Single-Page-Anwendung | Webanwendung bestehend aus einzelner Root-Datei, deren Inhalt dynamisch nachgeladen wird |
| Navbar | Navigations Bar/Leiste  Meist am oberen Rand des Programmes zu finden |
| OpenStreetMap | Open Source Alternative zu Google Maps  <https://www.openstreetmap.de> |
| OSRM | Open Source Routing Machine  API für Routen Generierung zwischen mehreren Punkten  <https://project-osrm.org> |

# Quellenverzeichnis

Building a Text Similarity checker using Cosine Similarity in JavaScript and HTML  
Eyuel Berga Woldemichael   
<https://medium.com/analytics-vidhya/building-a-text-similarity-checker-using-cosine-similarity-in-javascript-and-html-75722d485703>

How to Build a Simle Movie Recommender System with Tags  
Johnson Kuan  
<https://towardsdatascience.com/how-to-build-a-simple-movie-recommender-system-with-tags-b9ab5cb3b616>

Laravel Documentation – The Default Route Files  
<https://laravel.com/docs/10.x/routing#the-default-route-files>

# Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Dies gilt auch für Quellen aus eigenen Arbeiten.

Ich versichere, dass ich diese Arbeit oder nicht zitierte Teile daraus vorher nicht in einem anderen Prüfungsverfahren eingereicht habe.

Mir ist bekannt, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs mittels einer Plagiatserkennungssoftware auf ungekennzeichnete Übernahme von fremdem geistigem Eigentum überprüft werden kann.

Lindlar, 14.03.2023 Kevin Röttel

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ort, den Unterschrift

1. Siehe Glossar [↑](#footnote-ref-1)
2. Laravel Documentation – The Default Route Files [↑](#footnote-ref-2)
3. Siehe Glossar [↑](#footnote-ref-3)
4. Siehe Glossar [↑](#footnote-ref-4)
5. Siehe Glossar [↑](#footnote-ref-5)
6. Siehe Glossar [↑](#footnote-ref-6)
7. Building a Text Similarity checker using Cosine Similarity in JavaScript and HTML [↑](#footnote-ref-7)
8. Code nach: Building a Text Similarity checker using Cosine Similarity in JavaScript and HTML [↑](#footnote-ref-8)
9. How to Build a Simle Movie Recommender System with Tags [↑](#footnote-ref-9)