

Höhere Lehranstalt für Wirtschaftsingenieurwesen  
Ausbildungsschwerpunkt: Betriebsinformatik

# DIPLOMARBEIT

## Eventplaner

### Konzeption und Entwicklung eines Veranstaltungsplaners als webbasierte Lösung

**Ausgeführt im Schuljahr 2015/16 von:**

Simon Ellensohn	10aWI
Martin Leimser	10aWI
Julian Martin	10aWI

**Betreuer:**

Dipl.-Ing. Diethard Kaufmann  
Benjamin Meier, MSc

# Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbständig und ohne fremde Hilfe verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

Diplomanden:

Datum und Unterschrift:

Simon Ellensohn

---

Martin Leimser

---

Julian Martin

---

# Abstract

The aim of the diploma thesis *Eventplaner* was to develop a web-based planner that shows the user current events in their nearer proximity. The thesis was developed in cooperation with TOWA GmbH based in Bregenz.

At first ToWa received the job from Heidelberg to develop such an event planner for the city. Said project was on hold, thus ToWa offered us that we develop the project as our diploma thesis. While working we decided to make the event planner platform-based so that it can also be used in e.g. Vorarlberg.

It was particularly exciting for us to use our learned programming skills in the context of a big project and in the process expand them greatly.

It is planned that ToWa takes over the source code of the project and possibly uses it as base or for reference in future projects.

# Vorwort & Danksagung

Die Diplomarbeit Eventplaner hatte das Ziel, einen webbasierten Planer zu entwickeln, der dem Benutzer aktuelle Events in seiner näheren Umgebung anzeigt. Die Arbeit wurde in Zusammenarbeit mit dem Unternehmen ToWa GmbH mit Sitz in Bregenz entwickelt.

Ursprünglich bekam ToWa von Heidelberg den Auftrag, einen Eventplaner für die Stadt zu entwickeln. Da dieses Projekt im Stillstand war, bot ToWa uns an, dass wir das Projekt als unsere Diplomarbeit entwickeln. Während der Arbeit sind wir zum Entschluss gekommen, den Eventplaner als Plattform umzusetzen, damit dieser auch z.B. in Vorarlberg eingesetzt werden kann.

Besonders spannend für uns war, unsere gelernten Programmierfähigkeiten im Rahmen eines großen Projektes einzusetzen und gleichzeitig auch zu erweitern.

Geplant ist es, dass ToWa den Source Code des Projekts übernimmt und ihn eventuell bei zukünftigen Projekten als Basis oder Referenz verwendet.

Während wir an diesem Projekt arbeiteten, hatten wir mit vielen verschiedenen Personen zu tun gehabt. An dieser Stelle möchten wir uns für deren Wille zur Kooperation und Hilfeleistung bedanken.

Ein besonderer Dank gilt unseren Projektbetreuern, Prof. Diethard Kaufmann und Herr Benjamin Meier seitens der ToWa GmbH, die uns stets bei Schwierigkeiten zur Seite standen.

Nicht zuletzt gilt unser Dank Frau Elena Zabrodina, die aus unseren *Mockups* ein ausgezeichnetes Design erstellte und uns stets mit den benötigten Vorlagen bediente, was die Umsetzung für uns stark erleichterte.

# Hinweis zur geschlechtsneutralen Formulierung

Um eine geschlechtsneutrale als auch eine gut lesbare Formulierung zu gewährleisten, werden für geschlechtsspezifische Wörter Rollen definiert, die durchgehend als männlich bzw. weiblich verwendet werden.

In folgender Tabelle werden die verwendeten Rollen aufgelistet:

Männlich	Weiblich
Benutzer	Administratorin
Moderator	Kundin

# Hinweis zur Textformatierung

Verschiedene Textformatierungen in diesem Dokument werden einheitlich verwendet.

In folgender Tabelle wird erklärt, wofür verschiedene Formatierungen verwendet werden.

Formatierung	Erklärung
<i>Kursiv</i>	Begriffe, die im Glossar erklärt werden.
Abkürzung (Abk.)	Abkürzungen werden bei erster Verwendung ausgeschrieben, gefolgt von der Abkürzung in Klammer. Abkürzungen sind im Abkürzungsverzeichnis aufgelistet.
“Anführungszeichen”	Anführungszeichen werden für technische Ausdrücke wie Programmfunktionen oder Attribute verwendet.

# Inhaltsverzeichnis

1	Impressum .....	1
1.1	Projektteam.....	1
1.1.1	Simon Ellensohn.....	1
1.1.2	Martin Leimser .....	1
1.1.3	Julian Martin .....	2
1.2	Projektbetreuer .....	2
1.2.1	Dipl.-Ing. Diethard Kaufmann.....	2
1.2.2	Benjamin Meier, MSc.....	3
1.3	Projektmitarbeiterin .....	3
1.3.1	Elena Zabrodina .....	3
2	ToWa GmbH.....	4
3	Projektmanagement.....	5
3.1	Allgemein .....	5
3.2	Projektauftrag .....	6
3.3	Projektzieleplan .....	7
3.3.1	Hauptziele.....	7
3.3.2	Nebenziele.....	8
3.3.3	Nichtziele .....	8
3.4	Projektorganisation und -organigramm .....	9
3.5	Projektstrukturplan .....	11
3.6	Projektterminplan .....	13
3.7	Meilensteinplan .....	14
3.8	Risikoanalyse.....	15
3.9	Projektabschlussbericht .....	16
3.9.1	Projektverlauf.....	16

3.9.2	Zielerreichung .....	16
3.9.3	Ausblick .....	16
4	Analysephase .....	17
4.1	Evaluierung Eventplaner .....	18
4.1.1	Events.at .....	18
4.1.2	Oeticket .....	18
4.1.3	ländleTICKET .....	18
4.1.4	Ticketmaster .....	18
4.1.5	Facebook .....	18
4.1.6	Zusammenfassung .....	19
4.2	Use-Case Diagramm .....	20
5	Einarbeitung .....	21
5.1	WordPress .....	21
5.2	Laravel .....	22
5.3	Vergleich zwischen WordPress und Laravel .....	23
5.4	Entwicklungsumgebung .....	25
6	Design .....	29
6.1	Mockups .....	30
6.1.1	Startseite .....	30
6.1.2	Eventansicht .....	31
6.2	Designentwurf .....	32
6.3	Designimplementierung .....	33
6.4	Umgesetztes Design .....	35
6.4.1	Anmeldeseite .....	35
6.4.2	Events .....	36
6.4.3	Eventansicht .....	37
6.4.4	Einstellungen .....	38



7	Entwicklung / Umsetzung.....	39
7.1	Allgemein .....	39
7.1.1	Ordnerstruktur.....	39
7.2	Artisan.....	43
7.3	Blade.....	44
7.4	Datenbank - MySQL .....	45
7.4.1	Verwendung von MySQL.....	45
7.4.2	Entity Relation-Modell.....	46
7.5	Benutzerverwaltung .....	49
7.5.1	Autorisierung.....	49
7.5.2	Registrierung .....	50
7.5.3	Authentifizierung .....	52
7.5.4	Passwort vergessen .....	53
7.5.5	Interessenverwaltung.....	53
7.5.6	Profil bearbeiten.....	53
7.6	Eventverwaltung .....	54
7.6.1	Erstellen von Events .....	54
7.6.2	Bearbeiten von Events.....	58
7.6.3	Löschen von Events .....	58
7.7	Locationverwaltung .....	59
7.7.1	Erstellen von Locations.....	59
7.7.2	Bearbeiten von Locations .....	59
7.7.3	Löschen von Locations .....	60
7.8	Künstlerverwaltung .....	61
7.8.1	Erstellen von Künstlern.....	61
7.8.2	Bearbeiten von Künstlern.....	61
7.8.3	Löschen von Künstlern .....	61

7.9	Bilderverwaltung .....	62
7.10	Anfragenverwaltung .....	63
7.11	Suche und Filter .....	65
7.11.1	Bearbeiten von Künstlern .....	65
7.11.2	Events .....	66
7.11.3	Künstler .....	68
7.11.4	Locations .....	69
7.12	Newsletter .....	70
7.13	Asynchrone Datenübertragung mit AJAX .....	71
8	Testing .....	73
8.1	Automatisierte Tests .....	73
8.1.1	White-Box-Tests .....	74
8.1.2	Black-Box-Tests .....	75
8.2	Manuelle Tests .....	77
8.2.1	UATs .....	77
9	Fazit .....	78
10	Hilfsmittel .....	79
11	Glossar .....	87
12	Abbildungsverzeichnis .....	91
13	Tabellenverzeichnis .....	94
14	Abkürzungsverzeichnis .....	95
15	Literaturverzeichnis .....	96

# 1 Impressum

Das Impressum des Projektes beinhaltet alle Personen die am Projekt mitgearbeitet bzw. das Projektteam betreut haben.

## 1.1 Projektteam

Das Projektteam bestand aus drei Personen, die alle an der Erstellung der Webseite arbeiteten und auch die Diplomarbeit schrieben. Die Projektteammitglieder besuchen alle den 5. Jahrgang an der HTL Dornbirn, im Schwerpunkt Betriebsinformatik des Zweiges Wirtschaftsingenieurwesen.

### 1.1.1 Simon Ellensohn



*Abb. 1 Simon Ellensohn*

Als Projektleiter (PL) war Simon stets für das Projektmanagement und der Aufgabenverteilung zuständig. Da er sehr große Kenntnisse im Bereich der Hypertext Preprocessor (*PHP*) Programmierung besitzt wurde er neben der Rolle als Projektleiter der leitende Programmierer. Er war vor allem für große Teile des Backends und der Benutzeroberfläche zuständig. Seine Motivationskraft und ständige Hilfsbereitschaft wurde vom Projektteam sehr geschätzt.

### 1.1.2 Martin Leimser



*Abb. 2 Martin Leimser*

Martins Themenschwerpunkt lag in der Erstellung und Instandhaltung der Projektpläne. Er nahm somit ein Teil der Arbeit vom PL ab, damit dieser einen größeren Fokus auf die Programmierung setzen konnte. Neben den Projektplänen war Martin auch in der Programmierung beschäftigt, mit dem Hauptfokus auf der Suche. Außerdem brachte er seine eigenen Ideen in das Konzept der Benutzeroberfläche mit ein.

### 1.1.3 Julian Martin



*Abb. 3 Julian Martin*

Julian arbeitete in allen Teilen des Projektes mit. Begonnen mit dem Projektmanagement über die Programmierung des Backends bis zum Gestalten der Benutzeroberfläche. Seine Hauptaufgaben lagen meist in der Umsetzung einzelner Komponente für die Benutzeroberfläche. Er folgte den Aufgaben des PL stetig und erweiterte teils seine Aufgaben selbstständig.

## 1.2 Projektbetreuer

Dem Projektteam standen zwei Projektbetreuer zur Seite. Benjamin Meier wurde von ToWa zur Verfügung gestellt. Herr Diethard Kaufmann, unser Lehrer in Betriebsinformatik und betriebliche Informationssysteme (BIBI), war einverstanden unser Projektbetreuer seitens der Schule zu sein.

### 1.2.1 Dipl.-Ing. Diethard Kaufmann



*Abb. 4 Diethard Kaufmann*

Herr Diethard Kaufmann betreute das Projekt von seitens der HTL Dornbirn. Er unterrichtet an der HTL im Bereich Betriebsinformatik. Weiters arbeitet er als Java-Entwickler bei Inet-logistics in Dornbirn. Sein Feedback zu projektbezogenen Dokumenten und seine Hilfe im Bereich der Terminplanung wurden vom Projektteam sehr geschätzt.

### 1.2.2 Benjamin Meier, MSc



*Abb. 5 Benjamin Meier*

Herr Benjamin Meier ist Leiter der Entwicklung bei ToWa. Er wurde dem Projektteam als Betreuer von ToWa zur Seite gestellt. Durch seine Erfahrung in der Web-Programmierung konnte er bei vielen technischen Schwierigkeiten helfen. Seine Ratschläge im Bereich der Entwicklung wurden sehr geschätzt.

## 1.3 Projektmitarbeiterin

Zusätzlich zu den Projektbetreuern wurde dem Projektteam eine Projektmitarbeiterin von ToWa zur Verfügung gestellt. Frau Elena Zabrodina erstellte diverse Ansichten der Benutzeroberfläche der Webseite.

### 1.3.1 Elena Zabrodina



*Abb. 6 Elena Zabrodina*

Frau Elena Zabrodina ist eine Designerin und Grafikerin von ToWa. Mittels vom Projektteam erstellten Mockups entwarf sie Vorlagen für das konkrete Web-Design. Bei der Umsetzung des Designs stand sie dem Projektteam stets für Ratschläge und Feedback zur Seite.

## 2 ToWa GmbH



Die ToWa GmbH ist eine Digitalagentur aus Vorarlberg. Eine Internetagentur (= Digitalagentur) ist ein auf Dienstleistungen spezialisiertes Unternehmen, welches im Auftrag von Kundinnen die Konzeption, Gestaltung, Programmierung und Pflege von Webseiten, Blogs und anderen digitalen Kommunikationskanälen übernimmt.

Ihr Firmensitz ist in Bregenz. Ihr Hauptaugenmerk gilt der Entwicklung von Webseiten mit WordPress, einem CMS (Content Management Systems) und neuerdings auch der Entwicklung mit *Laravel*.

Das Unternehmen hat 16 Mitarbeiter und ist eine der führenden Digitalagenturen in Vorarlberg.

ToWa hat bereits einige Preise gewonnen.

Zum einen den Red Dot Award 2014 im Bereich Kommunikationsdesign für die Neuveröffentlichung Ihrer Internetpräsenz.

Weiters wurden Sie 2014 als 21. im Kreativranking aller Digitalagenturen Österreichs ausgezeichnet.

Diverse Projekte von ToWa sind:

- Junge Wirtschaft Vorarlberg (<http://www.jwv.at/>)
- Vorarlberger Landesversicherung (<http://www.vlv.at/>)
- Hirschmann (<http://www.hirschmann-automotive.com/>)

## 3 Projektmanagement

Das Projektmanagement (PM) diente während dem Projekt als Übersicht für das Erreichen der Ziele und dem Zeitmanagement.

### 3.1 Allgemein



Bevor mit der Programmierung begonnen werden konnte, wurden Projektpläne erstellt. Diese wurden speziell vor der Programmierung erstellt, damit alle Ziele vor dem Start der Programmierung bereits festgelegt sind. Dadurch kann eine größere Rücksicht auf die festgelegten Risiken genommen werden.

Die Projektpläne wurden laufend aktualisiert, um festzustellen, ob alle definierten Termine eingehalten werden konnten.

Es wurden alle Projektpläne, außer der Projektterminplan (PTP) und der Projektstrukturplan (PSP), mit Google Docs erstellt. Für die Erstellung des PSP und PTP wurde zum einen WBStool 8 und zum anderen Microsoft Project 2016 verwendet.

## 3.2 Projektauftrag

Im Projektauftrag wurden die Projektziele und die geplanten Start- und Endtermine für das Projekt festgelegt. Weiters werden das Projektteam, der Auftraggeber und die Betreuer aufgelistet. Mit der Unterschrift des Projektauftrags von Projektauftraggeber und Projektleiter wurde das Projekt offiziell gestartet.

Projektauftrag	
<b>Projektstartereignis</b> <ul style="list-style-type: none"> <li>Schulbeginn 2015/16</li> </ul>	<b>Projektstarttermin</b> <ul style="list-style-type: none"> <li>14.09.2015</li> </ul>
<b>Inhaltliches Projektendereignis</b> <ul style="list-style-type: none"> <li>Fertigstellung des Eventplaners und der Diplomarbeit</li> </ul> <b>Formales Projektendereignis</b> <ul style="list-style-type: none"> <li>Abgabe der Diplomarbeit und der Software</li> </ul>	<b>Projektendtermine</b> <ul style="list-style-type: none"> <li>08.04.2016</li> </ul>
<b>Projektziele</b> <ul style="list-style-type: none"> <li>Konzeption und Entwicklung eines Eventplaners als webbasierte Lösung</li> </ul> <b>Zusatzziel</b> <ul style="list-style-type: none"> <li>Automatische Adaption auf weitere Städte möglich</li> </ul>	<b>Nichtziel</b> <ul style="list-style-type: none"> <li>Entwicklung einer separaten Applikation des Eventplaners für mobile Geräte</li> </ul>
<b>Hauptaufgaben (Projektphasen)</b> <ul style="list-style-type: none"> <li>Projektmanagement</li> <li>Analyse</li> <li>Design</li> <li>Programmierung</li> <li>Testing</li> <li>Dokumentation / Diplomarbeit</li> </ul>	<b>Projektauftraggeber</b> <ul style="list-style-type: none"> <li>Florian Wassel</li> </ul> <b>Projektbetreuer</b> <ul style="list-style-type: none"> <li>Diethard Kaufmann</li> <li>Benjamin Meier</li> </ul>
<b>Projektleiter</b> <ul style="list-style-type: none"> <li>Simon Ellensohn</li> </ul>	<b>Projektteam</b> <ul style="list-style-type: none"> <li>Martin Leimser</li> <li>Julian Martin</li> </ul>
<div style="display: flex; justify-content: space-around; align-items: flex-end;"> <div style="text-align: center;">               Florian Wassel, (Projektauftraggeber)         </div> <div style="text-align: center;">               Simon Ellensohn, (Projektleiter)         </div> </div>	

Tab. 1 Projektauftrag



### 3.3 Projektzieleplan

Mit dem Projektzieleplan wurden die Haupt-, Neben- und Nichtziele des Projekts festgelegt. Hauptziele müssen für einen erfolgreichen Projektabschluss erreicht werden. Nebenziele sind optional und müssen nicht unbedingt erfüllt werden. Mittels Nichtzielen wurden Projektgrenzen definiert.

#### 3.3.1 Hauptziele

- Programmierung einer Webseite für das Unternehmen ToWa GmbH, die Events in der Umgebung anzeigt.
  - Events werden in Form von Kacheln aufgelistet. Durch Klicken auf eine Kachel kommt man auf die Eventpage und erhält dort weitere Informationen über das Event, wie z.B. Veranstaltungsort, Termin oder genauere Beschreibung des Events.
  - Die Events werden zusätzlich, in einer separaten Kartenansicht, als Marker eingezeichnet. Durch Klicken auf einen Marker (Die zur Darstellung von Events dienen) werden weitere Informationen angezeigt und man kann auch weiter auf die Eventpage wechseln.
- Einteilung in vier Arten von Benutzern (nicht angemeldete Benutzer, angemeldete Benutzer, Moderatoren und Administratorinnen), wobei sich jeder als angemeldeter Benutzer registrieren kann.
  - Zur Registrierung ist die E-Mail-Adresse (die gleichzeitig als Benutzername gilt), der Vor- und Zuname und die Adresse notwendig. Alternativ kann die Registrierung über existierende Accounts der sozialen Netzwerke Facebook, Google+ und Twitter getätigt werden.
  - Alle Benutzer können sich über Events informieren.
  - Angemeldete Benutzer können zu Moderatoren und Administratorinnen erhoben werden.
  - Angemeldete Benutzer können Events speichern, die dann von Administratorinnen und Moderatoren veröffentlicht werden können.
- Erstellen einer Suche der Events für alle Benutzer.
  - Es kann nach Bezeichnung, Kategorie und Datum gefiltert werden.

- Erstellen der Create-, Read-, Update- und Delete-Funktionen (CRUD) für alle Administratorinnen
  - Erstellen, Bearbeiten, Lesen und Löschen von Events und Benutzern.

### 3.3.2 Nebenziele

- Eine automatische Adaption auf weitere Städte ist möglich.
  - Lösung 1: Eine einzelne große Plattform, die den momentanen Standort abfragt und Events aus der Umgebung anzeigt. Es wäre dort auch möglich Events von anderen Städten anzusehen.
  - Lösung 2: Mehrere individuelle Seiten für verschiedene Städte.
- Auflistung von empfohlenen Events, die durch mitprotokollieren der Aktivitäten eines Benutzers bestimmt werden; z.B. wenn ein Benutzer an mehreren Konzertevents teilnimmt oder sich speziell solche Events anschaut, werden ihm Events der Kategorie Konzert empfohlen.

### 3.3.3 Nichtziele

- Eine Applikation für mobile Geräte wird nicht entwickelt.

### 3.4 Projektorganisation und -organigramm

In folgender Tabelle wird dargestellt wie das Projekt organisiert ist. Es werden in dieser Tabelle Betreuer, Auftraggeber, Projektleiter und Teammitglieder zusammen mit ihren jeweiligen Aufgabenbereichen und benötigten Skills aufgelistet.

Projektorganisation		
Projektrolle	Aufgabenbereiche/Skills	Name
Projektauftraggeber	<ul style="list-style-type: none"> <li>– Vorgaben geben</li> <li>– kommunikationsfähig</li> </ul>	Florian Wassel
Projektbetreuer	<ul style="list-style-type: none"> <li>– Hilfe bei Problemen, Korrektur von Dokumenten bzw. Hinweise auf Ergänzungen</li> <li>– Fachwissen, hilfsbereit</li> </ul>	Diethard Kaufmann, Benjamin Maier
Projektleiter	<ul style="list-style-type: none"> <li>– Koordination und Leitung des Projekts, Projektmanagement, Entwicklung, Design, Dokumentation</li> <li>– Motivierend, organisiert, kommunikationsfähig, engagiert, Fachwissen</li> </ul>	Simon Ellensohn
Projektteammitglied	<ul style="list-style-type: none"> <li>– Entwicklung, Design, Dokumentation</li> <li>– Motiviert, teamfähig, kommunikationsfähig, engagiert, Fachwissen</li> </ul>	Martin Leimser, Julian Martin

Tab. 2 Projektorganisation

Folgende Abbildung (Abb. 8) ist ein Projektorganigramm. Es stellt den hierarchischen Aufbau des Projekts dar.

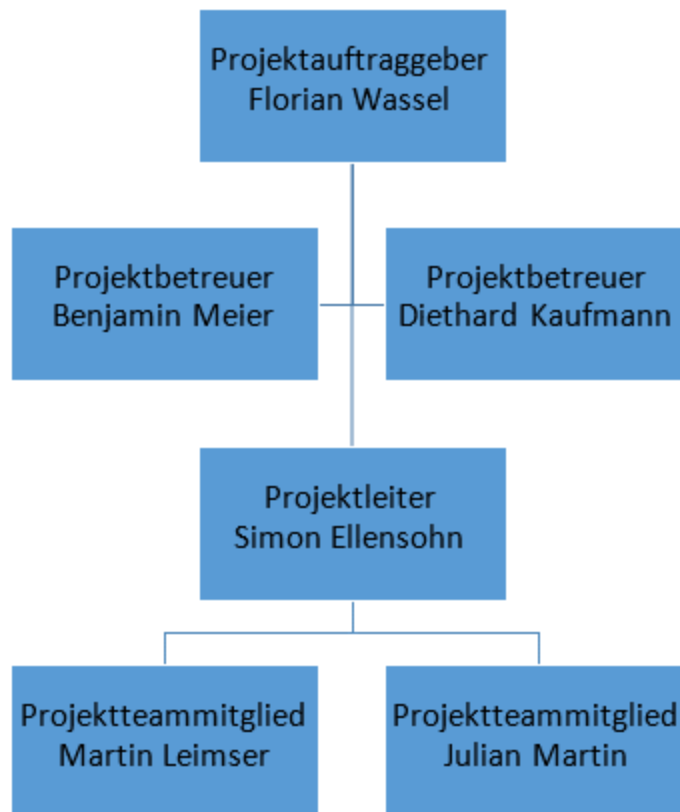


Abb. 8 Projektorganigramm

## 3.5 Projektstrukturplan

Im Projektstrukturplan (PSP) wurden die gesamten Arbeitsschritte des Projekts in Phasen gegliedert, welche ferner in Arbeitspakete unterteilt wurden. Phasen und Arbeitspakete wurden grundsätzlich chronologisch abgearbeitet. Es kam jedoch gewöhnlich vor, dass mehrere Arbeitspakete parallel bearbeitet wurden.

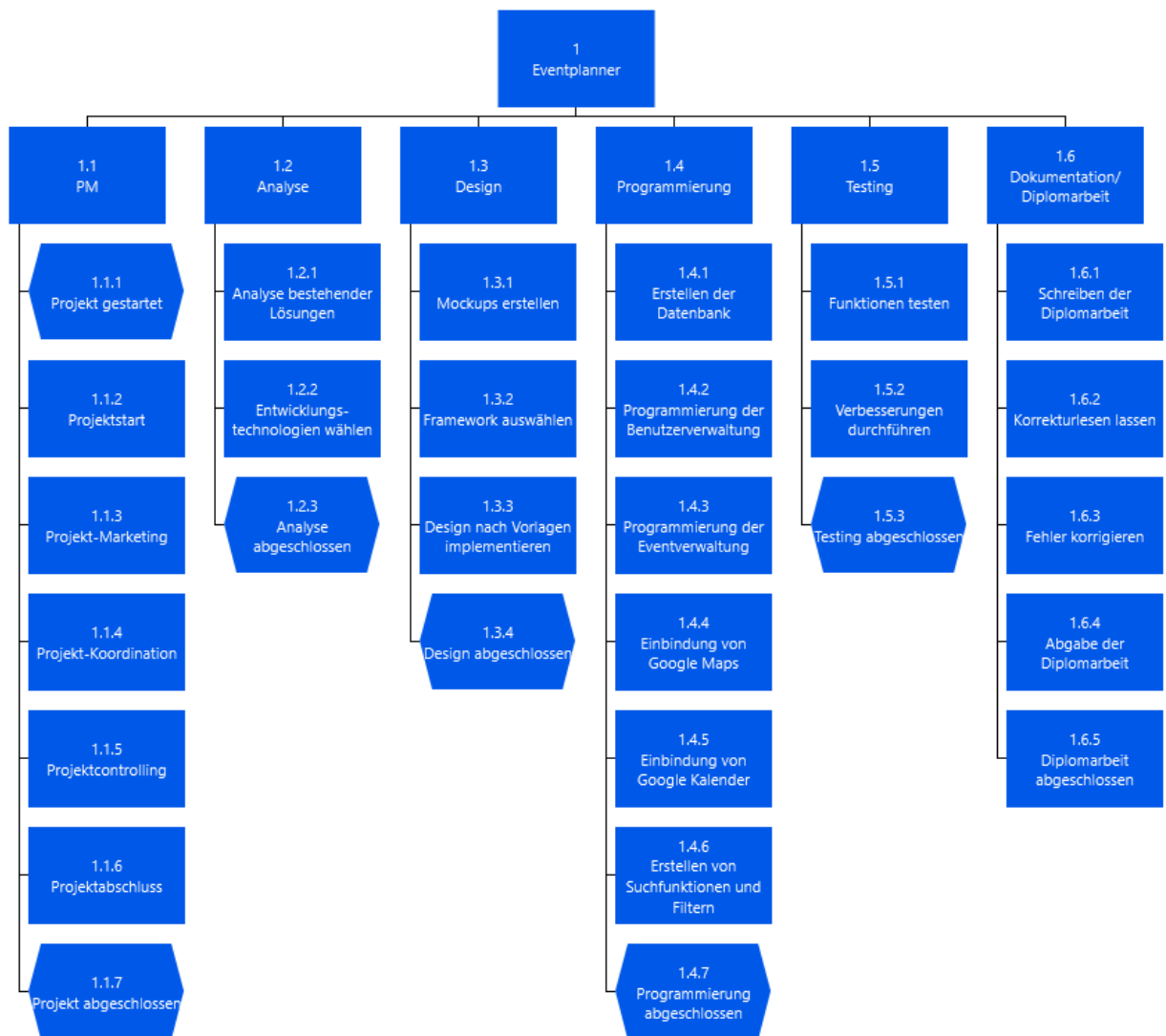


Abb. 9 Projektstrukturplan

Neben den Arbeitspaketen gibt es Meilensteine (hier als Sechsecke dargestellt). Meilensteine werden in der Regel verwendet, um zu signalisieren, dass ein wichtiger Teil des Projekts begonnen, abgeschlossen oder fertiggestellt wurde.

Der PSP ist die Basis des Projektterminplans (PTP) - Phasen und Arbeitspakete wurden direkt in den Terminplan übernommen.

Der PSP wurde während des Projekts als Hilfe bei der Organisation und Aufgabenverteilung verwendet. Bei einem Projekt dieses Umfangs war es wichtig, eine gewisse Struktur in die verschiedenen Aufgaben zu bringen. Der PSP half dank der angewandten Struktur sehr, da die Arbeitspakete möglichst chronologisch angeordnet wurden.

## 3.6 Projektterminplan

Im Projektterminplan wurden die Daten von Arbeitsbeginn und Ende jedes Arbeitspakets geplant. Der PTP gab eine sehr grobe Planung der Termine vor, welche im Laufe des Projekts zum Teil voneinander abwichen, da Arbeitspakete vor oder nach dem Endtermin abgeschlossen wurden.

PSP-Code	Vorgangsname	Dauer	Anfang	Ende
<b>1</b>	<b>Projektmanagement</b>	<b>150 Tage</b>	<b>Mon 14.09.15</b>	<b>Fre 08.04.16</b>
1.1	Projekt gestartet	0 Tage	Mon 14.09.15	Mon 14.09.15
1.2	Projektstart	1 Tag	Mon 14.09.15	Mon 14.09.15
1.3	Projektmarketing	150 Tage	Mon 14.09.15	Fre 08.04.16
<b>1.4</b>	<b>Projektkoordination</b>	<b>146 Tage</b>	<b>Fre 18.09.15</b>	<b>Fre 08.04.16</b>
<b>1.5</b>	<b>Projektcontrolling</b>	<b>146 Tage</b>	<b>Fre 18.09.15</b>	<b>Fre 08.04.16</b>
1.6	Projektabschluss	1 Tag	Fre 08.04.16	Fre 08.04.16
1.7	Projekt abgeschlossen	0 Tage	Fre 08.04.16	Fre 08.04.16
<b>2</b>	<b>Analyse</b>	<b>12 Tage</b>	<b>Die 15.09.15</b>	<b>Mit 30.09.15</b>
2.1	Analyse bestehender Lösungen	7 Tage	Die 22.09.15	Mit 30.09.15
2.2	Entwicklungstechnologie auswählen	1 Tag	Die 15.09.15	Die 15.09.15
2.3	Analyse abgeschlossen	0 Tage	Mit 30.09.15	Mit 30.09.15
<b>3</b>	<b>Design</b>	<b>120 Tage</b>	<b>Die 15.09.15</b>	<b>Mon 29.02.16</b>
3.1	Mockups erstellen	10 Tage	Die 22.09.15	Son 04.10.15
3.2	Framework auswählen	1 Tag	Mit 16.09.15	Mit 16.09.15
3.3	Design nach Vorlagen implementieren	61 Tage	Mon 07.12.15	Mon 29.02.16
3.4	Design abgeschlossen	0 Tage	Mon 29.02.16	Mon 29.02.16
<b>4</b>	<b>Programmierung</b>	<b>133 Tage</b>	<b>Die 06.10.15</b>	<b>Don 07.04.16</b>
4.1	Erstellen der Datenbank	84 Tage	Die 06.10.15	Fre 29.01.16
4.2	Programmierung der Benutzerverwaltung	19 Tage	Die 06.10.15	Fre 30.10.15
4.3	Programmierung der Eventverwaltung	84 Tage	Die 06.10.15	Fre 29.01.16
4.4	Einbindung von Google Maps	11 Tage	Mon 26.10.15	Mon 09.11.15
4.5	Einbindung von Google Kalender	34 Tage	Mon 26.10.15	Don 10.12.15
4.6	Erstellen von Suchfunktionen und Filtern	91 Tage	Mon 26.10.15	Mon 29.02.16
4.7	Programmierung abgeschlossen	0 Tage	Mon 29.02.16	Mon 29.02.16
<b>5</b>	<b>Testing</b>	<b>45 Tage</b>	<b>Mon 01.02.16</b>	<b>Fre 01.04.16</b>
5.1	Funktionen testen	45 Tage	Mon 01.02.16	Fre 01.04.16
5.2	Verbesserungsarbeiten durchführen	45 Tage	Mon 01.02.16	Fre 01.04.16
5.3	Testing abgeschlossen	0 Tage	Fre 01.04.16	Fre 01.04.16
<b>6</b>	<b>Dokumentation / Diplomarbeit</b>	<b>70 Tage</b>	<b>Mon 04.01.16</b>	<b>Fre 08.04.16</b>
6.1	Schreiben der Diplomarbeit	70 Tage	Mon 04.01.16	Fre 08.04.16
6.2	Korrekturlesen lassen	25 Tage	Mon 22.02.16	Fre 25.03.16
6.3	Verbesserungsarbeiten durchführen	35 Tage	Mon 22.02.16	Fre 08.04.16
6.4	Abgabe der Diplomarbeit	1 Tag	Fre 08.04.16	Fre 08.04.16
6.5	Diplomarbeit abgeschlossen	0 Tage	Fre 08.04.16	Fre 08.04.16

Abb. 10 Projektterminplan

### 3.7 Meilensteinplan

Der Meilensteinplan zeigt chronologisch zentrale Punkte im Projekt (Meilensteine), die erreicht oder abgeschlossen wurden. Dafür wurden die Meilensteine mit dem zugehörigen PSP-Code und einem ungefähren Plantermin in eine Tabelle eingetragen.

Meilensteinplan		
PSP-Code	Meilenstein	Plantermin
2.3	Analyse abgeschlossen	15.09.2015
3.4	Design abgeschlossen	10.10.2015
4.7	Programmierung abgeschlossen	29.02.2016
5.3	Testing abgeschlossen	31.03.2016
6.5	Diplomarbeit abgeschlossen	07.04.2016

*Tab. 3 Meilensteinplan*



### 3.8 Risikoanalyse

Eine Risikoanalyse wurde zu Beginn des Projekts durchgeführt, um potentielle Risiken zu identifizieren. Die Risiken werden in einer Matrix nach Eintreffen und Auswirkung auf das Projekt beurteilt. Zu den gefährlichen Risiken wurden parallel Lösungsmaßnahmen definiert, um das Eintreten des Risikos frühzeitig zu verhindern.

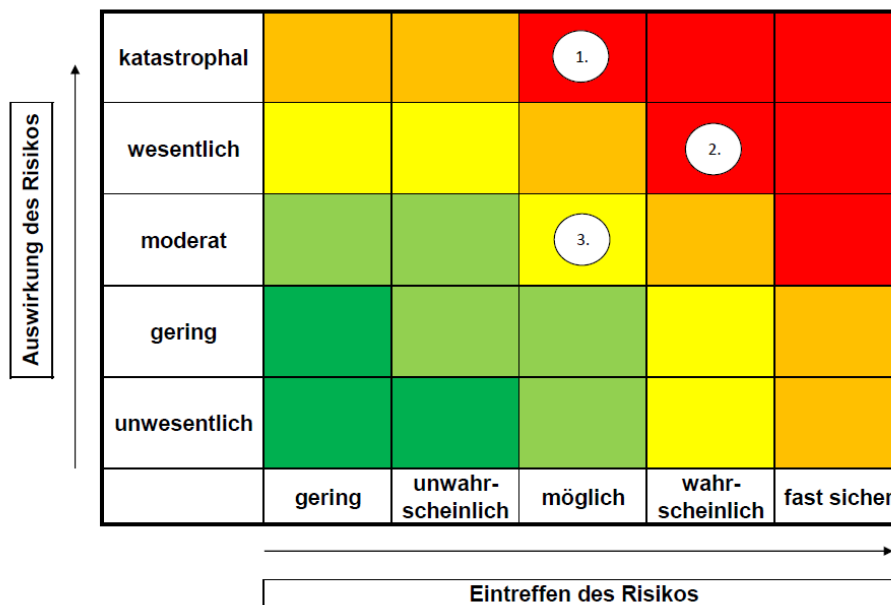


Abb. 11 Risikoanalyse

In folgender Tabelle werden die oben eingezeichneten Risiken mit der jeweiligen Nummer aufgelistet, auf Gefahr bewertet und Maßnahmen zur Lösung beschrieben.

Nr.	Risiko	Gefahr	Maßnahme(n)
1	Fehlendes Fachwissen in <i>Laravel</i>	hoch	Einlernen in <i>Laravel</i> mittels Laracasts
2	Fehlendes Fachwissen für Designumsetzung	hoch	Frühzeitiges Einlernen in Syntactically Awesome Style Sheets (Sass)
3	Falsche Einschätzung des Funktionsumfangs in Bezug auf Umsetzungsdauer	mittel	Mehr Zeit in die Umsetzung investieren, um rechtzeitig fertig zu werden

Tab. 4 Risikoanalyse

## 3.9 Projektabschlussbericht

Für den Projektabschlussbericht fand am Ende des Projekts ein Abschlussmeeting statt, bei dem das Projekt auf die Erreichung der Projektziele besprochen wurde.

### 3.9.1 Projektverlauf

Das Projekt verlief nach der Meinung des Projektteams sowie des Auftraggebers sehr gut.

Zu Beginn gab es kleine Schwierigkeiten mit der Aufgabenverteilung und Motivation, aber im Laufe des Projekts verlief alles nach Plan und durch dies konnten viele Erfahrungswerte gesammelt werden.

Es traten im Projektverlauf keine unvorhergesehenen Ereignisse ein.

### 3.9.2 Zielerreichung

Bis auf das Nebenziel: "Auflistung von empfohlenen Events, die durch Mitprotokollieren der Aktivitäten eines Users bestimmt werden", wurden alle Ziele erreicht. Die Funktionalität des Projekts wurde sogar weiter ausgebaut, als ursprünglich geplant.

Das Projekt war ein Erfolg, da die Software von ToWa zukünftig eingesetzt wird. Zusätzlich konnte das Projektteam Ihre Fähigkeiten stark ausbauen und, das Projekt kann als Referenz bei zukünftigen Bewerbungen verwendet werden.

### 3.9.3 Ausblick

Es ist geplant, dass ToWa den Source Code des Eventplaners übernimmt, damit ihn die Firma bei zukünftigen, ähnlichen Projekten als Basis oder Referenz verwenden kann.

## 4 Analysephase

In der Analysephase wurde eine Evaluierung von bereits vorhandenen, ähnlichen, Produkten durchgeführt. Neben dieser Evaluierung wurden auch die gesamten Funktionen des Eventplaners festgelegt und diverse Use Case (zu Deutsch: Anwendungsfall) Diagramme erstellt.

Um sich von anderen Eventplanern abzuheben, wurde auf ein modernes Design, zusammen mit einer benutzerfreundlichen und selbsterklärenden Bedienung, sowie auf die Verwendung von modernen Technologien gesetzt. Eine Anmeldung mit existierenden Accounts von den sozialen Netzwerken Facebook, Google+ und Twitter wurde implementiert. Weiters ist die Kartenansicht, auf der Eventlocations angezeigt werden, ein zentrales Element dieser Lösung. Außerdem war es ein wichtiger Punkt, dass jeder einen Eventantrag erstellen kann. Dadurch ergab sich auch die Möglichkeit, private Events zu erstellen und den Link, der zum Event führt, nur mit Freunden zu teilen.

Der Grund für die Eigenentwicklung eines neuen Eventplaners war die Stadt Heidelberg. Da die Stadt keine geeignete Plattform für die Verwaltung deren Events besitzt, bekam ToWa den Auftrag eine Verwaltungssoftware zu entwickeln. Das Projekt war bei ToWa im Stillstand und wurde deshalb dem Projektteam übergeben.

## **4.1 Evaluierung Eventplaner**

Vor dem Beginn der Entwicklung wurden verschiedene existierende Eventplaner auf Funktionsumfang und Web-Design evaluiert. Ziel dabei war es, Möglichkeiten zu finden, sich von der Konkurrenz abzuheben.

### **4.1.1 Events.at**

Events.at ist eine Internetseite die von Telekurier Online Medien GmbH betrieben wird. Sie wird für ganz Österreich betrieben, hat aber Schwächen in den ländlichen Bereichen. Die Plattform konzentriert sich auf große Ballungsräume wie etwa Wien.

Durch das veraltete Design wird die Internetseite schnell unübersichtlich und auch schwer bedienbar.

### **4.1.2 Oeticket**

oeticket.com wird von der CTS EVENTIM AG & Co. KGaA verwaltet. Eventim bietet ein Modernes Design, das aber teilweise unübersichtlich wirkt. Es werden Karten für die Veranstaltungen direkt verkauft. Negativ fällt auf, dass keine Anmeldungen aus sozialen Netzwerken möglich sind. Es gibt auch keine Kartenansicht.

### **4.1.3 ländleTICKET**

Ländleticket ist eine regionale Lösung für Vorarlberg. Das Design ist jedoch sehr veraltet. Karten können dafür direkt auf der Internetseite gekauft werden.

### **4.1.4 Ticketmaster**

Ticketmaster gehört zu Live Nation Entertainment, welches das größte Entertainment Unternehmen der Welt ist. Die Internetseite hat ein modernes Design, jedoch ist sie nicht sehr benutzerfreundlich. Hier können wieder Tickets online erworben werden.

### **4.1.5 Facebook**

Facebook ist das größte soziale Netzwerk der Welt. Es wird ein sehr großes Publikum angesprochen und es kann eine Veranstaltung sehr leicht mit Freunden geteilt werden. Lokale Veranstaltungen zu finden gestaltet sich jedoch oft schwierig.

### 4.1.6 Zusammenfassung

In folgender Tabelle sind die eben beschriebenen Lösungen mit deren Vor- und Nachteilen als Zusammenfassung aufgelistet.

Lösung	Vorteile	Nachteile
events.at	+ In ganz Österreich verfügbar	– Geringes Angebot für kleinere Regionen – Veraltetes Design – Unübersichtlich – Viel Werbung
oeticket.com	+ Ticketverkauf + Modernes Design + Künstler folgen + Große Reichweite	– Keine Anmeldung mit Accounts aus sozialen Netzwerken – Keine Kartenansicht
laendleticket.com	+ Ticketverkauf	– Veraltetes Design
ticketmaster.de	+ Große Reichweite + Ticketverkauf	– Nicht sehr benutzerfreundlich
facebook.com	– Große Reichweite – Teilen mit Freunden (Einladen, ...) – Sehr viele registrierte Benutzer	– Finden von lokalen Veranstaltungen sehr versteckt oder umständlich

Tab. 5 Evaluierung von Eventplanern

## 4.2 Use-Case Diagramm

Das Use-Case Diagramm zeigt die Use Cases (Anwendungsfälle), die während der Benutzung des Eventplaners auftreten und welche Akteure (hier die verschiedenen Benutzerrollen) an welchem Use Case teilhaben.

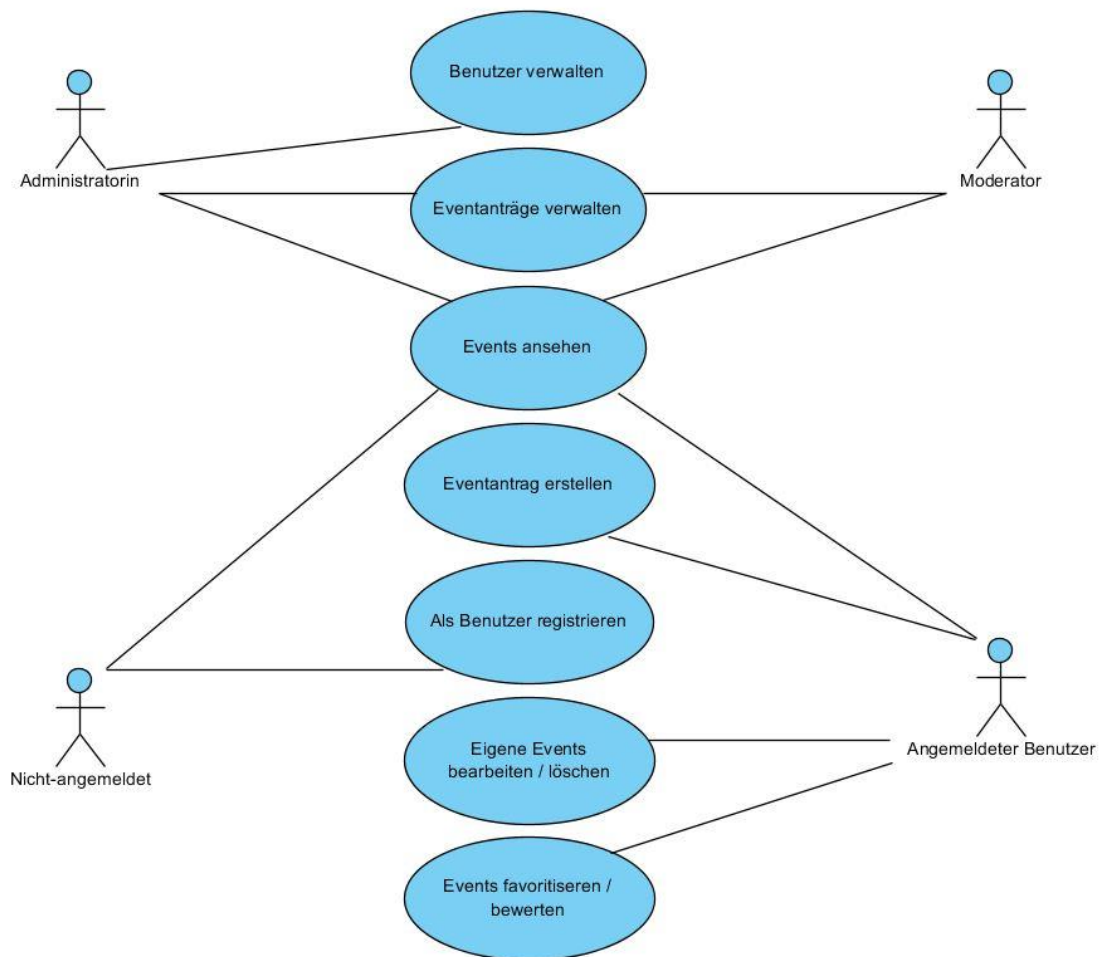


Abb. 12 Use-Case Diagramm

## 5 Einarbeitung

Bevor das eigentliche Projekt beginnen konnte, musste noch festgelegt werden, wie die Webseite entwickelt wird. Auf Grund von Vorkenntnissen aus früheren Projekten in der Schule, wurde *PHP* als Programmiersprache gewählt. WordPress wurde von ToWa für das Projekt empfohlen. Sie haben bereits große Vorkenntnisse im Bereich WordPress. Bei der Wahl von WordPress wäre ToWa eine große Hilfe.

Weiters kam auch noch AngularJS (Ein JavaScript *Framework* von Google) in Frage, dies wurde aber wieder verworfen, auf Grund von zu wenigen Vorkenntnissen. Nach einer Recherche von *PHP-Frameworks*, kamen WordPress und *Laravel (PHP Framework)* in die engere Auswahl.

### 5.1 WordPress

Zu Beginn war die Umsetzung mittels des CMS WordPress geplant. ToWa entwickelt ihre Webprojekte überwiegend mit WordPress und Projektbetreuer Herr Meier ist ein Fachmann im Umgang mit diesem CMS. Aus diesem Grund schien es anfangs die optimale Lösung, WordPress für die Entwicklung zu verwenden.

Die Einarbeitung in WordPress begann Anfang August 2015. Bei einem Workshop erklärte Herr Meier die Grundlagen des CMS und wie damit grundsätzlich eine Webseite erstellt wird. Die restliche Einarbeitung erfolgte über Online-Tutorials. Es stellte sich schnell heraus, dass man sich mit WordPress im Backend, mittels vorprogrammierter Plugins und Design-Templates, schnell eine brauchbare Webseite zusammenstellen kann.

Schwierigkeiten gab es jedoch speziell im Verständnis des eigentlichen *PHP*-Codes der WordPress-Webseite und der damit verbundenen eigenständigen Entwicklung von Programmfunktionen und des Web-Designs. Der Code wirkte häufig unübersichtlich und nicht intuitiv. Für ein akzeptables Verständnis des Codes wäre eine sehr lange Einarbeitungsphase notwendig gewesen.

Auf Grund dieser Einarbeitungsschwierigkeiten wurde Anfang September 2015, mit der Einwilligung seitens ToWa, entschieden, den Eventplaner mit dem *PHP-Framework Laravel* zu entwickeln.

## 5.2 Laravel

*Laravel* ist ein *Framework* für die Programmiersprache *PHP*, das den Funktionsumfang von *PHP* erweitert bzw. leichter einsetzbar macht. *Laravel* liefert von Haus aus einige Funktionen, die sehr nützlich sind. Sehr wichtig ist vor allem das Command Line Interface (CLI) *Artisan*. Mit *Artisan* können Dateien nach bestimmten Vorlagen erstellt, oder bestimmte Dateien ausgeführt werden. Es können zum Beispiel *Migrations* (Dateien die Datenbanktabellen erstellen oder entfernen) von *Artisan* ausgeführt werden.

*Laravel* 5.1 wurde im Juni 2015 veröffentlicht und erhält Updates für 2 Jahre und Sicherheit Patches für 3 Jahre.<sup>1</sup> Durch diesen Langzeitsupport wurde *Laravel* 5.1 verwendet. Durch Laracasts (eine Online-Tutorial Webseite) wurde der Einstieg sehr erleichtert. Diese Videos, von Jeffrey Way, erleichtern es auch sehr komplexe Themenstellungen schnell und einfach zu lösen. *Laravel* wurde 2014 von sitepoint.com zum besten *Framework* des Jahres gewählt.

---

<sup>1</sup>Vgl. (Wikipedia, Laravel, 2016)



## 5.3 Vergleich zwischen WordPress und Laravel

In folgender Tabelle werden WordPress und *Laravel* auf einige für die Entwicklung entscheidende Punkte miteinander verglichen.

	WordPress	Laravel
Code-Verständlichkeit	Für Einsteiger in das CMS kann der Code komplex und unübersichtlich wirken. Auch mit <i>PHP</i> -Vorkenntnissen ist eine längere Einarbeitung notwendig, um gut mit WordPress entwickeln zu können.	Mit <i>PHP</i> -Vorkenntnisse ist die Funktionsweise des <i>Frameworks</i> schnell zu erlernen. Die Einarbeitung wird durch Laracast-Tutorials erleichtert, bei denen mitunter auch komplexe Programmabläufe verständlich erklärt werden.
Community	WordPress hat eine sehr große Community und eine Vielzahl von Supportforen.	Die Community von <i>Laravel</i> ist bereits groß und wächst stark an. Es sind bei vielen Problemen schnell Lösungen in den <i>Laravel</i> -Foren zu finden.
Dokumentation	Es gibt mit <i>codex.wordpress.org</i> eine umfangreiche Online-Dokumentation auf Deutsch und anderen Sprachen.	<i>Laravel</i> hat eine sehr umfangreiche Online Dokumentation, diese ist jedoch nur auf Englisch verfügbar.

Versionsverwaltung	Die Versionsverwaltung mit WordPress stellte sich ins besonders bei Verwendung von vielen Plugins als sehr komplex und umständlich heraus.	Mit Git ist die Versionsverwaltung sehr einfach. Datenbank <i>Migrations</i> erleichtern die Versionskontrolle der Datenbank.
--------------------	--	---

Tab. 6 Vergleich zwischen WordPress und Laravel

Auf Grund der sichtlich höheren Einsteigerfreundlichkeit von *Laravel* und der Tatsache, dass sich das *Framework* genau so gut wie WordPress für die Entwicklung eines Eventplaners eignete, wurde *Laravel* vorgezogen.

## 5.4 Entwicklungsumgebung

Entwickelt wurde in *PHPStorm* (IDE) auf Windows. Der lokale Web- und Datenbankserver liefen in einer virtuellen Maschine (*Homestead*), welche die Macher von *Laravel* vorkonfiguriert bereitstellen. Diese virtuelle Maschine läuft durch *Vagrant* im Hintergrund und hat keine negative Wirkung auf die Leistung. Außerdem sind die ganzen Berechtigungen für *Laravel* richtig konfiguriert, wodurch sofort mit der Programmierung begonnen werden kann.

Für die Versionskontrolle wurde *Git* verwendet. Mithilfe von *SourceTree* und dessen grafischer Benutzeroberfläche wurde der Code verwaltet und zusammengefügt. Für die Programmierung wurden zwei *Branches* verwendet. Auf "dev" (kurz für development) war stets die aktuellste Version des Projektes während auf "master" ein- bis zweimal der Code von "dev" gemergt wurde (je nachdem wie viele neue Funktionen bzw. Bug fixes an einem Tag erstellt wurden).

Es wurden zwei *Branches* verwendet, da ToWa einen Testserver bereitstellte. Somit konnten die Projektbetreuer und Projektmitarbeiter stets dem Programmierfortschritt folgen und das Team auf Fehler im Programm hinweisen.

Auf dem Testserver war *Git* konfiguriert, dass nach jedem push (vom Client auf den Server) zuerst überprüft wurde, ob es sich um den "master" *Branch* handelt. Falls es sich um den „master“ *Branch* handelte, wurden automatisch alle neuen Dateien in den richtigen Ordner kopiert und die Stylesheets neu kompiliert.

In folgendem Verteilungsdiagramm (Abb. 13) ist die lokale Programmierumgebung dargestellt.

Dabei zeigt das Diagramm die virtuelle Umgebung *Vagrant*, in dem der Webserver (*nginx*) und Datenbank-Server (*MySQL*) verwendet wurde.

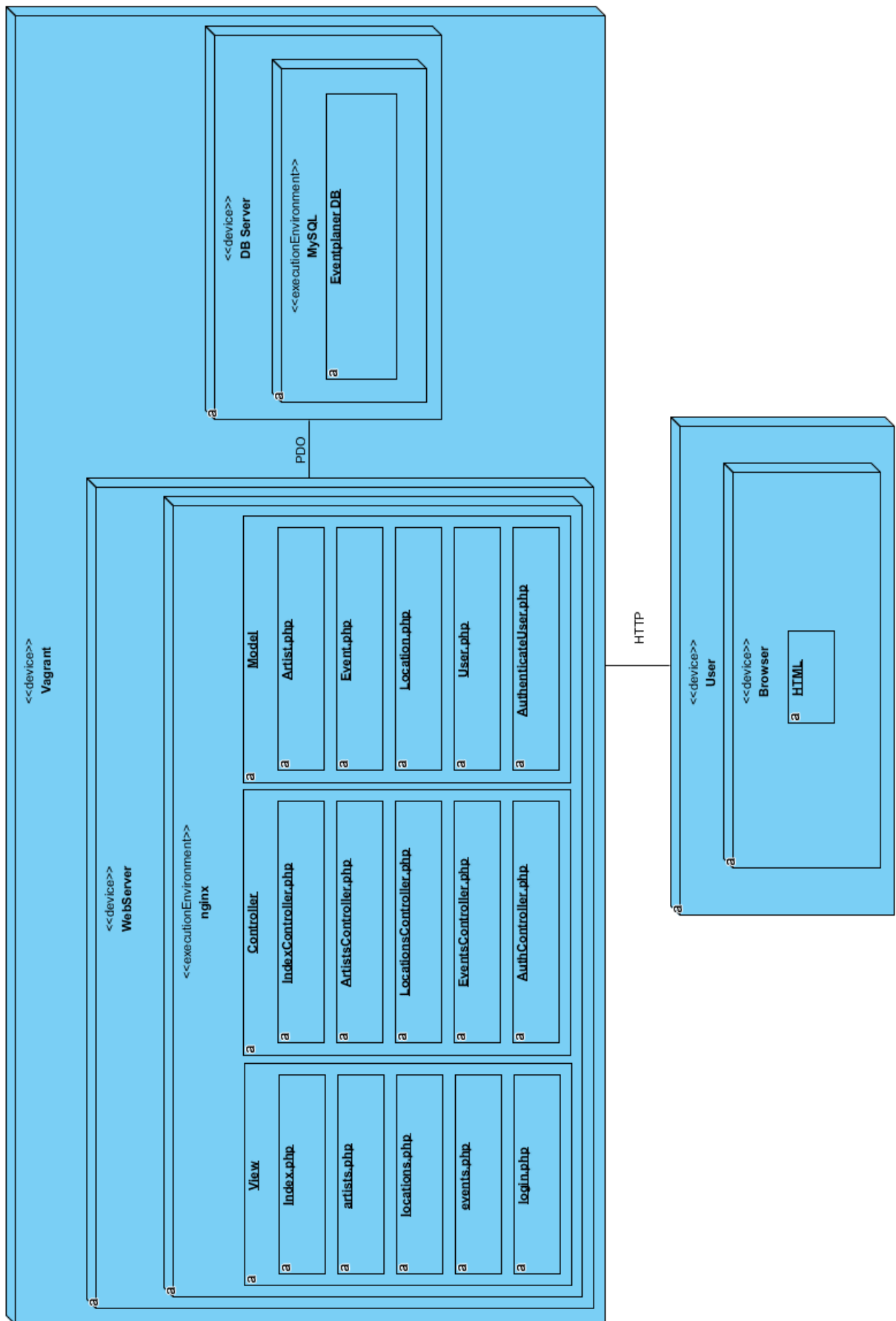


Abb. 13 Verteilungsdiagramm - Lokal

Im folgenden Verteilungsdiagramm (Abb. 14) ist die Testumgebung dargestellt. Der Unterschied zur lokalen Umgebung liegt darin, dass Apache als Webserver verwendet wurde und die Server nicht in einer virtuellen Maschine laufen. Außerdem konnte von außen auf die Applikation zugegriffen werden.

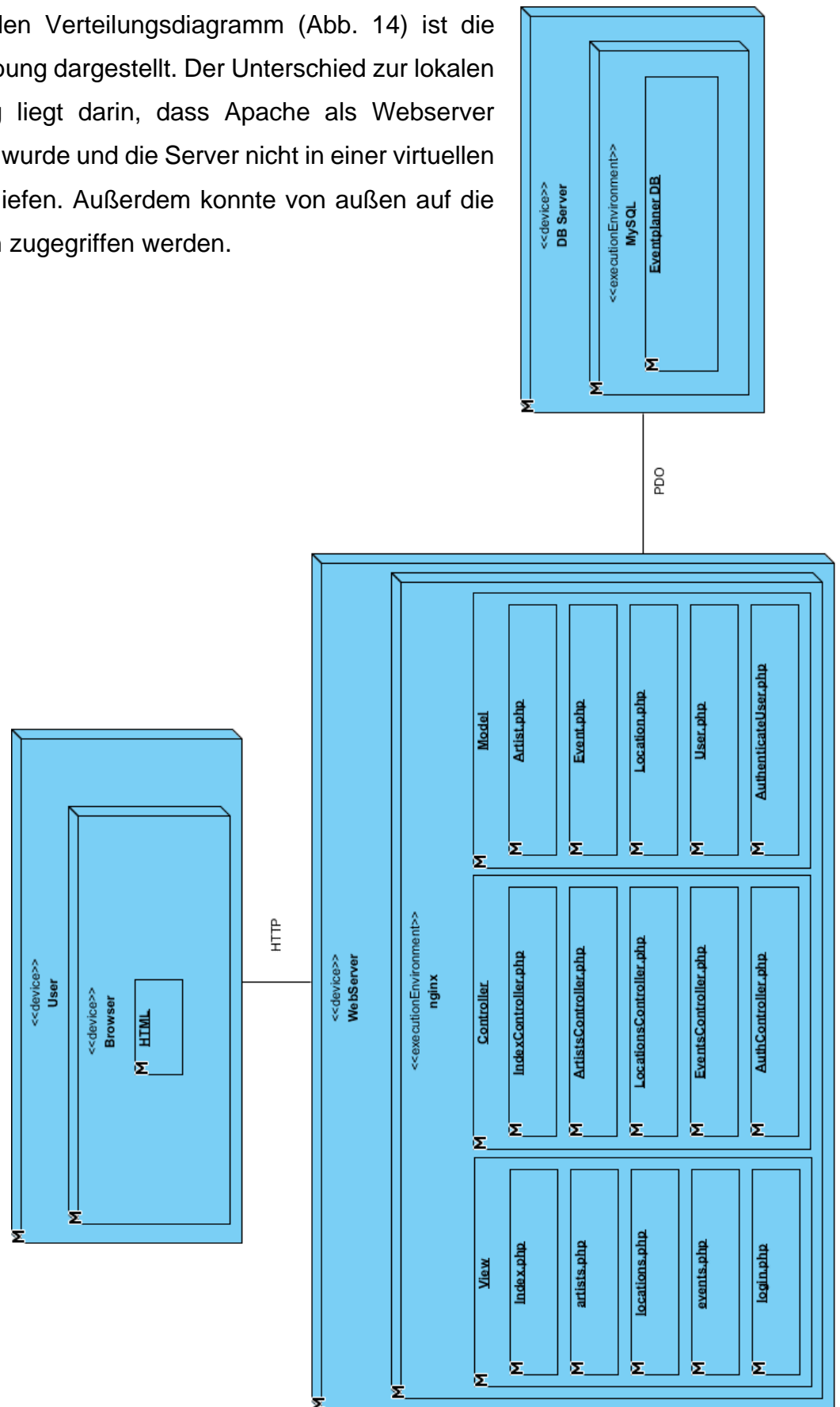


Abb. 14 Verteilungsdiagramm - Testserver

In folgender Abbildung (Abb. 15) ist ein Big Picture, welches für die Entwicklungsumgebung verwendete Software und Technologien auflistet. Weiters werden die wichtigsten Tabellen der Datenbank mit ihren grundlegenden Attributen dargestellt. Im Punkt Benutzer werden die verschiedenen Rollen mit ihrem Funktionsumfang aufgelistet. Das Big Picture stammt von vor dem Beginn der Entwicklung und diente dort als Überblick für das Projekt. Im Laufe des Projekts wurden einige Punkte ergänzt bzw. korrigiert.

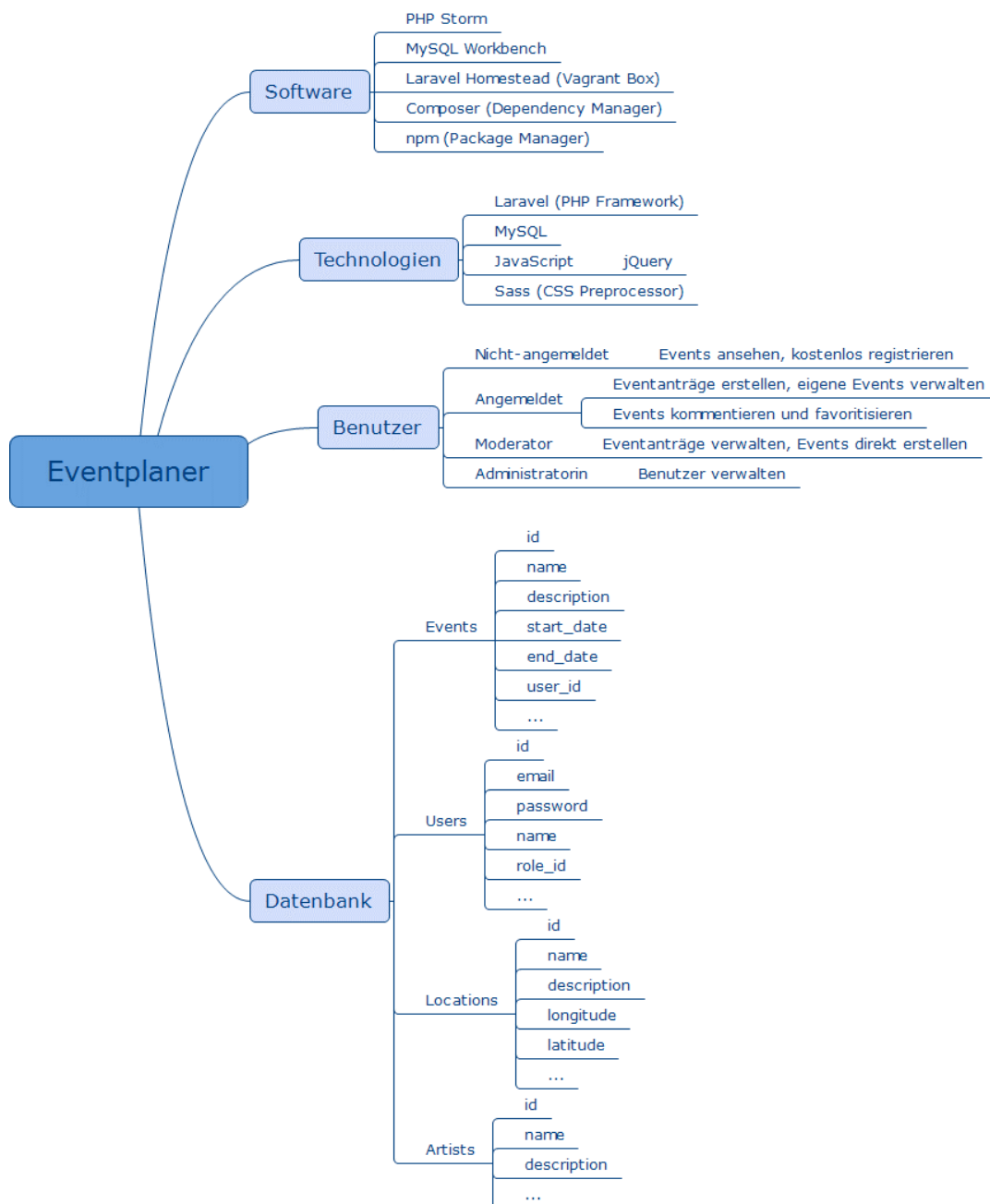


Abb. 15 Big Picture

## 6 Design

Anfangs wurden für das Design der Webseite nur die Standardkomponenten des Bootstrap-Frameworks verwendet. ToWa bot an, ein modernes Design für den Eventplaner zu entwerfen. Für dieses Design verwendete Designerin Frau Zabrodina vom Projektteam händisch skizzierte *Mockups*. Die Vorlagen waren Anfang Dezember 2015 fertig und die Umsetzung konnte gestartet werden. Für die Umsetzung wurde der Cascading Style Sheets (CSS) Preprocessor Sass verwendet.

Für die Einarbeitung in CSS und Sass wurden wieder Tutorials benutzt. Eine Quelle von Tutorials war [sass.codeschool.com](http://sass.codeschool.com). Die Einarbeitung war wesentlich einfacher als bei *Laravel*. Kenntnisse im CSS Bereich waren aber bereits vorhanden.

Seit Version 3 wird eine neue Dateinamenerweiterung benutzt. Diese lautet „.scss“ (Sassy CSS), der große Unterschied ist, dass wieder Klammern benötigt werden, um den Code übersichtlicher zu machen.

Ein sehr großer Vorteil ist, dass es Variablen gibt. Dadurch lassen sich große Veränderungen an der Benutzeroberfläche sehr schnell vornehmen. Ein weiterer Punkt sind Mixins, dadurch lässt sich Code sehr leicht wiederverwenden.

Der Unterschied zwischen CSS und SCSS ist am besten an einem Code Beispiel zu erkennen.

Im Beispiel von Abbildung 16 wird in der Navigation in allen „a“ Tags die Randfarbe geändert und die Schriftart verdunkelt.

```
//CSS
.navigation a {
  border-color: #333;
  color: darken(#333, 5%);
}

//SCSS
$variable: #333;

.navigation {
  a {
    border-color: $variable;
    color: darken($variable, 5%);
  }
}
```

Abb. 16 CSS / SCSS Vergleich

## 6.1 Mockups

Die *Mockups* wurden vom Projektteam auf Papier erstellt. Die erstellten *Mockups* bestimmen das grobe Design der Oberfläche, welche Frau Zabrodina für die Fertigstellung verwendete.

### 6.1.1 Startseite

Auf der Startseite werden die aktuellen Events dargestellt. Neben den Events können die Benutzer, von der Startseite aus, nach bestimmten Events suchen. Zur Anmeldeseite oder Einzelansicht eines Events navigieren und die verschiedenen Kategorien auswählen.

Die folgende Abbildung (Abb. 17) zeigt einen ersten Entwurf der Startseite.

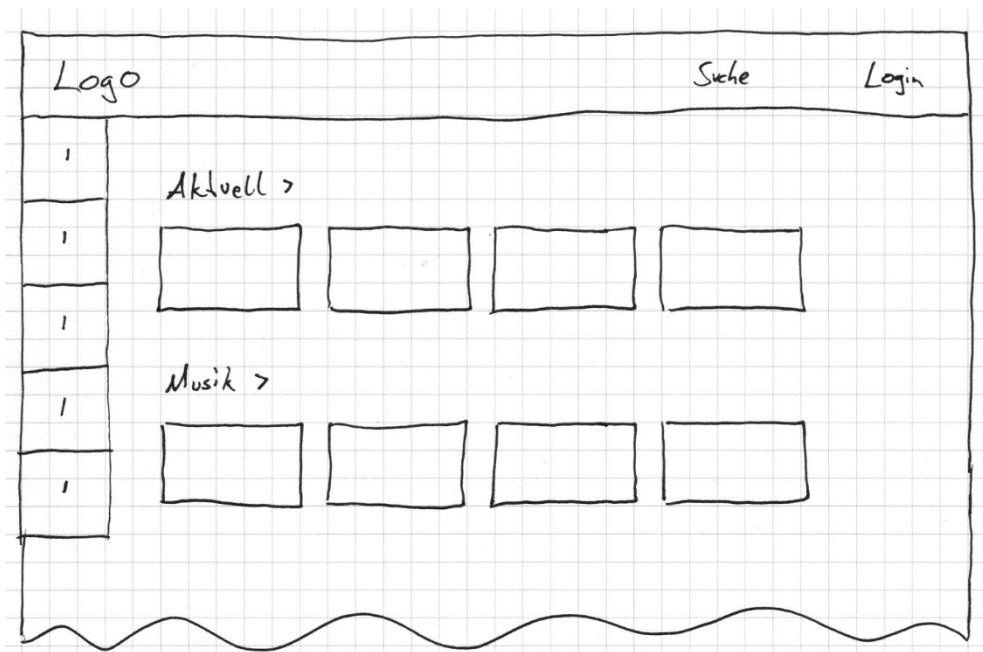


Abb. 17 Startseite Mockup



### 6.1.2 Eventansicht

Bei der Eventansicht werden alle wichtigen Informationen eines Events dargestellt. Neben den Informationen, welche sowohl den Veranstaltungsort als auch die betreffenden Künstler betreffen, gibt es einen Abschnitt bei dem die registrierten Benutzer kommentieren und das Event bewerten können.

In der folgenden Abbildung (Abb. 18) ist ein grober Entwurf der Eventansicht zu sehen.

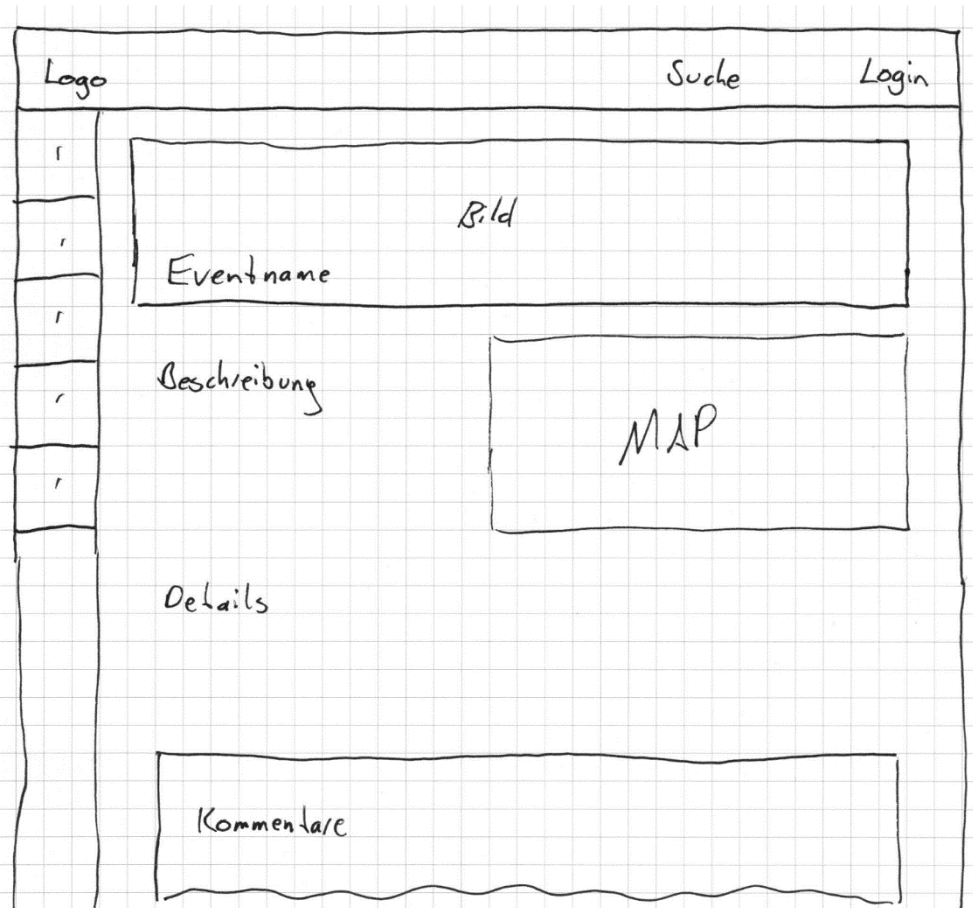


Abb. 18 Eventansicht Mockup

## 6.2 Designentwurf

Frau Elena Zabrodina entwickelte ein modernes Design für den Eventplaner, welches für die junge Zielgruppe des Projekts ansprechend ist. Das Design wurde mit Photoshop entwickelt.

Folgende Abbildung (Abb. 19) zeigt ein Entwurf einer Eventansicht von Frau Zabrodina.

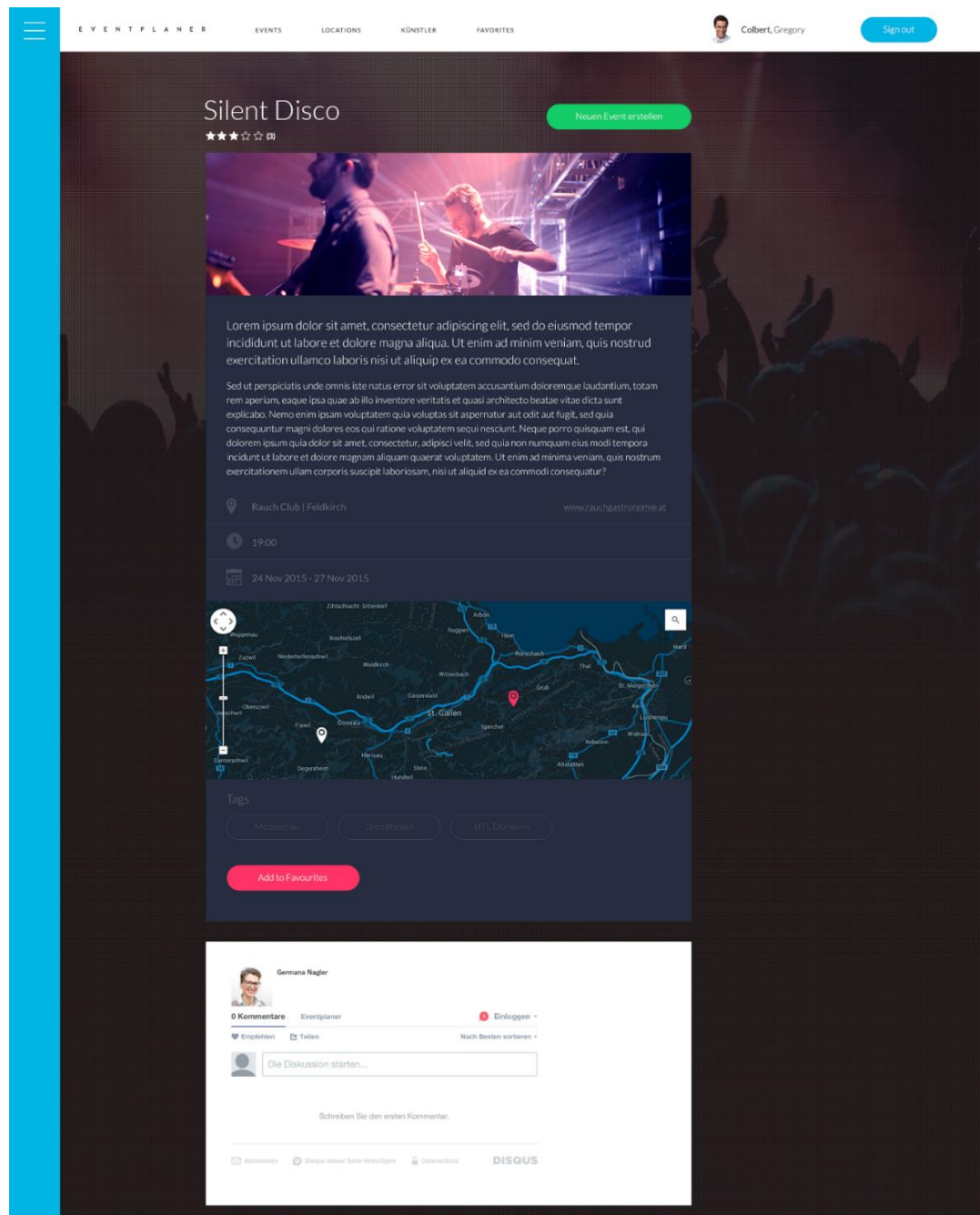


Abb. 19 Eventansicht Designentwurf

## 6.3 Designimplementierung

Für die Umsetzung des Designs wurde Sass verwendet. Sass bietet eine Verschachtelung von CSS Klassen, was den Schreibaufwand stark mindert. Durch Gulp werden alle Dateien in ordinäre CSS Daten kompiliert, wobei diese automatisch noch weiter minimiert werden. Minimiert bedeutet, dass alle Leerräume und Absätze entfernt werden.

Die entworfenen Vorlagen von Frau Zabrodina wurden über „assets.adobe.com“ dargestellt. Ein Vorteil der Adobe Creative Cloud ist, dass bestimmte Elemente auf der Vorlage ausgewählt werden können, wobei deren Größe und Hintergrundfarben genau angegeben werden.

Probleme gab es vor allem zu Beginn der Umsetzung. Durch die Verschachtelung wurde der Code teilweise zu komplex. Der Code zu Beginn war wesentlich schlechter als der Code kurz vor Abschluss des Projektes.

Ein weiteres Problem war die Umsetzung der Animationen. Die Hamburger Menüs sind CSS Animationen und erforderten viel Aufwand. Das Ausklappen des Seitenmenüs und der Navigationsleiste auf dem Smartphone war sehr aufwändig, da es für eine Vielzahl von verschiedenen Displaygrößen umgesetzt werden musste.

Das Umsetzen der statischen Elemente war nicht schwierig, benötigte jedoch sehr viel Zeit bis alles richtig positioniert war. Als Schriftart für die gesamte Seite wurde Lato verwendet, diese wird von *Google Fonts* in verschiedenen Stärken geladen.

In folgender Tabelle sind die verwendeten Bibliotheken des Designs aufgelistet.

Name	Beschreibung
Bootstrap	Bootstrap ist ein freies <i>CSS-Framework</i> , das Designvorlagen für Grids, Buttons, Tabellen, Formulare und andere Elemente bietet.
Bootstrap Datetimepicker	Date/Time Picker, der auf Bootstrap basiert.
Cropper	Cropper ist ein jQuery Plugin, das eine Bildzuschneide-Funktion implementiert.
Font Awesome	Font Awesome stellt stark anpassbare Icons bereit.
Material Floating Button	Material Floating Button stellt ein schwebendes Menü mit Action Buttons bereit. Es ist im Material Design von Google gehalten.
Normalize	Normalize normalisiert die Darstellung von einer Vielzahl von Elementen.
Select2	Stark anpassbare jQuery Select Box, die mitunter die Implementierung von Tags unterstützt.

Tab. 7 Designbibliotheken

## 6.4 Umgesetztes Design

Das finale Design basiert auf den Entwürfen von Frau Zabrodina, einige Elemente wurden während der Entwicklung selbstständig entworfen und von Frau Zabrodina akzeptiert. Da das Design *responsive* ist, werden folgende Abbildungen sowohl auf einem Desktop, als auch auf einem Smartphone und Tablet dargestellt.

### 6.4.1 Anmeldeseite

Bei der Anmeldeseite ist es neben dem Anmelden möglich, das Passwort zurückzusetzen und ein neues Konto zu erstellen.

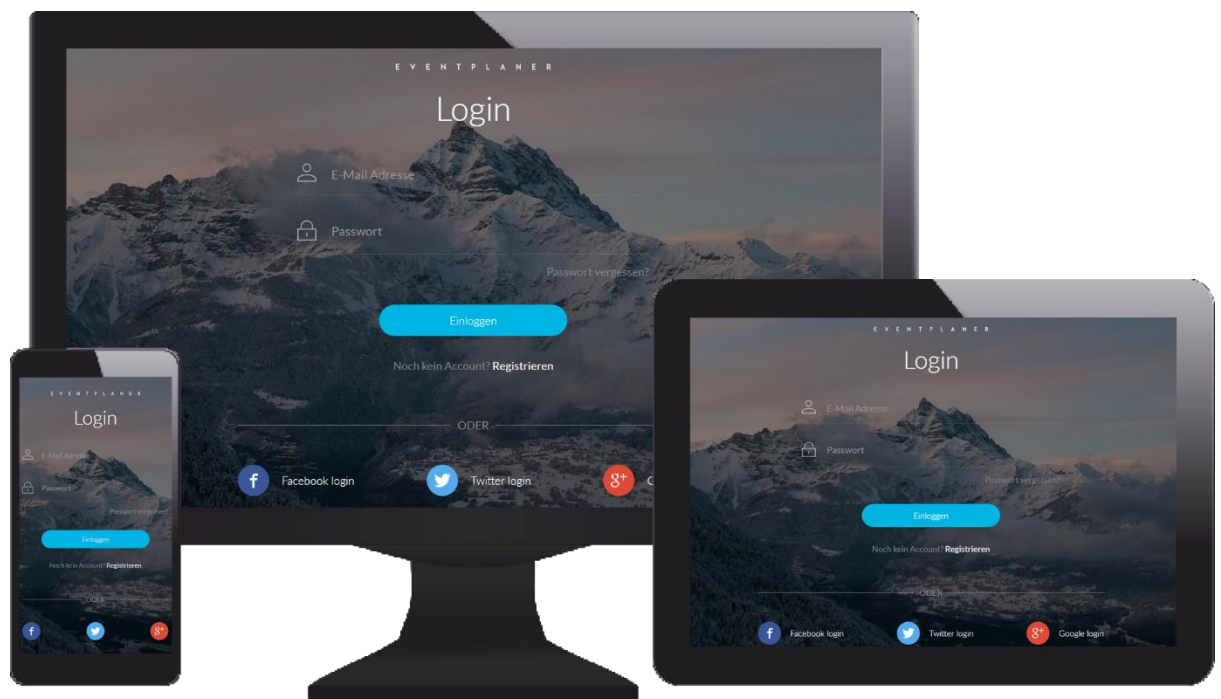


Abb. 20 Screenshots Anmeldeseite

## 6.4.2 Events

Auf der Eventseite sind alle kommenden Events dargestellt. Mit dem Suchsymbol kann man nach einem Event suchen und durch Betätigen des “Details” Buttons wird man zur jeweiligen Eventansicht geleitet.

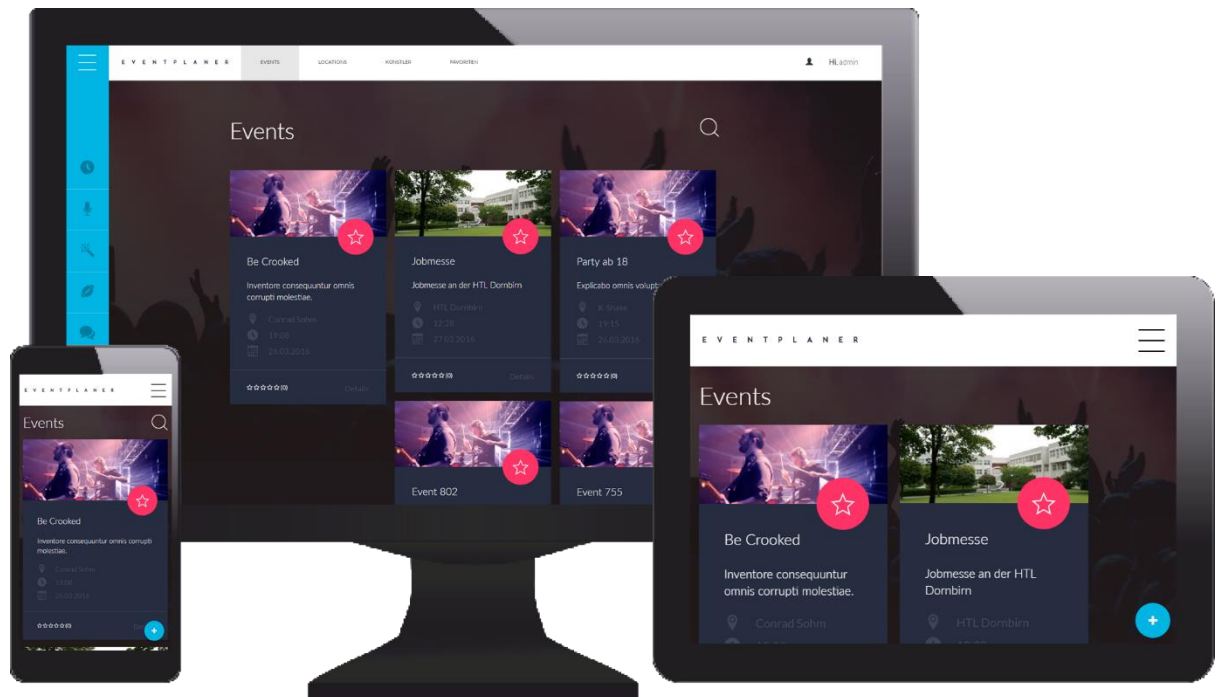


Abb. 21 Screenshots Eventseite

### 6.4.3 Eventansicht

Bei der Eventansicht werden alle Informationen zum ausgewählten Event dargestellt. Außerdem kann der Benutzer am Event teilnehmen, als Favorit hinzufügen und bewerten.

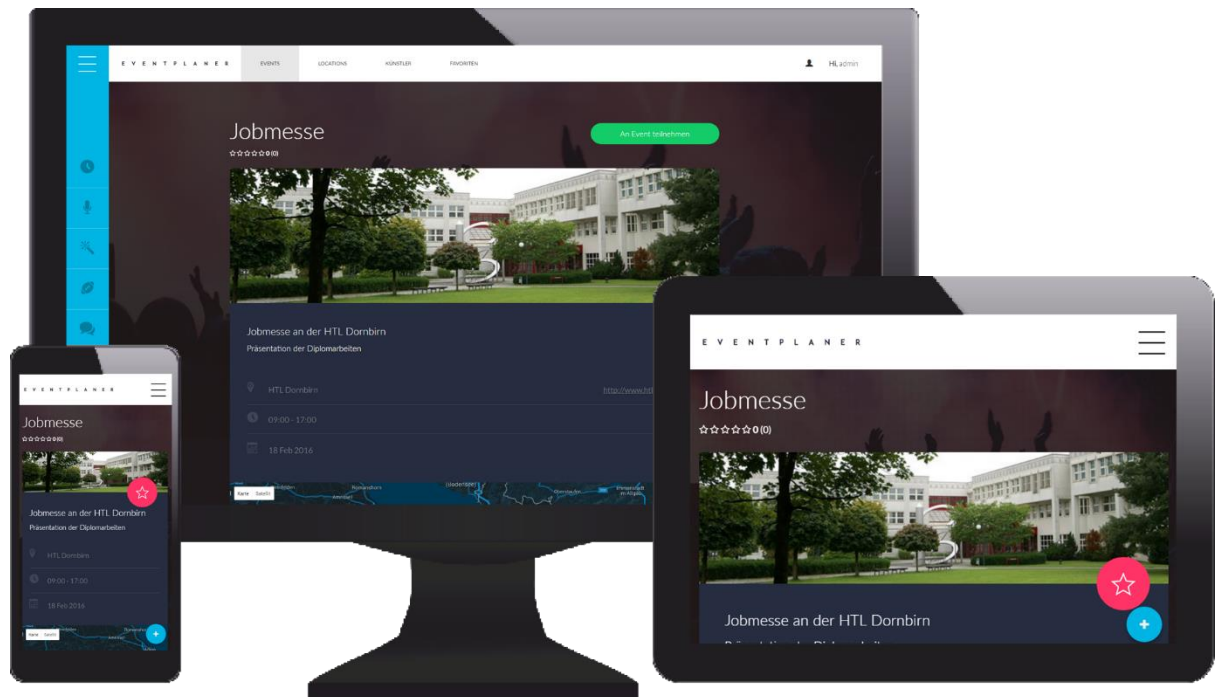


Abb. 22 Screenshots Eventansicht



## 6.4.4 Einstellungen

In den Einstellungen kann jeder angemeldete Benutzer Profil, Newsletter und erstellte Events / Locations und Künstler verwalten. In folgender Abbildung (Abb. 23) ist eine Administratorin angemeldet, welche mehr Einstellungen als ein normaler Benutzer zur Verfügung hat.

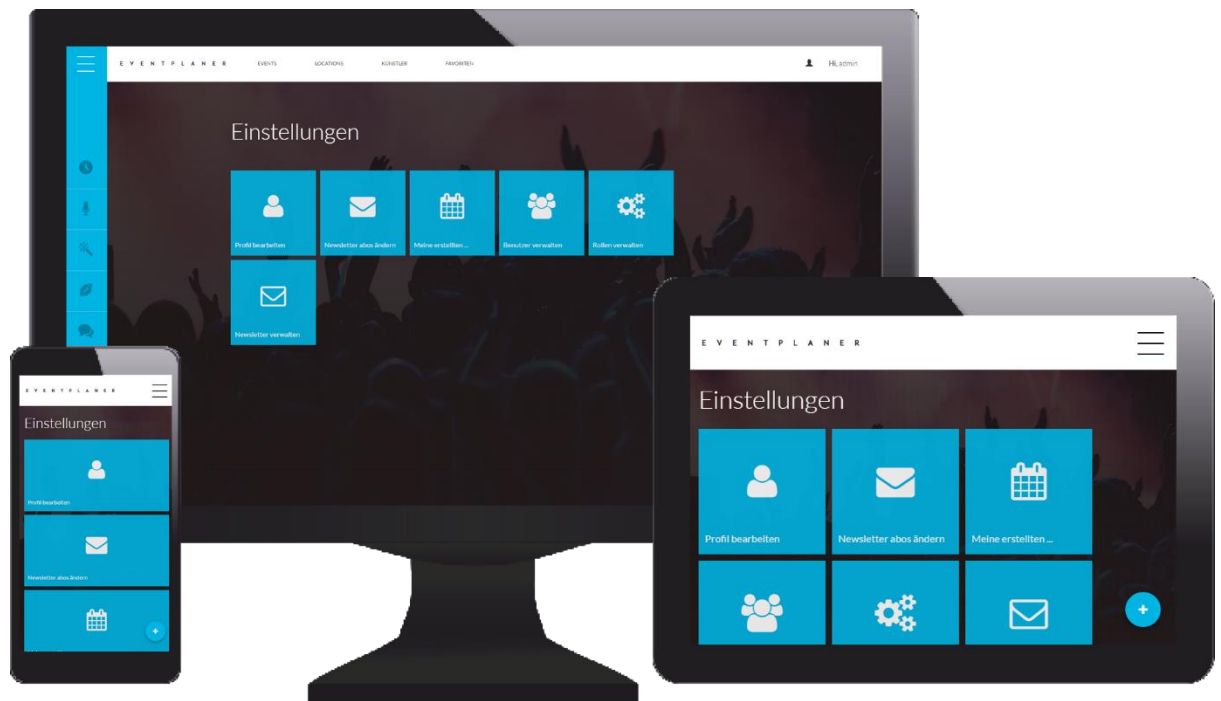


Abb. 23 Screenshots Einstellungen



## 7 Entwicklung / Umsetzung

Die Entwicklung erfolgte in der Programmiersprache *PHP*, es wurde das auf *PHP* basierende *Framework Laravel* verwendet.

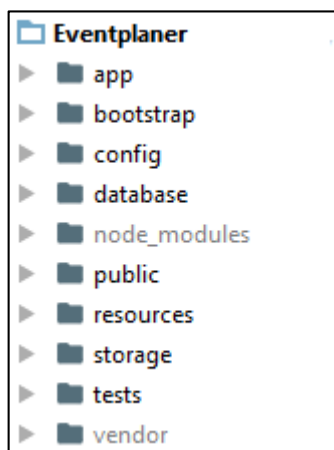
### 7.1 Allgemein

Die Entwicklung in *Laravel* ist sehr strukturiert und auch nicht zu kompliziert, da es dem MVC-Muster (Model View Controller) unterliegt. Es gibt klare Ordnerstrukturen, welche eingehalten werden müssen. Dadurch ergeben sich aber höhere Einarbeitungszeiten. In der Einarbeitungszeit wurden Tutorials gelesen bzw. angesehen. Nach der erfolgreichen Einarbeitung bestanden Grundkenntnisse von *Laravel*. Bei größeren Problemen bei der weiteren Umsetzung konnte die Dokumentation von *Laravel* gelesen werden.

Die Scripts und Stylesheets wurden mithilfe von Gulp (Build Tool) kompiliert und zusammengefügt. Das heißt, dass die einzelnen Sass-Dateien und noch andere benötigten Stylesheets, wie zum Beispiel Bootstrap, in eine CSS-Datei umgewandelt wurden.

Gulp besitzt noch eine weitere hilfreiche Funktion, welche automatisch bei einer Dateiänderung die Dateien neu kompiliert und direkt in den Browser injiziert. Dadurch wird das ständige Neuladen der Seite erspart und die Geschwindigkeit der Programmierung stark beschleunigt.

#### 7.1.1 Ordnerstruktur



Die Abbildung (Abb. 24) zeigt die ganze Ordnerstruktur von *Laravel*. Die wichtigsten Ordner werden in den nächsten Seiten genauer erklärt.

Abb. 24 Ordnerstruktur

### 7.1.1.1 App

Der App Ordner (Abb. 25) enthält die wichtigsten Dateien. Im Grundordner befinden sich die Eloquent Models, wobei jede Datenbanktabelle ein eigenes Model benötigt. Eloquent erleichtert das Arbeiten mit Datenbanktabellen. Es kann durch einen einfachen Befehl die ganze Tabelle abgerufen werden, ohne dabei Standard Query Language (SQL) Code zu verwenden.

Der erste wichtige Ordner ist der Http Ordner. Der Unterordner Controllers enthält die ganzen Controller der Anwendung. Die Controller sind das Grundgerüst der Anwendung. Im Middleware Ordner befinden sich Middlewares, die überprüfen, ob zum Beispiel ein Benutzer angemeldet ist. Im Requests Ordner befinden sich Requests, die die Eingaben von Formularen auf Richtigkeit überprüfen. Es werden Regeln eingetragen, die bei jeder Eingabe überprüft werden. Dadurch lassen sich falsche Eingaben verhindern und es senkt die Rechenleistung auf dem Server.

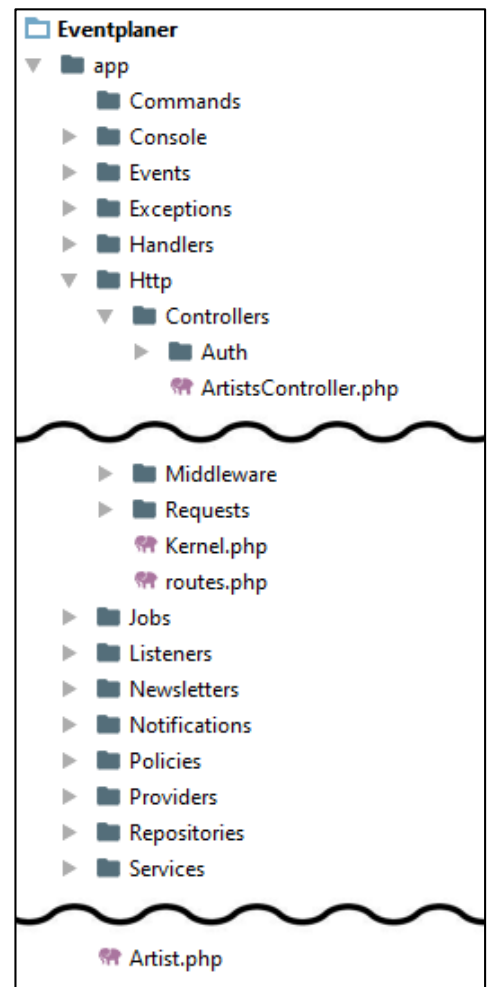


Abb. 25 App Ordner

Im Http Ordner befindet sich auch die „routes.php“ Datei. In dieser Datei wird festgelegt, was passiert, wenn man eine bestimmte Adresse aufruft. Dabei wird meistens eine Klasse in einem Controller aufgerufen, es kann aber auch direkt in dieser Datei programmiert werden.

Im Newsletters Ordner befindet sich die Datei „Newsletterlist“, dies ist ein Interface für die Datei im Unterordner *MailChimp*. Dabei ist das Anmelden und Abmelden von Newslettern enthalten.

Im Notifications Ordner sind Newsletter enthalten. Wenn ein neuer Antrag erstellt wurde, wird eine E-Mail an einen Moderator oder eine Administratorin gesendet. Benutzer, welche einen Antrag gestellt haben, werden über einen Newsletter benachrichtigt, ob deren Antrag akzeptiert oder abgelehnt wurde.

Im Repositories Ordner befindet sich die UserRepository Datei. Sie ermöglicht das Anmelden mit einem sozialen Netzwerk (Facebook, Google+ und Twitter). Dabei wird zuerst überprüft, ob dieser Benutzer bereits existiert, wenn ja, wird er einfach angemeldet und wenn er noch nicht besteht, wird er erstellt und weitergeleitet zum Bestätigen der Berechtigungen bei einem sozialen Netzwerk.

Der Service Ordner enthält den „Appmailer“, er ist zuständig für das eigentliche Absenden der Email Newsletter. Weiteres befindet sich auch noch die Datei zum Eintragen der Events in einen Google Kalender in diesem Ordner.

#### 7.1.1.2 Config

Im config Ordner befinden sich die ganzen Einstellungen zum *Framework*. Dabei gibt es verschiedene Dateien in diesem Ordner, durch die verschiedene Teile der Anwendung veränderbar sind.

#### 7.1.1.3 Database

Im database Ordner gibt es drei verschiedene Unterordner: *factories*, *migrations* und *seeds*. In folgender Tabelle werden diese drei Datei Typen beschrieben.

Typ	Beschreibung
factories	Im factories Ordner befinden sich Dateien zum automatischen Einfügen von Daten in eine Datenbank.
<i>migrations</i>	<i>Migrations</i> erleichtern das Erstellen der Datenbank. Durch <i>Migrations</i> kann auch die Entwicklungsdatenbank sehr leicht übertragen werden. Beim Bearbeiten von einer Tabelle wird eine neue <i>Migration</i> erstellt, die dann alles automatisch erledigt.
seeds	Eine Datenbank kann durch Seeds gefüllt werden. Der Unterschied zu den factories ist, dass man Seeds nur auf dem <i>Artisan</i> CLI ausführen kann. Factories können auch in Funktionen ausgeführt werden.

Tab. 8 Beschreibung der Unterordner des „database“ Ordners

#### 7.1.1.4 Public

Im public Ordner befindet sich der Grundordner des Web-Servers. In ihm befinden sich die kompilierten CSS und JavaScript (JS) Dateien, weiteres befinden sich hier die Bilder für die Anwendung. Hier werden auch die ganzen Titelbilder der verschiedenen Events, Künstler und Locations abgespeichert. Weiters befindet sich hier die index.php Datei. Sie ist der Startpunkt für die Anwendung.

#### 7.1.1.5 Resources

Der resources Ordner enthält fünf Unterordner, diese sind: assets, css, js, lang und views. Im assets Ordner befinden sich alle „scss“ Dateien. Diese werden von Gulp kompiliert und als „all.css“ in den css Ordner gelegt.

Dieser css Ordner enthält auch noch die ganzen css Dateien der verschiedenen Libraries. Diese werden in das all.css kompiliert. Dasselbe gilt für den js Ordner. Dieser Ordner wird in ein „all.js“ kompiliert.

Im view Ordner befinden sich die ganzen Front-End Ansichten. Diese benutzen *Blade* als Vorlage.

#### 7.1.1.6 Tests

Im test Ordner befinden sich die ganzen *PHPUnit Tests*. Diese Tests sind *White-Box-Tests*. Das bedeutet, dass der Code bekannt ist und auf den Code ein Test aufgebaut wird. *Laravel* erleichtert diese Tests. Es gibt Tests bei denen das Erstellen von Events, Künstler und Locations überprüft wird. Weiters gibt es auch noch einen Test zum Überprüfen, ob ein neuer Benutzer erstellt werden kann.

Durch die Konfiguration des Test-Servers wird bei jedem *Merge* mit dem master *Branch* automatisch jeder Test ausgeführt und nur bei erfolgreichem Absolvieren dieser Tests wird es dann auch anschließend veröffentlicht.

## 7.2 Artisan

*Artisan* ist ein Command-line Interface (CLI) Tool, welches bereits in *Laravel* enthalten ist. Mit *Artisan* können nicht nur vorgegebene Befehle ausgeführt werden, es können auch Befehle erstellt werden. Diese Befehle können auch im Code ausgeführt werden, dazu wird die Funktion “call” benötigt. Um alle Befehle von *Artisan* anzuzeigen, muss der Befehl “php artisan list” eingegeben werden.

Die Kommandos, die während der Entwicklung verwendet wurden, sind in folgender Tabelle aufgelistet.

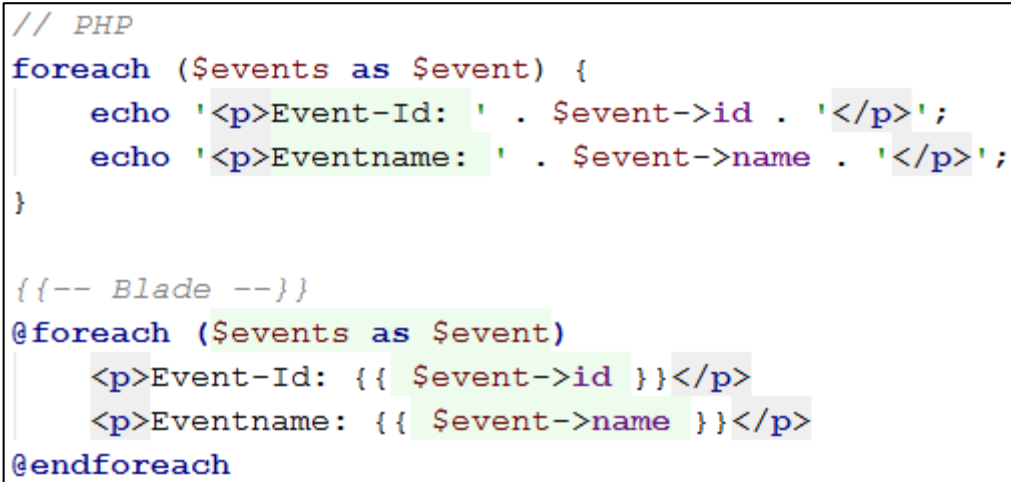
Befehl	Beschreibung
migrate	Führt alle Datenbank <i>Migrations</i> aus.
db:seed	Führt alle Seeds aus.
make:*	Erstellt eine Datei. Dabei kann der Stern durch controller, <i>migration</i> , model, request, seed, etc. ersetzt werden.
routes:list	Zeigt alle <i>REST</i> Routen an.
help	Zeigt Hilfe zu den Befehlen an.

Tab. 9 Beschreibung Artisan Befehle

## 7.3 Blade

*Blade* ist eine Templating Engine, die von *Laravel* bereitgestellt wird. Eine Templating Engine gibt dem Entwickler eine sehr prägnante Syntax, um Daten (z.B. aus einer Datenbank) in einer „view“ darzustellen. Im Fall von *Blade* wird der Code zu *PHP* kompiliert, wodurch auch normaler *PHP* Code neben dem *Blade* Code geschrieben werden kann.

Vergleich zwischen *PHP* und *Blade*

The image shows a code block with a light blue background and a thin black border. It contains two code snippets side-by-side. The left snippet is PHP code for a foreach loop, and the right snippet is Blade templating engine code for the same loop. The PHP code uses 'foreach', 'echo', and string concatenation. The Blade code uses '@foreach', '@endforeach', and Blade's interpolation syntax {{ }}. Both snippets iterate over '\$events as \$event' and output the event's ID and name, each on a new line.

```
// PHP
foreach ($events as $event) {
    echo '<p>Event-Id: ' . $event->id . '</p>';
    echo '<p>Eventname: ' . $event->name . '</p>';
}

{{-- Blade --}}
@foreach ($events as $event)
    <p>Event-Id: {{ $event->id }}</p>
    <p>Eventname: {{ $event->name }}</p>
@endforeach
```

Abb. 26 Vergleich zwischen *PHP* und *Blade* Syntax

In obiger Abbildung (Abb. 26) wird sowohl mit *PHP* als auch *Blade* eine Foreach Schleife geschrieben. Bei dieser Schleife wird eine Liste von Events durchlaufen und von jedem einzelnen Event in der Liste die ID und der Name ausgegeben.

## 7.4 Datenbank - MySQL

Als Datenbank wird das relationale Datenbanksystem MySQL verwendet. In der Datenbank werden sämtliche Informationen über alle Events, Locations, Künstler und Benutzer in eigenen Tabellen abgespeichert und sind über Relationen miteinander verknüpft.

### 7.4.1 Verwendung von MySQL

Hauptgrund MySQL zu verwenden war, dass alle Projektmitglieder über ein Jahr Erfahrung mit dem Datenbanksystem hatten und daher keine Einarbeitung notwendig war. Ein weiterer Faktor war, dass *Laravel* die Arbeit mit einem relationalen Datenbanksystem erheblich erleichtert. *Laravel* bietet das Kommandozeilen-Tool *Artisan* und *Datenbank Migrations* an. Eine *Migration* ist ein *PHP* Dokument, in welches Anweisungen zur Erstellung, Bearbeitung und/oder Löschung von Datenbanktabellen geschrieben werden. *Migrations* werden dann in *Artisan* per Befehl ausgeführt.

## 7.4.2 Entity Relation-Modell

Ein Entity Relation-Modell (ER) ist eine grafische Darstellung einer relationalen Datenbank. In ihm werden einerseits Datenbanktabellen mit ihren Attributen und andererseits die Relationen zwischen den Tabellen dargestellt. Das Modell dient grundsätzlich dazu einen groben Überblick einer Datenbank zu geben. In der MySQL Workbench kann mit Reverse Engineering automatisch aus einer bestehenden Datenbank das zugehörige ER-Modell generiert werden.

Die größten und zentralen Tabellen der Datenbank sind die „events“, „users“, „locations“ und „artists“ Tabellen. Diese sind, um verschiedene Programmfunktionen zu realisieren, mehrfach direkt oder über Zwischentabellen verknüpft. Beispielsweise gibt es zwischen „events“ und „users“ eine „event\_favorite“ Tabelle, mit der ein Event einem Benutzer zugeordnet wird.

### 7.4.2.1 Events Tabelle

In der „events“ Tabelle werden alle Informationen über ein Event gespeichert. Die Attribute eines Events sind in der Grafik zu sehen.

Wichtige Attribute sind:

- Name des Events
- Kurze und normale Beschreibung
- Start- und Enddatum
- User- und Location ID, über die das Event klar einem User bzw. einer Location zugeordnet wird
- Release, mit dem der Zustand des Events gekennzeichnet wird
- Private, mit dem gekennzeichnet wird, ob das Event privat ist

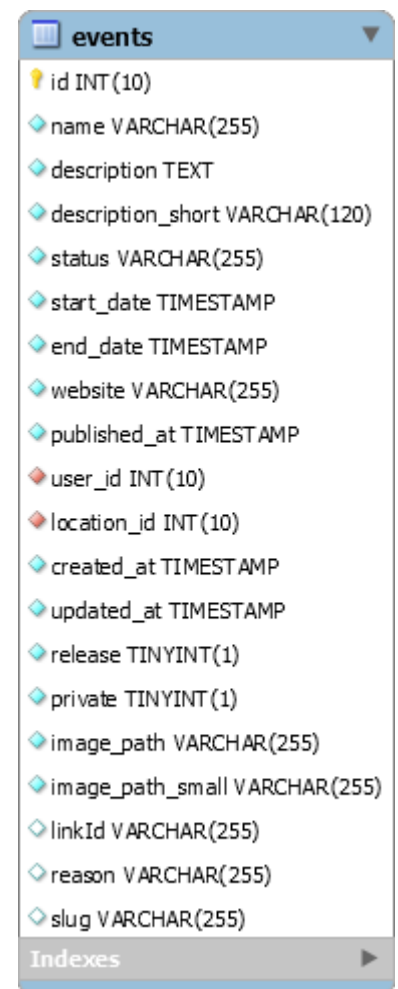


Abb. 27 Events-Tabelle



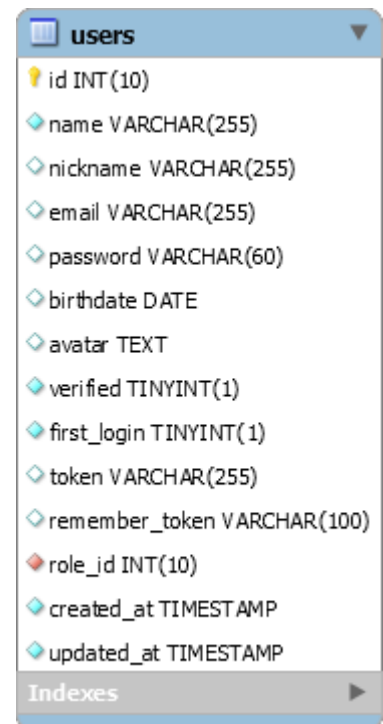
#### 7.4.2.2 Users Tabelle

In der „users“ Tabelle werden Informationen über einen Benutzer gespeichert.

Die Rolle des Benutzers wird über einer weiteren Tabelle „roles“ gespeichert, die mit der Benutzertabelle verknüpft ist.

Wichtige Attribute sind:

- Name und Nickname
- Email, die zur Anmeldung verwendet wird
- Passwort
- Verified, mit dem gekennzeichnet wird, ob die Benutzerregistrierung bestätigt wurde



users	
id	INT(10)
name	VARCHAR(255)
nickname	VARCHAR(255)
email	VARCHAR(255)
password	VARCHAR(60)
birthdate	DATE
avatar	TEXT
verified	TINYINT(1)
first_login	TINYINT(1)
token	VARCHAR(255)
remember_token	VARCHAR(100)
role_id	INT(10)
created_at	TIMESTAMP
updated_at	TIMESTAMP
Indexes	

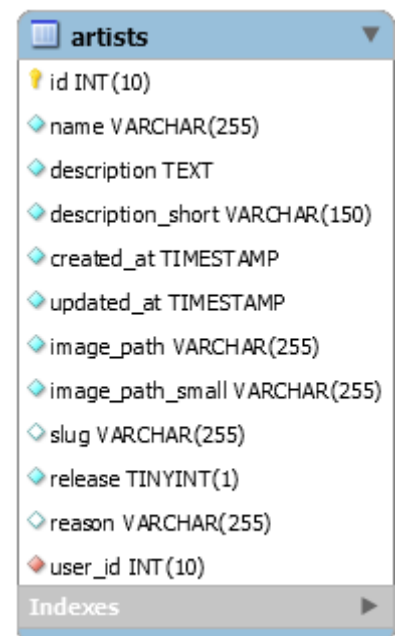
Abb. 28 Users-Tabelle

#### 7.4.2.3 Artists Tabelle

Die „artists“ Tabelle enthält Informationen über alle Künstler oder Gruppen, die bei Events auftreten.

Wichtige Attribute sind:

- Name
- Kurze und normale Beschreibung



artists	
id	INT(10)
name	VARCHAR(255)
description	TEXT
description_short	VARCHAR(150)
created_at	TIMESTAMP
updated_at	TIMESTAMP
image_path	VARCHAR(255)
image_path_small	VARCHAR(255)
slug	VARCHAR(255)
release	TINYINT(1)
reason	VARCHAR(255)
user_id	INT(10)
Indexes	

Abb. 29 Artists-Tabelle

#### 7.4.2.4 Locations Tabelle

Die „locations“ Tabelle enthält alle Informationen über die Locations, an denen Events stattfinden.

Wichtige Attribute sind:

- Name
- Kurze und normale Beschreibung
- Longitude und Latitude, die für die Implementierung von Google Maps benötigt werden
- Country-, State, City- und Street-ID, über die mittels anderen Tabellen die Adresse genau gespeichert wird.

locations	
id	INT(10)
name	VARCHAR(255)
description	TEXT
description_short	VARCHAR(150)
lng	DOUBLE
lat	DOUBLE
country_id	INT(10)
state_id	INT(10)
city_id	INT(10)
street_id	INT(10)
created_at	TIMESTAMP
updated_at	TIMESTAMP
image_path	VARCHAR(255)
image_path_small	VARCHAR(255)
slug	VARCHAR(255)
release	TINYINT(1)
reason	VARCHAR(255)
user_id	INT(10)
Indexes	

Abb. 30 Locations-Tabelle

## 7.5 Benutzerverwaltung

Die Benutzerverwaltung umfasst mehrere Punkte, welche benötigt werden, um den Benutzern eine große Möglichkeit von Optionen zu bieten. Diese Optionen unterscheiden sich auch von Benutzer zu Benutzer, da diese in bestimmte Rollen eingeteilt werden.

### 7.5.1 Autorisierung

Es gibt eine Einteilung in vier verschiedene Benutzerrollen. Wenn sich ein neuer Benutzer registriert, wird dieser automatisch in die Gruppe "Angemeldeter Benutzer" eingeteilt.

Nicht-angemeldet	Angemeldeter Benutzer	Moderator	Administratorin
Events ansehen	Eventanträge erstellen	Direktes Erstellen von Events	Benutzer verwalten
Kostenlos registrieren	Eigene Events bearbeiten und entfernen	Eventanträge verwalten	
	Events als Favorit speichern		
	Events kommentieren		

Tab. 10 Benutzerrollen

Nur Administratoren können die Rolle eines Benutzers ändern.

## 7.5.2 Registrierung

Die Registrierung erfolgt per E-Mail-Adresse und einem Passwort. Das Passwort der Benutzer wird mit einer Hashfunktion verschlüsselt. Optional ist eine Registrierung mit einem bestehenden Konto von Facebook, Google+ oder Twitter möglich.

### 7.5.2.1 Registrierung über Facebook, Google+ und Twitter

Für die Registrierung mit einem vorhandenen Konto bei einem dieser sozialen Netzwerke, muss über dessen Entwicklerseite ein Application Programming Interface(API)-Key angefordert werden. Für den Austausch wird eine Erweiterung von *Laravel* namens *Socialite* verwendet. Mithilfe dieser Erweiterung wird der Austausch von Tokens über das *OAuth* Protokoll, welche für die Authentifizierung benötigt sind, erleichtert. Bei einer erfolgreichen Registrierung werden die Daten des Benutzers (ID, Name, Email-Adresse, Profilbild, etc.) in die Datenbank gespeichert.

Um die Anmeldung mit *OAuth* darzustellen wurde folgendes Sequenzdiagramm (Abb. 31) erstellt.

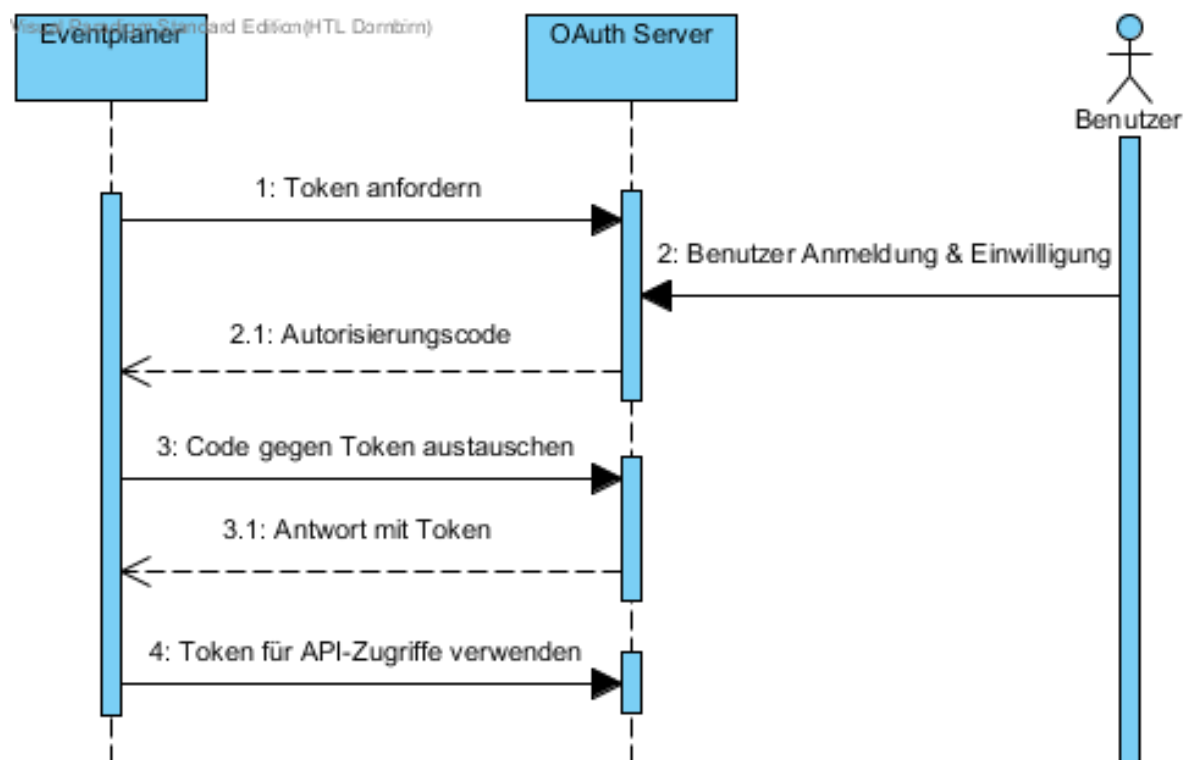


Abb. 31 Anmeldung Sequenzdiagramm

Im Sequenzdiagramm ist folgender Ablauf dargestellt. Zuerst fordert die Webanwendung einen Token vom entsprechenden *OAuth* Server an. Wenn die Webanwendung eine Berechtigung für einen Token besitzt (API-Key), wird der Benutzer auf eine Bestätigungsseite weitergeleitet. Bei dieser Seite muss sich der Benutzer gegebenenfalls anmelden (z.B. bei Google+) und die Berechtigungen der Webanwendung (z.B. Zugriff auf die E-Mail-Adresse) zulassen. Falls der Benutzer nicht mit diesen Berechtigungen einverstanden ist, wird die Anmeldung / Registrierung abgebrochen und zur Anmeldeseite geleitet. Bei einer Bestätigung der benötigten Berechtigungen wird ein Autorisierungscode vom *OAuth* Server generiert und der Webanwendung zugestellt. Um zu gewährleisten, dass der Autorisierungscode an die richtige Anwendung gesendet wurde, muss die Anwendung diesen Code mit dem bereits erhaltenen Token austauschen. Darauf wird ein neuer Token zugestellt, welcher dann für weitere Zugriffe verwendet werden kann (z.B. Google Kalender). Falls während des Ablaufs ein Fehler auftritt, wird der Benutzer immer auf die Anmeldeseite geleitet.

#### **7.5.2.2 Verlinkung mit mehreren sozialen Netzwerken**

Es besteht die Möglichkeit, nach der Registrierung (sei es mit einer Email-Adresse oder einem vorhandenen Konto eines sozialen Netzwerkes) ein weiteres Konto mit diesem zu verknüpfen. Dies ermöglicht die Anmeldung mit einem Klick.

### 7.5.3 Authentifizierung

Die Authentifizierung ist über zwei Arten möglich. Zum einen über eine E-Mail-Adresse und deren zugewiesenem Passwort und zum anderen über einen Account von Facebook, Google+ oder Twitter.

#### 7.5.3.1 Authentifizierung mit E-Mail-Adresse und Passwort

Ein Überblick über die Anmeldung mit E-Mail-Adresse und Passwort bietet folgendes Aktivitätendiagramm (Abb. 32).

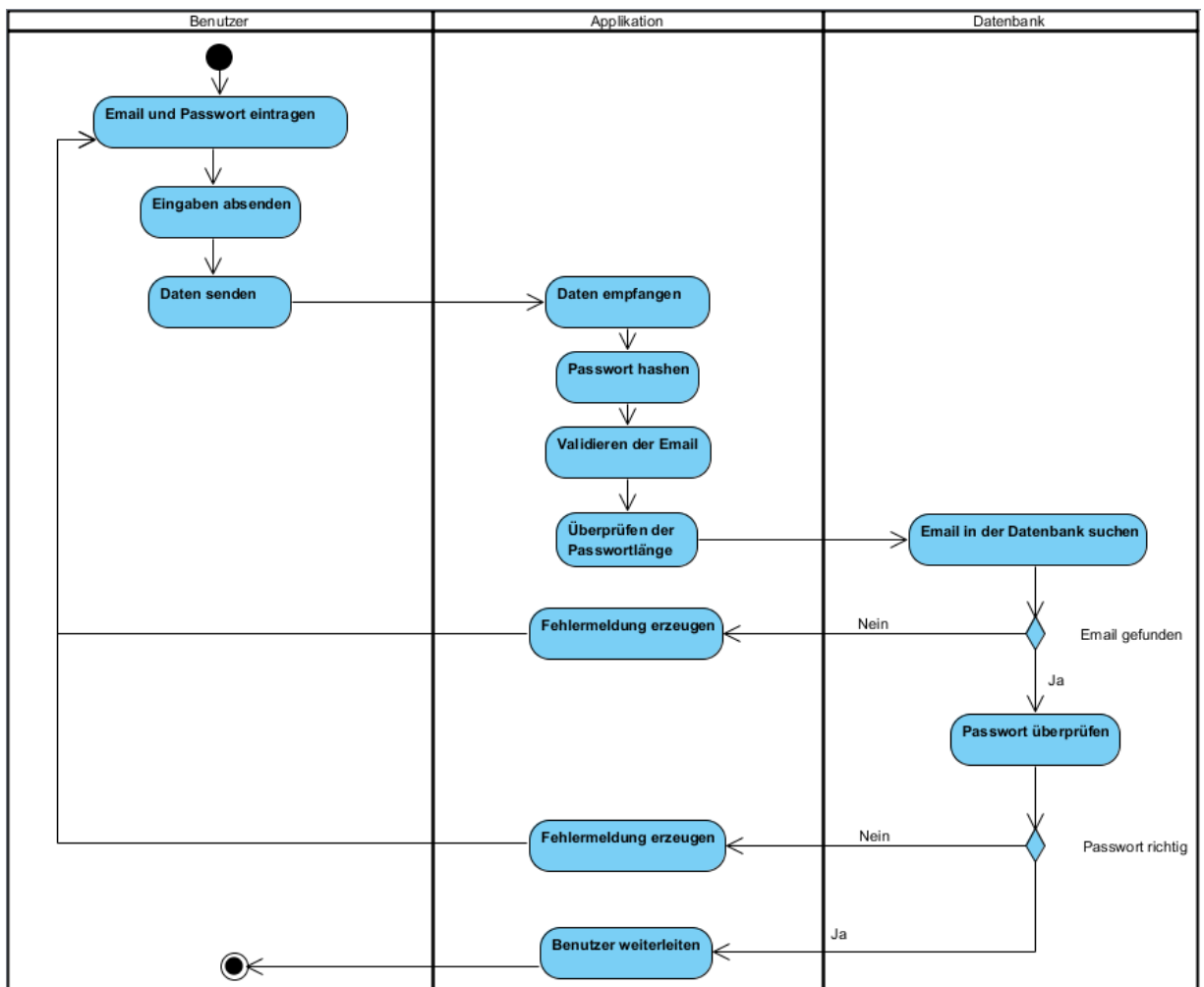


Abb. 32 Aktivitätsdiagramm - Anmeldung mit E-Mail-Adresse und Passwort

Der Ablauf der Anmeldung mit E-Mail und Passwort ist wie folgt. Die Daten des Benutzers werden an die Applikation gesendet und überprüft, ob das Format der E-Mail-Adresse gültig ist und das Passwort mindestens sechs Zeichen lang ist. Sind die Eingaben korrekt, so wird überprüft, ob ein Benutzer mit der eingegeben E-Mail-Adresse in der Datenbank vorhanden ist. Falls die E-Mail-Adresse vorhanden ist und das

eingeebene Passwort übereinstimmt, wird der Benutzer angemeldet und weitergeleitet. Tritt bei einer der Eingaben jedoch ein Fehler auf, so wird der Benutzer informiert, dass die Anmeldung fehlgeschlagen ist.

#### **7.5.3.2 Authentifizierung mit Facebook, Google+ oder Twitter**

Bei der Authentifizierung mit Facebook, Google+ oder Twitter wird der gleiche Ablauf wie bei der Registrierung durchgeführt (Abb. 31). Der Unterschied liegt darin, dass zuerst die vorhandenen Daten des Benutzers mit denen aus dem sozialen Netzwerk verglichen werden und kein neuer Benutzer erstellt wird. Verglichen werden die Daten anhand der ID des Kontos auf dem sozialen Netzwerk. Falls ein Unterschied in den Daten erkannt wird, werden die Daten in der Datenbank mit den neuen ersetzt. Zuletzt wird der Benutzer angemeldet, falls keine Fehler auftreten.

#### **7.5.4 Passwort vergessen**

Falls ein Benutzer das Passwort vergessen hat, kann dieses zurückgesetzt werden. Es wird ein Token generiert, welcher per E-Mail an den Benutzer geschickt wird. Öffnet der Benutzer den Link in der E-Mail, wird er auf eine Seite weitergeleitet, bei der ein neues Passwort angegeben werden kann.

#### **7.5.5 Interessenverwaltung**

Nach der ersten Anmeldung können Interessen festgelegt werden. Dies erfolgt über eine optionale Angabe des Geburtsdatums und des Angebens für welche Eventkategorien man sich interessiert. Dadurch wird versucht, die Startseite auf den Benutzer anzupassen. D.h. es werden vorrangig Events, die den Interessen entsprechen angezeigt.

#### **7.5.6 Profil bearbeiten**

Auf der „Profil bearbeiten“ Seite kann ein angemeldeter Benutzer persönliche Daten wie Name, Nickname und Geburtsdatum bearbeiten. Weiters können hier die Interessen geändert oder ein neues Passwort gesetzt werden.

## 7.6 Eventverwaltung

Für das Verwalten verschiedener Events wird der EventsController benötigt. Der EventsController enthält alle Funktionen die zur Verwaltung von Events benötigt werden. Das Erstellen eines Controllers wird durch *Laravel* wesentlich erleichtert. Zuerst werden in der routes.php Datei die Representational State Transfer(*REST*)-Befehle eingetragen. Anschließend kann einfach der Befehl „php artisan make:controller EventsController“ in *Artisan* ausgeführt werden. Dadurch wird alles automatisch eingerichtet. Die Datei „EventsController.php“ wird in dem richtigen Ordner erstellt und es werden „plain“ Klassen, also nur der Klassenname, erstellt.

### 7.6.1 Erstellen von Events

Die Seite für das Erstellen der Events wird im EventsController mit der Funktion „create“ aufgerufen. Anschließend wird eine „view“ geladen, in dieser „view“ können die Daten für das Event eingegeben bzw. ausgewählt werden. Bei Drücken des „Event hinzufügen“ Buttons werden die Daten auf mögliche Fehler in der Eingabe überprüft und falls keine Fehler auftreten wird dann die „Store-Funktion“ im EventsController aufgerufen. Mit dieser Funktion werden alle eingegebenen Daten in die Datenbank gespeichert und auch das Titelbild des Events gespeichert. Das Bild wird im Public Ordner unter Images gespeichert.



Folgende Ansicht (Abb. 33) stellt das Formular zum Erstellen eines Eventantrags dar. Die Ansichten für das Erstellen der Künstler und Locations sehen ähnlich aus.

## Neuer Erstantrag

**Name** Name

**Ort** Location auswählen [Gewünschte Location nicht in der Liste?](#)

**Künstler** Künstler auswählen [Gewünschter Künstler nicht in der Liste?](#)

**Kategorie** Kategorie auswählen

**Beschreibung**

Geben Sie hier eine Beschreibung ein

Abb. 33 Eventantrag erstellen

Eine grobe Übersicht über den Ablauf der Eventerstellung liefert folgendes Aktivitätsdiagramm (Abb. 34).

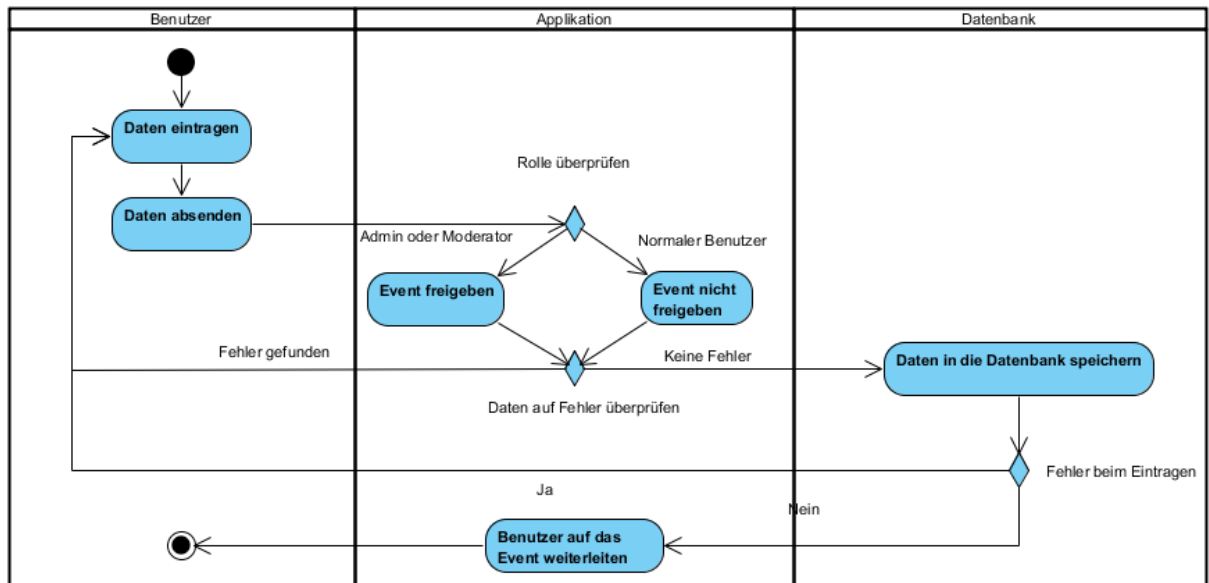


Abb. 34 Event erstellen Aktivitätsdiagramm

Um ein Event zu erstellen wird folgender Ablauf durchlaufen. Als Beispiel wird das Speichern des Namens verwendet.

Als Erstes wird in der „view“ in das Textfeld mit dem Namen „name“ der Name eingetragen. Dies ist in folgendem Ausschnitt (Abb. 35) erkennbar, hier wird auch die Template-Engine *Blade* verwendet.

```

<div class="form-group">
    {!! Form::label('name', 'Name', ['class' => 'col-sm-2']) !!}
    {!! Form::text('name', old('name'), ['class' => 'form-control',
        'placeholder' => 'Name', 'maxlength' => 50 ]) !!}
</div>
    
```

Abb. 35 Event erstellen - Formularausschnitt

Nach dem Einreichen des Formulars wird dieser „request“, auch Anfrage genannt, mithilfe von „EventsRequest“ überprüft. Dabei werden die Regeln überprüft, die auf folgendem Ausschnitt erkennbar sind (Abb. 36).

```
public function rules()
{
    return [
        'name' => 'required',
        'description' => 'required|min:5',
        'description_short' => 'required|min:5',
        'location_id' => 'required',
        'start_date' => 'required|date',
        'end_date' => 'required|date|after:start_date'
    ];
}
```

Abb. 36 EventRequest Regeln Codeauszug

Anschließend wird der eigentliche Controller aufgerufen. Dieser bearbeitet die Request. Dabei wird auf dem Bild ein neues Event erstellt, das dem aktuell angemeldeten Benutzer zugeordnet ist. Es wird alles aus der Request in die Datenbank gespeichert (Abb. 37).

```
public function store(EventRequest $request)
{
    try {
        // Create the event
        $event = auth()->user()->events()->create($request->all());
    }
}
```

Abb. 37 Speicherfunktion Codeauszug

Zum Schluss wird im Event Model noch überprüft, ob auch nur Daten übergeben wurden, die in die Datenbank eingetragen werden dürfen (Abb. 38). Wenn in den Daten eine Zeile vorhanden ist, die nicht einem Namen im Bild entspricht, wird das Event nicht erstellt und die bereits eingetragenen Daten werden wieder gelöscht.

```
protected $fillable = [  
    'name', 'description', 'description_short', 'status',  
    'start_date', 'end_date', 'website', 'published_at', 'user_id',  
    'location_id', 'release', 'private', 'link_id'  
];
```

Abb. 38 Füllbare Felder Codeauszug

### 7.6.2 Bearbeiten von Events

Das Bearbeiten von Events ist grundsätzlich dasselbe wie das Erstellen von Events. Es wird aber zuerst der bereits vorhandene Datensatz aus der Datenbank geladen und in die Hypertext Markup Language (*HTML*) „view“ eingefügt. Nach dem Bearbeiten wird die „Update-Funktion“ aufgerufen, die alle Daten in der Datenbank speichert.

### 7.6.3 Löschen von Events

Das Löschen von Events ist möglich für den Ersteller des Events und für Administratorinnen. Beim Betätigen des Löschen Buttons wird die „Delete-Funktion“ im EventsController aufgerufen, die den Übergabewert der Event ID hat. Diese Event ID wird benutzt, um den richtigen Datensatz aus der Datenbank zu löschen.

## 7.7 Locationverwaltung

Locations sind Orte, bei denen Events stattfinden. Diese Locations können von normalen Benutzern zusätzlich bei der Eventerstellung erstellt werden. Administratorinnen und Moderatoren sind in der Lage, eine Location bei einer eigenen "view" zu erstellen.

### 7.7.1 Erstellen von Locations

Das Formular zur Erstellung von Locations wird bei der Create-Funktion im LocationsController sowie beim EventsController aufgerufen. In diesem Formular ist die Google Maps API eingebunden, welche es dem Benutzer ermöglicht, nach einem Standort zu suchen. Das Ergebnis der Suche wird dann automatisch in die betreffenden Felder (Straße, Ort, Längengrad, Breitengrad, etc.) gefüllt. Übergeben werden die Daten indem die „Store-Funktion“ im LocationsController aufgerufen wird. Ein Validator überprüft vor dem Speichern, ob die Formate richtig eingehalten wurden und speichert diese darauf, sofern bei der Überprüfung keine Fehler auftreten.

### 7.7.2 Bearbeiten von Locations

Zur Bearbeitung einer Location wird dessen *Slug* der Edit-Funktion im LocationsController übergeben. Die Funktion gibt auf Basis des übergebenen *Slugs* die betreffende Location in der „Bearbeitungs-View“ zurück. In dieser "view" ist das gleiche Formular wie bei der Erstellung, jedoch werden die Eingabefelder automatisch mit der ausgewählten Location gefüllt. Durch das Betätigen des Bearbeiten Buttons werden die neuen Daten an die „Update-Funktion“ übergeben und der Datensatz aktualisiert.

### 7.7.3 Löschen von Locations

Es können nur Administratorinnen und Moderatoren Locations löschen. Diese können eine Location entfernen, indem sie bei der Übersicht der gewünschten Location auf den Button "Location entfernen" klicken. Darauf öffnet sich ein *Modal*, bei dem nochmals nachgefragt wird, ob die Location wirklich gelöscht werden soll. Nachdem diese Meldung akzeptiert wurde, wird die „Destroy“ Funktion im Controller aufgerufen, welche die Location inkl. deren Beziehungen zu anderen Datensätzen in der Datenbank löscht.

## 7.8 Künstlerverwaltung

Künstler sind Personen oder Gruppen, die bei einem Event auftreten. Sie können beim Erstellen eines Events zum Event hinzugefügt werden.

### 7.8.1 Erstellen von Künstlern

Das Erstellen von Künstlern erfolgt über die View zum Erstellen von Künstlern. Diese wird mit der „Create-Funktion“ im ArtistsController aufgerufen. In der „Create-View“ müssen alle benötigten Künstlerinformationen eingegeben werden. Anschließend wird der Künstler durch Klicken des „Künstler hinzufügen“ Buttons erstellt. Dabei wird die „Store-Funktion“ im ArtistsController aufgerufen, die die Eingabe überprüft und, wenn alle Eingaben korrekt sind, den Künstler in die Datenbank speichert.

### 7.8.2 Bearbeiten von Künstlern

Beim Bearbeiten von Künstlern wird mit Übergabe der ID ein vorhandener Datensatz aus der Datenbank geladen. Die Edit-View ist die gleiche wie die Create-View. Unterschied ist, dass die Formularelemente mit den jeweiligen Daten des Künstlers gefüllt werden. Der Künstler wird dann durch Klicken des „Künstler aktualisieren“ Buttons aktualisiert. Dabei wird die „Update-Funktion“ im ArtistsController aufgerufen, die die Eingaben überprüft und den Künstler in der Datenbank aktualisiert.

### 7.8.3 Löschen von Künstlern

Beim Löschen von Künstlern wird die „Delete-Funktion“ im ArtistsController aufgerufen. Diese hat die ID des Künstlers als Übergabewert und löscht mit ihr den Künstler aus der Datenbank. Dabei werden auch alle vom Künstler abhängigen Einträge aus anderen Tabellen gelöscht.

## 7.9 Bilderverwaltung

Es können bei Events, Locations und Künstlern Titelbilder hinzugefügt werden. Diese Bilder müssen ein bestimmtes Seitenverhältnis haben, wodurch es vor dem Hinzufügen zugeschnitten werden muss.

Um die Benutzerfreundlichkeit zu erhöhen, kann der Benutzer selbst auswählen, welcher Bereich ausgeschnitten werden soll. Dies funktioniert durch das auf JQuery basierende Plugin Cropper und der GD-Bibliothek von *PHP*.

Nachdem der Benutzer ein Bild ausgewählt hat, öffnet sich automatisch ein Fenster (*Modal*), bei dem das Bild zugeschnitten werden kann. Der Benutzer wird aufgefordert, das Bild zweimal, mit zwei unterschiedlich festgelegten Seitenverhältnissen, zuzuschneiden. Es werden zwei unterschiedliche Seitenverhältnisse benötigt, da eine unterschiedliche Größe für das Titelbild als für das Bild auf einer Karte verwendet wird.

In Abbildung 39 ist das *Modal* mit dem Zerschneidungs-Tool dargestellt.

Wenn der Benutzer das Bild zweimal zugeschnitten hat, werden die Daten des Bildes (Größe, Breite, etc.) mit dem Formular an den Controller übermittelt. Im Controller werden die Bilder mithilfe von *PHP-GD* an den festgelegten Stellen zugeschnitten und in einen entsprechenden Ordner auf dem Server gespeichert. Der Pfad zu diesem Ordner hängt vom Typ (Event, Location, Künstler) und dessen Name ab, wie zum Beispiel: „/public/images/events/bildtest“

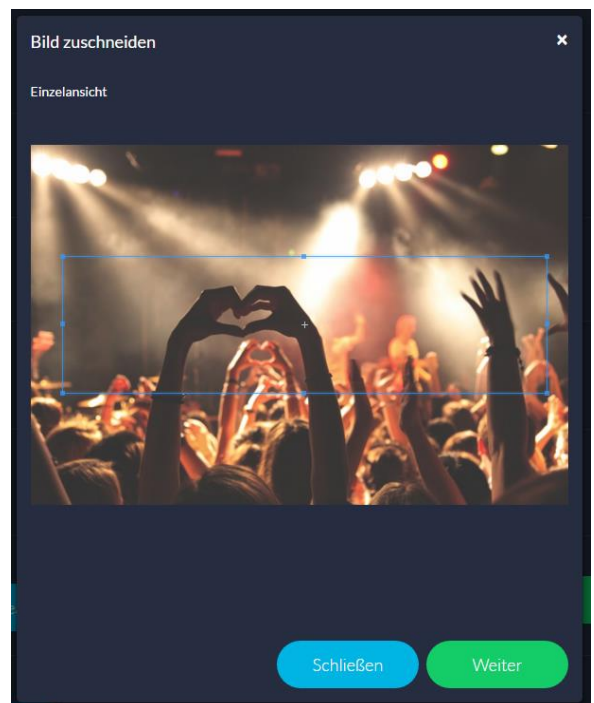


Abb. 39 Bild zuschneiden



## 7.10 Anfragenverwaltung

Wenn ein angemeldeter Benutzer ein Event, eine Location oder einen Künstler erstellt, muss dies von einem Moderator oder einer Administratorin freigegeben werden. Wenn eine Administratorin oder ein Moderator etwas erstellt, wird dies automatisch freigegeben.

Gelöst wurde dies durch eine Spalte in der Datenbanktabelle. Dabei gibt es drei mögliche Werte.

1. Noch nicht freigegeben
2. Freigegeben
3. Abgelehnt

Die Zustände, die ein Event während der Anfrage durchlaufen kann, werden in folgendem Zustandsdiagramm (Abb. 40) dargestellt.

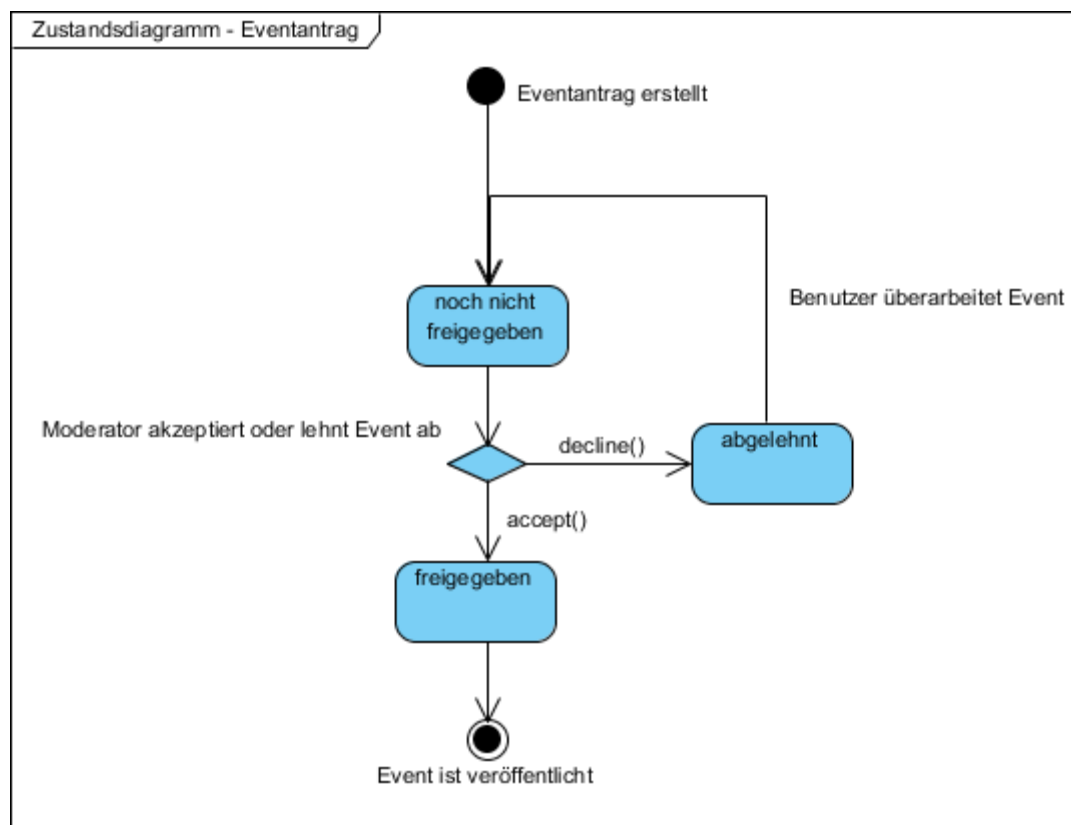


Abb. 40 Zustandsdiagramm - Eventantrag

Für die Moderatoren besteht die Möglichkeit, einen Ablehnungsgrund anzugeben. Dieser Ablehnungsgrund wird dem zuständigen Benutzer per E-Mail gesendet. Der Ersteller des Antrags kann darauf diesen überarbeiten. Der Auftrag wird anschließend erneut, an die Moderatoren und Administratorinnen übergeben.

Moderatoren und Administratorinnen sind bei der Ansicht eines Antrags in der Lage, mögliche Teile des Antrags individuell zu verwalten. D.h., wenn ein Eventantrag mit noch nicht vorhandenen Künstlern bzw. nicht vorhandener Location erstellt wird, können beispielsweise nur die Künstler angenommen werden.

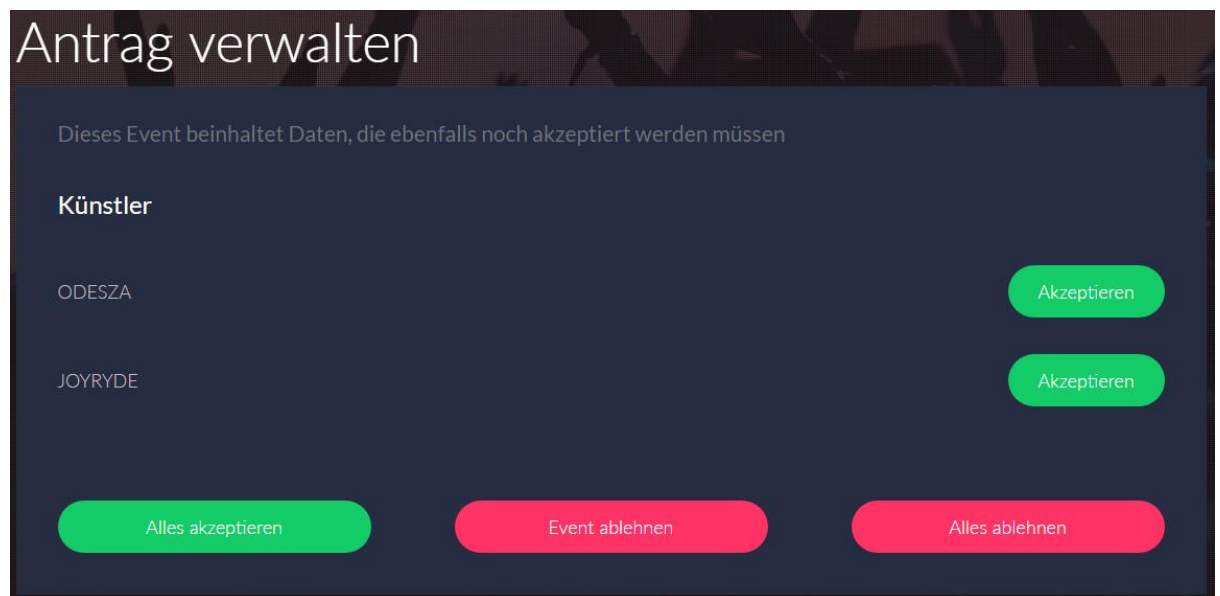


Abb. 41 Antrag verwalten

## 7.11 Suche und Filter

Um das Finden von bestimmten Events, Locations und Künstlern zu erleichtern, wurden verschiedene Suchfunktionen und Filter entwickelt. Es gibt eine generelle Schnellsuche für die ganze Seite und speziellere Suchen und Filter für Events, Locations und Künstler.

### 7.11.1 Bearbeiten von Künstlern

Auf der Startseite gibt es oben eine Schnellsuche. Durch Klicken auf den Such-Button (Lupe rechts oben) erscheint ein Suchfeld. Mit dem Suchfeld kann nach Events, Locations und Künstlern gesucht werden kann. Dazu muss ein Suchwert eingegeben werden. Suchtreffer werden darauf automatisch per *AJAX*-Call geladen.

In folgender Abbildung (Abb. 42) ist die Schnellsuche dargestellt, wobei nach „Event“ gesucht wird.

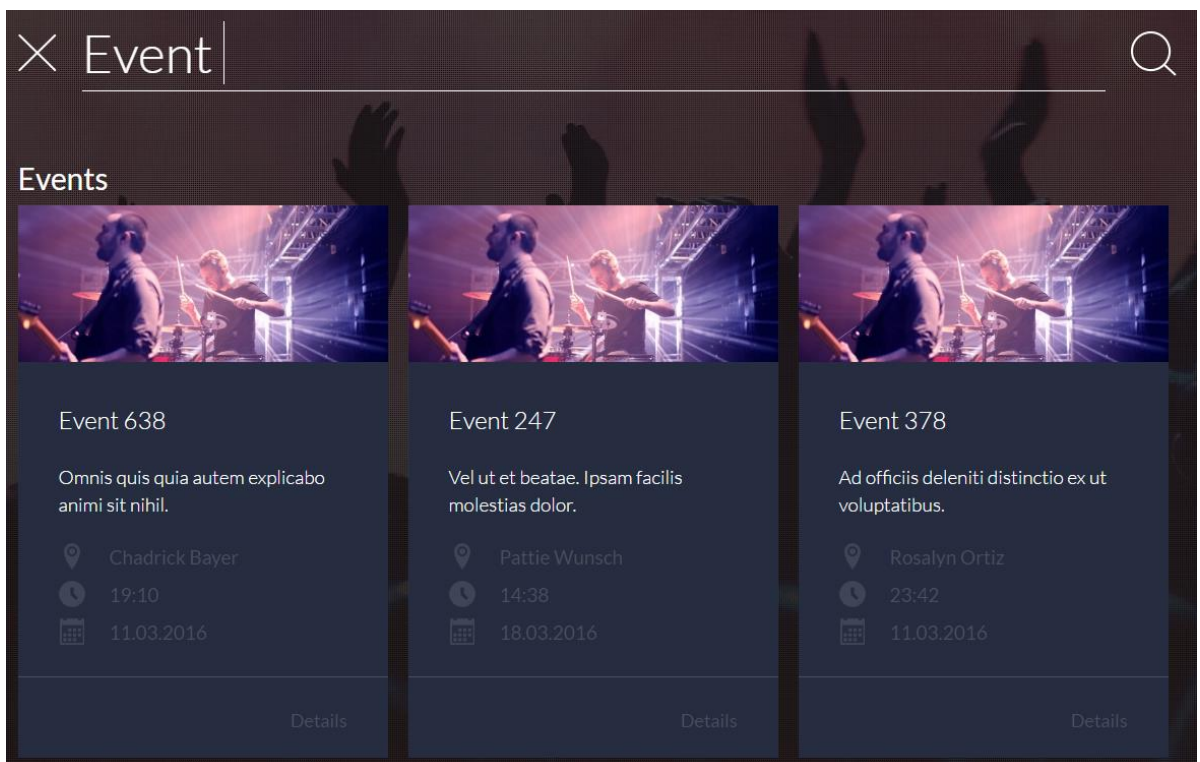


Abb. 42 Schnellsuche

### 7.11.2 Events

Events können mit einer Textsuche, einem Filter nach Tags, einem Kategorie-Filter und einem Datum-Filter gesucht werden.

Folgende Abbildung zeigt das Formular der Eventsuche.

Name	Event eingeben		
Ort	Ort eingeben		
Künstler	Künstler eingeben		
Tags	Tags auswäh		
Kategorie	<input type="radio"/> Charity	<input type="radio"/> Musik	<input type="radio"/> Public
	<input type="radio"/> Special	<input type="radio"/> Sport	
Von			Datum auswählen
Bis			Datum auswählen
<input type="button" value="Suchen"/>			

Abb. 43 Eventsuche

#### 7.11.2.1 Textsuche

Bei der Textsuche gibt es Eingabefelder für Name und Beschreibung, nach denen die Events gesucht werden können. Es werden dann nur noch Events, die auf die Eingabe zutreffen, angezeigt.

#### 7.11.2.2 Tag-Filter

Aus einer *Select2*-Liste können einen oder mehrere Tags ausgewählt werden. Events können beim Erstellen mit Tags getaggt werden. Durch diesen Filter werden nur noch Events angezeigt, die die ausgewählten Tags haben.

#### 7.11.2.3 Kategorie-Filter

Mit den Sidebar Buttons können Events nach Kategorien gefiltert werden. Es gibt sechs Kategorien: Aktuell (Standardauswahl), Musik, Charity, Sport, Public und Special. Durch wählen einer Kategorie werden auf der Startseite nur noch Events dieser Kategorie gelistet. Weiters kann die Kategorie auch bei der Eventsuche angegeben werden. Die Kategorie kann beim Erstellen eines Events ausgewählt werden.

#### 7.11.2.4 Datum-Filter

Events können nach einem einzelnen Datum oder einem Datumsbereich gefiltert werden. Zum Filtern nach einem einzelnen Datum muss entweder nur das Von- oder nur das Bis-Datum angegeben werden. Um nach Events in einem Bereich zu filtern, müssen beide Daten angegeben werden.

### 7.11.3 Künstler

Künstler können über eine Textsuche, einen Genre-Filter und einen ABC-Filter gesucht werden.

#### 7.11.3.1 Textsuche

Mit der Textsuche können Künstler nach Name und Beschreibung gesucht werden. Es werden dann nur noch Künstler, die auf die Eingabe zutreffen, angezeigt.

#### 7.11.3.2 Genre-Filter

Künstler können nach ihrem jeweiligen Genre gefiltert werden. Auf der Künstlerseite werden alle Genres in einer *Select2*-Liste aufgelistet, von der dann ein oder mehrere Genres ausgewählt werden können. Darauf werden die Künstler nach den ausgewählten Genres gefiltert.

#### 7.11.3.3 ABC-Filter

Um Künstler schnell zu filtern, können sie nach ihrem Anfangsbuchstaben gefiltert werden. Im Headerbereich der Künstlerseite befindet sich eine horizontale Liste mit allen Buchstaben des Alphabets. Klickt man auf einen Buchstaben, werden nur noch Künstler, deren Name mit diesem Buchstaben beginnt, gelistet.

In folgender Abbildung ist der ABC-Filter dargestellt.

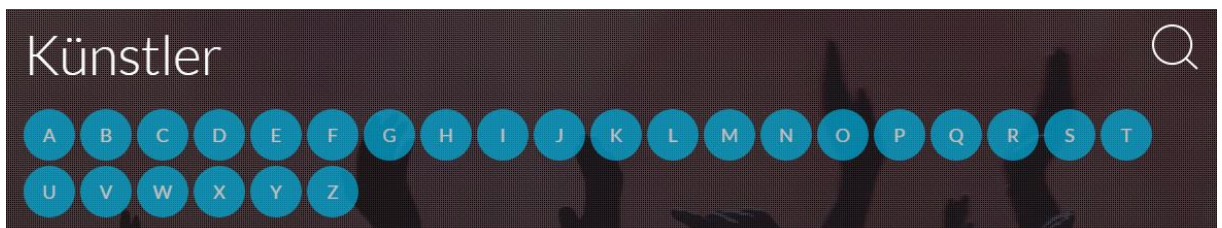


Abb. 44 Künstler ABC-Filter



### 7.11.4 Locations

Locations können mittels einer Textsuche und einem Umkreisfilter gesucht werden.

#### 7.11.4.1 Textsuche

Mit der Textsuche können Locations nach Name und Beschreibung gesucht werden. Es werden dann nur noch Locations, die auf die Eingabe zutreffen, angezeigt.

#### 7.11.4.2 Umkreisfilter

Locations im Umkreis des aktuellen Standorts des Benutzers können gefiltert werden. Der Benutzer kann dabei den Radius des Umkreises selbst festlegen. So wird auf der Karte der gewählte Umkreis, in Form eines Kreises angezeigt. Anschließend werden nur noch Locations innerhalb des Umkreises gezeigt.

Folgende Abbildung stellt den Umkreisfilter dar. Bei diesem Filter wird unten ein Umkreisradius von 10.000 Metern eingegeben. Darauf wird mit diesem Radius der Umkreis von der Position des Benutzers in der Karte eingezeichnet. Eventlocations im Umkreis werden mit Markern dargestellt.

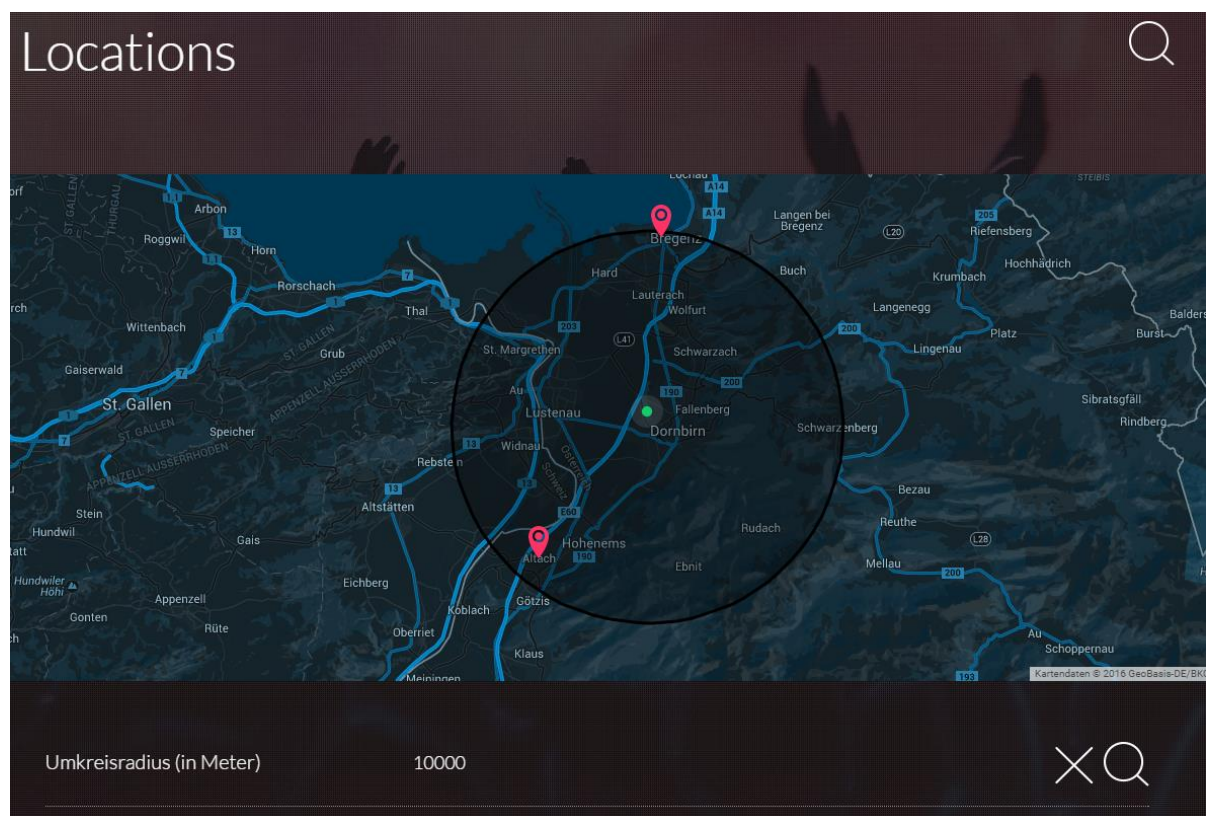


Abb. 45 Locations Umkreisfilter

## 7.12 Newsletter

Jeder angemeldete Benutzer hat die Möglichkeit, in den Einstellungen diverse Newsletter zu abonnieren. Dadurch erhält dieser zum Beispiel bei einer Änderung eines Events, welches dieser als Favorit gespeichert hat, eine Benachrichtigung per Email. Ein weiterer Newsletter schickt zweimal pro Woche eine Liste mit Events, die in dieser Woche stattfinden. Administratorinnen und Moderatoren erhalten einmal pro Tag eine E-Mail, in der alle neuen Anfragen aufgelistet sind.

Die Newsletter werden über den *MailChimp* Service versandt. *MailChimp* ist ein Email-Marketing-Service (EMS), welcher es ermöglicht, Newsletter an eine bestimmte Gruppe (Liste) zu schicken. Außerdem ermöglicht *MailChimp* die Erstellung von Vorlagen und eine Analyse der verschickten Emails.

Mithilfe dessen API kann die von *MailChimp* generierte ID der gewünschten Liste (Gruppe abonnierten Benutzer) im Code angegeben werden. Administratorinnen können außerdem neue Newsletter in der Datenbank abspeichern. Dazu muss ein Name, der Betreff der Email und die ID von *MailChimp* eingetragen werden. Jeder Benutzer hat dann die Möglichkeit, in den Einstellungen die eingetragenen Newsletter zu abonnieren bzw. abzustellen.



## 7.13 Asynchrone Datenübertragung mit AJAX

Für das Bewerten, Teilnehmen und Favorisieren von Events, Künstlern und Locations sowie der schnellen Suche, wird die Benutzerfreundlichkeit durch AJAX-Calls (Asynchronous JavaScript and XML) erhöht. Vorteil von AJAX ist, dass die Seite nicht neugeladen werden muss. Mithilfe von JS wird im Hintergrund der Request an den Server übermittelt. Der Server antwortet darauf und je nachdem, ob der AJAX-Call erfolgreich war, wird die Benutzeroberfläche aktualisiert.

Anhand folgender Abbildung (Abb. 46) kann ein AJAX-Call genauer erklärt werden. Bei dem Codeausschnitt handelt es sich um das Bewerten eines Events, Künstlers oder einer Location.

```
function postRating(rating) {
    $.ajax({
        url: '{{ $rate }}',
        type: "POST",
        data: {
            'id': '{{ $id }}',
            'rating': rating[1]
        },
        success: function (data) {
            $('#rating-').detach();

            for (var i = 1; i <= data[0]; i++) {
                $('#rating').append('<i id="rating-'
                    + i + ' class="fa fa-star"></i> ');
            }

            for (i = 1; i <= (5 - data[0]); i++) {
                $('#rating').append('<i id="rating-'
                    + (i + parseInt(data[0])) + ' class="fa fa-star-o"></i> ');
            }

            $('#rating').append('<span id="rating-total"><b> ' + data[2] + '</b></span>');
            $('#rating').append('<span id="rating-count"> (' + data[1] + ')</span>');
        }, error: function () {
            alert('{{ $name }} konnte nicht bewertet werden!');
        }
    });
}
```

Abb. 46 AJAX Codeauszug

Der Funktion “postRating” wird die Anzahl der Bewertung (1-5) übergeben. Diese Menge wird darauf im “data” Tag dem Attribut “rating” zugewiesen. Neben dem Attribut “rating” wird auch die “ID” des gewünschten Datensatz übermittelt. Diese wird benötigt, um die Bewertung dem richtigen Datensatz zuzuordnen.

Über dem “data” Tag muss noch der Typ des *AJAX*-Calls und dessen URL festgelegt werden. Die URL „{{ \$rate }}“ wird über die jeweilige *Laravel* “view” übergeben, da diese je nach Typ (Event, Künstler und Location) unterschiedlich ist.

Die *AJAX*-Calls basieren auf *REST*. In *REST* gibt es neben dem Typ *POST* drei weitere Befehle. Diese sind in folgender Tabelle aufgelistet.

Befehl	Beschreibung
GET	Eine Ressource vom Server erhalten
POST	Eine Ressource dem Server übergeben und erstellen
PUT	Eine vorhandene Ressource bearbeiten bzw. erstellen, falls diese noch nicht vorhanden ist
DELETE	Eine Ressource löschen

*Tab. 11 REST Befehle*

Im Codeausschnitt (Abb. 46) wird der Typ *POST* verwendet, da eine neue Bewertung erstellt wird.

Zuletzt wird noch, je nach erfolgreicher oder fehlerhafter Ausführung eine Funktion durchgeführt. Bei der „success“ Funktion wird die Anzahl der Bewertungen erhöht und der Durchschnitt aller Bewertungen aktualisiert. Hingegen wird bei der „error“ Funktion ein Alert-Fenster geöffnet und dem Benutzer klargemacht, dass ein Fehler aufgetreten ist.

## 8 Testing

Getestet wurde mit zwei Methoden. Einerseits mit automatisierten Tests (*White-Box* und *Black-Box-Tests*) und andererseits wurden manuelle Tests, sogenannte *User-Acceptance-Tests* (*UATs*), durchgeführt.

### 8.1 Automatisierte Tests

Die automatisierten Tests wurden mithilfe von *PHPUnit* ausgeführt. *White-Box-Tests* wurden verwendet, um einzelne Einheiten (Units) des Programm-Codes auf Funktionsfähigkeit zu testen. Durch *Laravel* sind auch *Black-Box-Tests* mit *PHPUnit* möglich.

### 8.1.1 White-Box-Tests

Bei den *White-Box-Tests* wird im Hintergrund getestet und nicht auf Basis der Benutzeroberfläche. In folgendem Codeausschnitt (Abb. 47) wird das Erstellen einer Location getestet.

```
class LocationTest extends TestCase
{
    use DatabaseTransactions;

    public function testArtistCreation()
    {
        $location = factory(App\Location::class, 1)->create();
        $this->seeInDatabase('locations', $location->toArray());
    }
}
```

Abb. 47 Location Test

Mithilfe einer *Factory*, welche wie bei den *Seedern* verwendet wird, können Datensätze mit generierten Inhalten erstellt werden. Bei diesem Beispiel wird nur ein Datensatz benötigt, was beim zweiten Parameter der *Factory*-Funktion festgelegt ist.

Zuletzt wird noch überprüft, ob der soeben erstellte Datensatz in der Datenbank vorhanden ist. Falls ja, ist der Test erfolgreich abgeschlossen.

Damit der erstellte Datensatz nicht gespeichert wird, wird das von *Laravel* bereitgestellte *Trait* „DatabaseTransactions“ verwendet. Durch dieses *Trait* wird beim Beginn des Tests eine Transaktion gestartet. Beim eigentlichen Testablauf werden dann die Daten in die Datenbank gespeichert. Beim Ende des Tests wird durch das verwendete *Trait* die Transaktion zurückgerollt, d.h., dass die erstellten Daten bzw. die Änderungen wieder gelöscht werden.

### 8.1.2 Black-Box-Tests

Im Gegensatz zu den *White-Box-Tests* ist bei den *Black-Box-Tests* der Source-Code nicht bekannt. Dadurch werden diese Tests auf Basis der Benutzeroberfläche durchgeführt. Ein solcher automatisierter *Black-Box-Test* wird mit folgendem Codeausschnitt (Abb. 48) erklärt. Der Ausschnitt ist ein Test für das Erstellen eines neuen Künstlers.

```
class ArtistTest extends TestCase
{
    use DatabaseTransactions;

    protected $user;

    public function setUp()
    {
        parent::setUp();
        $this->user = User::whereName('admin')->first();
    }

    public function testArtistCreation()
    {
        $this->actingAs($this->user)
            ->visit('artists/create')
            ->see('Neuen Künstler erstellen')
            ->type('Test Künstler', 'name')
            ->type('Lange Test Beschreibung', 'description')
            ->type('Kurze Test Beschreibung', 'description_short')
            ->press('Künstler hinzufügen')
            ->see('Der Künstler wurde erstellt!');
    }
}
```

Abb. 48 Artist Test

In dieser Testklasse gibt es zwei Funktionen. Bei der „setUp“ Funktion wird ein Benutzer mit der Berechtigung zum Erstellen eines neuen Künstlers aus der Datenbank ausgewählt und in die Variable „\$user“ gespeichert.

In „testArtistCreation“ läuft der eigentliche Test ab. Zuerst wird der Applikation übermittelt, dass der Test, mit dem zuvor festgelegten Benutzer, ausgeführt werden soll. Dieser navigiert dann auf die Seite, auf der die Form zum Erstellen eines Künstlers liegt. Darauf werden die Eingabefelder ausgefüllt und der „Künstler hinzufügen!“ Button bestätigt. Wenn dann die Meldung „Der Künstler wurde erstellt!“ ersichtlich ist, ist der Test erfolgreich abgeschlossen.

Wie beim *White-Box-Test* Beispiel wird „DatabaseTransactions“ verwendet, damit der erstellte Datensatz wieder gelöscht wird.

## 8.2 Manuelle Tests

Im Unterschied zu den automatischen Tests wurden die manuellen Tests von Personen durchgeführt.

### 8.2.1 UATs

*UATs* wurden auf mehreren Browsern (Chrome, Firefox und Safari) sowie auf mehreren Geräten (Desktop-PC, Laptop und Smartphones) und Betriebssystemen durchgeführt.

Bei dieser Art von Tests war der Hauptfokus auf den Hauptfunktionen und der Darstellung der Seite. Die Ergebnisse waren meist von Gerät zu Gerät unterschiedlich.

Das Feedback von der Designerin (Frau Zabrodina) war für das Projektteam sehr wichtig, da nicht alle Fehler bei der Darstellung sofort aufgefallen sind.

Durch die *UATs* wurden viele Darstellungsfehler identifiziert und ausgebessert. Das Feedback zur Darstellung in verschiedenen Browsern und Geräten war für das Projektteam sehr wichtig, da die Möglichkeit zum Testen auf Safari ansonsten nicht möglich gewesen wäre.

## 9 Fazit

Die Aufgabenstellung dieser Diplomarbeit war die Entwicklung eines webbasierten Eventplaners, mit dem sich Benutzer über aktuelle Events informieren sowie eigene Events erstellen können. Die Umsetzung war anfangs mit dem CMS WordPress geplant, nach einer Evaluierungsphase wurde jedoch auf eine Entwicklung mit dem *PHP Framework Laravel* umgestiegen.


Das Projekt verlief für das Projektteam sehr gut. Alle Teammitglieder konnten ihre Fähigkeiten im Bereich der Teamarbeit sowie der Webprogrammierung stark erweitern und dabei einige Erfahrung sammeln.


Erkenntnisse aus dem Projekt waren für das Team, dass es bei Softwareprojekten oft sehr schwierig ist einzuschätzen wie lange man für die Entwicklung eines einzelnen Bestandteils der Software benötigt. Zusätzlich wurde erkannt, dass die Entwicklung mit *Frameworks* wie *Laravel* und *Sass* deutlich bequemer und schneller erfolgt.


Aus dem Projekt entstand ein funktionsfähiger Eventplaner mit modernem Design. Es ist geplant, dass ToWa den Source Code des Eventplaners übernimmt, damit ihn die Firma bei zukünftigen, ähnlichen Projekten als Basis oder Referenz verwenden kann.




## 10 Hilfsmittel


<p>Bitbucket</p> <p><a href="https://bitbucket.org/">https://bitbucket.org/</a></p>	 <p>Abb. 49 Bitbucket – Logo (Atlassian, 2016)</p>
<p>Bitbucket ist ein Online-Dienst, der eine Versionverwaltung mit Git unterstützt.</p>	


<p>Bootstrap</p> <p><a href="http://getbootstrap.com/">http://getbootstrap.com/</a></p>	 <p>Abb. 50 Bootstrap – Logo (Bootstrap, 2016)</p>
<p>Bootstrap ist ein freies <i>CSS-Framework</i>, das Designvorlagen für Grids, Buttons, Tabellen, Formulare und andere Elemente bietet. Bootstrap wurde als Basis für das Design verwendet.</p>	

<p>Composer</p> <p><a href="https://getcomposer.org/">https://getcomposer.org/</a></p>	 <p>Abb. 51 Composer – Logo (WizardCat, 2016)</p>
<p>Composer ist ein Dependency Manager für <i>PHP</i>. Mit ihm lassen sich angegebene Dependencies (z.B. Libraries) per Kommandozeilenbefehl automatisch installieren bzw. updaten.</p>	


<p><b>Cropper</b></p> <p><a href="https://fengyuanchen.github.io/cropper/">https://fengyuanchen.github.io/cropper/</a></p>	
<p>Cropper ist ein jQuery Plugin, dass eine Bildzuschneide-Funktion implementiert. Es wurde für das Zuschneiden von Titelbildern verwendet.</p>	


<p><b>CSS / Sass</b></p> <p><a href="http://sass-lang.com/">http://sass-lang.com/</a></p>	 <p><i>Abb. 52 Sass – Logo (Sass, 2016)</i></p>
<p>CSS ist eine Sprache, mit der beschrieben wird, wie HTML Dokumente im Webbrowser dargestellt werden sollen. Sass ist ein CSS Preprocessor, der bei der Entwicklung von Webdesign hilft. Sass-Code ist strukturierter und übersichtlicher als CSS-Code. Der Sass-Code wird dann anschließend zu normalem CSS kompiliert.</p>	


<p><b>Disqus</b></p> <p><a href="https://disqus.com/">https://disqus.com/</a></p>	 <p><i>Abb. 53 Disqus – Logo (Disqus, 2016)</i></p>
<p>Disqus ist ein freier Kommentar Service. Er wird für die Implementierung von Event-Kommentaren verwendet.</p>	


<p><b>Git</b></p> <p><a href="https://git-scm.com/">https://git-scm.com/</a></p>	 <p><i>Abb. 54 Git – Logo (Jason Long, 2016)</i></p>
<p>Git ist eine Software für die Versionsverwaltung. Git wurde mit Bitbucket verwendet, um den Source Code online zu sichern und alle Änderungen festzuhalten.</p>	


<p style="text-align: center;"><b>Google Analytics</b></p> <p style="text-align: center;"><a href="https://analytics.google.com/">https://analytics.google.com/</a></p>	<div data-bbox="1085 203 1254 367" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 55 Google Analytics – Logo (Google, Google</i></p>
<p>Durch Google Analytics konnte der Traffic auf dem Testserver kontrolliert werden. Es wird visualisiert, wer wann auf welcher Seite mit welchem Gerät gewesen ist.</p>	
<p style="text-align: center;"><b>Google Docs</b></p> <p style="text-align: center;"><a href="https://docs.google.com/">https://docs.google.com/</a></p>	<div data-bbox="1121 645 1224 772" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 56 Google Docs – Logo (Google, Google Apps Setup, 2016)</i></p>
<p>Google Docs ist ein Textverarbeitungsprogramm, welches ähnliche Funktionen wie Microsoft Word besitzt, jedoch hat es den Vorteil, dass mehrere Personen gleichzeitig an einem Dokument arbeiten können.</p>	
<p style="text-align: center;"><b>Google Drive</b></p> <p style="text-align: center;"><a href="https://drive.google.com/">https://drive.google.com/</a></p>	<div data-bbox="1091 1133 1251 1270" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 57 Google Drive – Logo (Google, Google Apps Setup, 2016)</i></p>
<p>Google Drive wurde für den Austausch aller projektrelevanten Dokumente verwendet.</p>	
<p style="text-align: center;"><b>Google Sheets</b></p> <p style="text-align: center;"><a href="https://sheets.google.com/">https://sheets.google.com/</a></p>	<div data-bbox="1118 1574 1225 1704" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 58 Google Sheets – Logo (Google, Google Apps Setup, 2016)</i></p>
<p>Google Sheets ist ein Tabellenkalkulationsprogramm. Die ToDo-Liste und die Zeitaufzeichnung wurden mit Google Sheets erstellt.</p>	

<p>Google Slides</p> <p><a href="https://slides.google.com/">https://slides.google.com/</a></p>	 <p><i>Abb. 59 Google Slides – Logo (Google, Google Apps Setup, 2016)</i></p>
<p>Mit Google Slides wurden die Präsentationen und Statusberichte erstellt, wie bei Google Docs ist es möglich, gleichzeitig an einem Dokument zu arbeiten.</p>	

<p>HTML 5</p> <p><a href="https://www.w3.org/html/">https://www.w3.org/html/</a></p>	 <p><i>Abb. 60 HTML 5 – Logo (W3, 2016)</i></p>
<p><i>HTML</i> ist die grundlegende Sprache zur Entwicklung von Webseiten. <i>HTML</i> wurde für die Struktur der einzelnen Webseiten verwendet.</p>	


<p>JavaScript</p> <p><a href="https://www.javascript.com/">https://www.javascript.com/</a></p>	 <p><i>Abb. 61 JavaScript – Logo (voodootikigod, 2016)</i></p>
<p>JavaScript ist eine Skriptsprache, die für dynamische Webseiten verwendet wird. JavaScript wird beispielsweise verwendet, um neue Seiteninhalte zu laden ohne die gesamte Seite neuzuladen.</p>	


<p>jQuery</p> <p><a href="https://jquery.com/">https://jquery.com/</a></p>	 <p><i>Abb. 62 jQuery – Logo (jQuery, 2016)</i></p>
<p>jQuery ist eine JavaScript Library, die die Programmierung von bestimmten JavaScript-Funktionen deutlich erleichtert und beschleunigt.</p>	



<p>Laravel</p> <p><a href="https://laravel.com/">https://laravel.com/</a></p>	 <p>Abb. 63 Laravel – Logo (Laravel, 2016)</p>
<p><i>Laravel</i> ist ein <i>PHP Web-Framework</i>, das für die gesamte Entwicklung verwendet wurde. Ein <i>Web-Framework</i> bietet eine Vielzahl von vordefinierten Klassen und Funktionen, um die Entwicklung zu vereinfachen und zu beschleunigen.</p>	


<p>MailChimp</p> <p><a href="http://mailchimp.com/">http://mailchimp.com/</a></p>	 <p>Abb. 64 MailChimp – Logo (MailChimp, 2016)</p>
<p><i>Mailchimp</i> ist ein Email Marketing Dienst mit dem E-Mails automatisch anhand einer Vorlage zu einer Gruppe von Empfängern gesendet werden. Der Dienst wird für das Versenden von Newslettern verwendet.</p>	


<p>Masonry</p> <p><a href="http://masonry.desandro.com/">http://masonry.desandro.com/</a></p>	
<p>Masonry ist eine JavaScript Library, die Elemente in einem Grid-Layout optimal positioniert. Masonry wird für die Darstellung von den Karten verwendet.</p>	

<p>Microsoft Project 2016</p> <p><a href="https://products.office.com/project/">https://products.office.com/project/</a></p>	 <p>Abb. 65 Microsoft Project – Logo (Pepsi9072, 2016)</p>
<p>MS Project ist ein Projektmanagement Programm, das für die Erstellung des Projektterminplans verwendet wurde.</p>	


<p>MySQL</p> <p><a href="https://www.mysql.de/">https://www.mysql.de/</a></p>	 <p>Abb. 66 MySQL – Logo (MySQL, 2016)</p>
<p>MySQL ist ein relationales Datenbanksystem, das als Datenbank verwendet wird.</p>	

<p>Node.js / NPM</p> <p><a href="https://nodejs.org/">https://nodejs.org/</a>  <a href="https://www.npmjs.com/">https://www.npmjs.com/</a></p>	 <p>Abb. 68 Node.js – Logo (Node.js, 2016)</p>  <p>Abb. 67 NPM – Logo (npm, 2016)</p>
<p>Node.js ist eine Laufzeitumgebung für die Entwicklung von Web-Applikationen. Node ist aus mehreren Modulen aufgebaut. Zur Verwaltung von Modulen gibt es den Node Packet Manager (NPM).</p>	


<p>PHP</p> <p><a href="http://php.net/">http://php.net/</a></p>	 <p>Abb. 69 PHP – Logo (PHP, PHP: Download Logos, 2016)</p>
<p>PHP ist eine Sprache zum Entwickeln von Webanwendungen bzw. dynamischen Webseiten.</p>	


<p>PHPStorm</p> <p><a href="https://www.jetbrains.com/phpstorm/">https://www.jetbrains.com/phpstorm/</a></p>	 <p>Abb. 70 PhpStorm – Logo (JetBrains, 2016)</p>
<p>PHPStorm ist eine integrierte Entwicklungsumgebung (IDE) für PHP und wurde zum Schreiben des Programmcodes verwendet.</p>	

<p style="text-align: center;"><b>SourceTree</b></p> <p style="text-align: center;"><a href="https://www.sourcetreeapp.com/">https://www.sourcetreeapp.com/</a></p>	<div data-bbox="1098 208 1249 353" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 71 SourceTree – Logo (Atlassian, 2016)</i></p>
<p>SourceTree ist eine grafische Benutzeroberfläche für die Versionsverwaltung des Source Codes, die die Verwaltung von Code übersichtlicher und benutzerfreundlicher macht.</p>	
<p style="text-align: center;"><b>Trello</b></p> <p style="text-align: center;"><a href="https://trello.com/">https://trello.com/</a></p>	<div data-bbox="1091 696 1257 857" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 72 Trello – Logo (Trello, 2016)</i></p>
<p>Trello ist ein webbasiertes Kanban Programm. Auf einem <i>Board</i> werden offene <i>Tasks</i> erstellt, die abgearbeitet werden müssen. Trello wurde als ToDo Liste verwendet.</p>	
<p style="text-align: center;"><b>Vagrant / Homestead</b></p> <p style="text-align: center;"><a href="https://www.vagrantup.com/">https://www.vagrantup.com/</a>  <a href="https://github.com/laravel/homestead/">https://github.com/laravel/homestead/</a></p>	<div data-bbox="1098 1191 1249 1339" data-label="Image"> </div> <p style="text-align: center;"><i>Abb. 73 Vagrant – Logo (Fco.plj, 2016)</i></p>
<p>Anstatt bei der Programmierung auf dem lokalen System zu programmieren, wurde eine Vagrantbox von <i>Laravel</i> namens <i>Homestead</i> verwendet. In <i>Homestead</i> sind alle Einstellungen (Webserver, MySQL, etc.) bereits an <i>Laravel</i> angepasst, was ein großer Vorteil ist.</p>	

<p>VirtualBox</p> <p><a href="https://www.virtualbox.org/">https://www.virtualbox.org/</a></p>	 <p><i>Abb. 74 VirtualBox – Logo (Oracle Corporation, 2016)</i></p>
<p>VirtualBox ist eine Virtualisierungssoftware, in der <i>Homestead</i> ausgeführt wird.</p>	

<p>Visual Paradigm</p> <p><a href="https://www.visual-paradigm.com/">https://www.visual-paradigm.com/</a></p>	 <p><i>Abb. 75 Visual Paradigm – Logo (Visual Paradigm, 2016)</i></p>
<p>Visual Paradigm ist ein Programm zur Erstellung von <i>UML</i> Diagrammen.</p>	

<p>WBStool 8</p> <p><a href="http://www.wbstool8.com/">http://www.wbstool8.com/</a></p>	 <p><i>Abb. 76 WBStool – Logo (Schoder, 2016)</i></p>
<p>WBStool ist ein Projektmanagement Programm, das zur Erstellung des Projektstrukturplans verwendet wurde.</p>	

<p>XMind</p> <p><a href="http://www.xmind.net/">http://www.xmind.net/</a></p>	 <p><i>Abb. 77 XMind – Logo (XMind, 2016)</i></p>
<p>XMind ist ein Programm zur Erstellung von Mind-Maps. Es wurde für die Erstellung des Big Pictures verwendet.</p>	



# 11 Glossar

Begriff	Beschreibung
<b>AJAX</b>	Konzept der asynchronen Datenübertragung zwischen einem Browser und dem Server. <sup>2</sup>
<b>Artisan</b>	Kommandozeilen Interface von <i>Laravel</i> .
<b>Blade</b>	Templating Engine, die den Umgang mit <i>HTML/PHP</i> erleichtert.
<b>Black-Box-Test</b>	Automatisierter Test, bei dem die innere Funktionsweise der Software nicht bekannt ist.
<b>Branch</b>	Teilung des Source Codes in mehrere Entwicklungszweige, auf denen dann separat weiterentwickelt werden kann.
<b>CSS</b>	Dient für die Gestaltung von Webseiten.
<b>Google Fonts</b>	Erleichtert das Einbinden von diversen Schriftarten in eine Webseite. Der Service läuft auf einem Google Server und ist deshalb sehr schnell und zuverlässig. Alle verfügbaren Schriftarten sind Open Source.
<b>Framework</b>	“Ein <i>Framework</i> ist selbst noch kein fertiges Programm, sondern stellt den Rahmen zur Verfügung, innerhalb dessen der Programmierer eine Anwendung erstellt, wobei u. a. durch die in dem <i>Framework</i> verwendeten Entwurfsmuster auch die Struktur der individuellen Anwendung beeinflusst wird.” <sup>3</sup>
<b>Homestead</b>	Lokale Entwicklungsumgebung, die von <i>Laravel</i> bereitgestellt wird.

<sup>2</sup> Vgl. (Wikipedia, Ajax (Programmierung), 2016)

<sup>3</sup> Zitat (Wikipedia, Framework, 2016)

<b>HTML</b>	Dient zur Strukturierung von Inhalten wie Texten, Bildern und Hyperlinks in Dokumenten. <sup>4</sup>
<b>Laravel</b>	<i>PHP-Framework</i> , das die Entwicklung mit <i>PHP</i> erleichtert.
<b>MailChimp</b>	E-Mail-Marketing Dienst mit dem E-Mails automatisch anhand einer Vorlage zu einer Gruppe von Empfängern gesendet werden.
<b>Migration</b>	Datenbank <i>Migrations</i> sind eine Versionskontrolle für Datenbanken. <i>Migrations</i> sind spezielle <i>PHP</i> Dateien in denen Code geschrieben wird, der bei einer Ausführung eine Datenbank bearbeitet, d.h. z.B. eine neue Tabelle erstellt oder ein Attribut einer vorhandenen Tabelle ändert.
<b>Merge</b>	Zusammenfügen von zwei oder mehreren <i>Branches</i> zu einem einzelnen Entwicklungsweig.
<b>Mockup</b>	In der Softwareentwicklung bezeichnet <i>Mockup</i> einen Prototyp der Benutzerschnittstelle einer zu erstellenden Software.
<b>Modal</b>	Ein <i>Modal</i> ist ein Dialogfenster, welches den Vorteil hat, dass die Seite nicht neu geladen werden muss. <i>Modale</i> werden verwendet, um die Benutzerfreundlichkeit zu erhöhen.
<b>Nginx</b>	Spricht „ <i>Engine X</i> “, ist ein Webserver.
<b>OAuth / OAuth Token</b>	Mithilfe des <i>OAuth</i> Protokolls kann auf die Daten eines Benutzers über eine andere Anwendung (Service) zugegriffen werden, ohne alle Details (speziell das Passwort) dessen Zugangsberechtigung preiszugeben. Anstelle der üblichen Benutzername und Passwort Kombination wird ein sogenannter Token generiert, dieser unterscheidet sich in einen Abfrage- und Zugangstoken. <sup>5</sup>

<sup>4</sup> Vgl. (Wikipedia, Hypertext Markup Language, 2016)

<sup>5</sup> Vgl. (Wikipedia, OAuth, 2016)

<b>PHP</b>	“ <i>PHP</i> ist eine Skriptsprache mit einer an C und Perl angelehnten Syntax, die hauptsächlich zur Erstellung dynamischer Webseiten oder Webanwendungen verwendet wird.” <sup>6</sup>
<b>PHP-GD</b>	Ermöglicht die Manipulation von Bildern.
<b>PHPStorm</b>	Entwicklungsumgebung von Jet Brains für die Entwicklung mit <i>PHP</i> .
<b>PHPUnit Test</b>	Automatisierter <i>White-Box-Test</i> , der eine einzelne Einheit (Unit) des Programmcodes auf richtige Funktionalität testet. Z.B. kann ein Test geschrieben werden, mit dem automatisch ein Event erstellt wird. Der Test gibt nach der Ausführung dann aus, ob die Erstellung erfolgreich war oder nicht.
<b>Responsive</b>	“Gestalterisches und technisches Paradigma zur Erstellung von Webseiten, so dass diese auf Eigenschaften des jeweils benutzten Endgeräts, vor allem Smartphones und Tabletcomputer, reagieren können.” <sup>7</sup>
<b>REST</b>	Architekturstil für die Kommunikation mit Web Services.
<b>Sass</b>	“Sass ist eine Stylesheet-Sprache, die als Präprozessor die Erzeugung von Cascading Style Sheets erleichtert.” <sup>8</sup>
<b>Seeder</b>	Methode zum Füllen einer Datenbank mit Testdaten.
<b>Select2</b>	Stark anpassbare jQuery Select Box, die mitunter die Implementierung von Tags unterstützt.

---

<sup>6</sup> Zitat (Wikipedia, PHP, 2016)

<sup>7</sup> Zitat (Wikipedia, Responsive Webdesign, 2016)

<sup>8</sup> Zitat (Wikipedia, Sass (Stylesheet-Sprache), 2016)

<b>Slug</b>	<p><i>Slug</i> ist eine Bezeichnung für den Teil einer URL, der für Menschen lesbar ist. Es ist normalerweise der letzte Abschnitt der URL und gibt z.B. den Namen einer Ressource klar verständlich an.</p> <p>Beispiel: „http://.../events/2016/03/04/test“</p> <p>Obige URL führt zu einem Event namens Test, das am 04.03.2016 stattfindet, was auch in der URL ersichtlich ist.</p>
<b>Socialite</b>	<p><i>Socialite</i> von <i>Laravel</i> dient als Schnittstelle zwischen der Applikation und der <i>OAuth</i>-Authentifizierung mit Facebook, Google+ und Twitter. Es verarbeitet die Authentifizierungscodes, was die Programmierung stark erleichtert.</p>
<b>Trait</b>	<p>Eine Methode zur Wiederverwendung von Code.</p> <p>“Ein <i>Trait</i> kann verwendet werden, um den Beschränkungen der einfachen Vererbung auszuweichen, indem er erlaubt, dass Mengen von Methoden frei in mehreren unabhängigen Klassen, die in verschiedenen Klassenhierarchien stecken, wiederzuverwenden.”<sup>9</sup></p>
<b>UAT</b>	<p>Von einem Benutzer manuell ausgeführter Test einer Software auf richtige Funktionalität oder Benutzerfreundlichkeit.</p>
<b>UML</b>	<p>“Grafische Modellierungssprache zur Spezifikation, Konstruktion und Dokumentation von Software-Teilen und anderen Systemen.”<sup>10</sup></p>
<b>White-Box-Test</b>	<p>Automatisierter Test, bei dem die innere Funktionalität der Software bekannt ist.</p>

Tab. 12 Glossar

<sup>9</sup> Zitat (PHP, Traits, 2016)

<sup>10</sup> Zitat (Wikipedia, Unified Modeling Language, 2016)

## 12 Abbildungsverzeichnis

Abb. 1 Simon Ellensohn .....	1
Abb. 2 Martin Leimser .....	1
Abb. 3 Julian Martin .....	2
Abb. 4 Diethard Kaufmann .....	2
Abb. 5 Benjamin Meier .....	3
Abb. 6 Elena Zabrodina .....	3
Abb. 7 ToWa – Logo (ToWa, 2016) .....	4
Abb. 8 Projektorganigramm .....	10
Abb. 9 Projektstrukturplan .....	11
Abb. 10 Projektterminplan .....	13
Abb. 11 Risikoanalyse .....	15
Abb. 12 Use-Case Diagramm .....	20
Abb. 13 Verteilungsdiagramm - Lokal .....	26
Abb. 14 Verteilungsdiagramm - Testserver .....	27
Abb. 15 Big Picture .....	28
Abb. 16 CSS / SCSS Vergleich .....	29
Abb. 17 Startseite Mockup .....	30
Abb. 18 Eventansicht Mockup .....	31
Abb. 19 Eventansicht Designentwurf .....	32
Abb. 20 Screenshots Anmeldeseite .....	35
Abb. 21 Screenshots Eventseite .....	36
Abb. 22 Screenshots Eventansicht .....	37
Abb. 23 Screenshots Einstellungen .....	38
Abb. 24 Ordnerstruktur .....	39
Abb. 25 App Ordner .....	40
Abb. 26 Vergleich zwischen PHP und Blade Syntax .....	44
Abb. 27 Events-Tabelle .....	46
Abb. 28 Users-Tabelle .....	47
Abb. 29 Artists-Tabelle .....	47
Abb. 30 Locations-Tabelle .....	48
Abb. 31 Anmeldung Sequenzdiagramm .....	50

Abb. 32 Aktivitätsdiagramm - Anmeldung mit E-Mail-Adresse und Passwort .....	52
Abb. 33 Eventantrag erstellen .....	55
Abb. 34 Event erstellen Aktivitätsdiagramm .....	56
Abb. 35 Event erstellen - Formularausschnitt.....	56
Abb. 36 EventRequest Regeln Codeauszug .....	57
Abb. 37 Speicherfunktion Codeauszug .....	57
Abb. 38 Füllbare Felder Codeauszug.....	58
Abb. 39 Bild zuschneiden.....	62
Abb. 40 Zustandsdiagramm - Eventantrag.....	63
Abb. 41 Antrag verwalten .....	64
Abb. 42 Schnellsuche .....	65
Abb. 43 Eventsuche .....	66
Abb. 44 Künstler ABC-Filter .....	68
Abb. 45 Locations Umkreisfilter.....	69
Abb. 46 AJAX Codeauszug.....	71
Abb. 47 Location Test .....	74
Abb. 48 Artist Test.....	75
Abb. 49 Bitbucket – Logo (Atlassian, 2016) .....	79
Abb. 50 Bootstrap – Logo (Bootstrap, 2016) .....	79
Abb. 51 Composer – Logo (WizardCat, 2016) .....	79
Abb. 52 Sass – Logo (Sass, 2016).....	80
Abb. 53 Disqus – Logo (Disqus, 2016).....	80
Abb. 54 Git – Logo (Jason Long, 2016) .....	80
Abb. 55 Google Analytics – Logo (Google, Google Analytics, 2016) .....	81
Abb. 56 Google Docs – Logo (Google, Google Apps Setup, 2016) .....	81
Abb. 57 Google Drive – Logo (Google, Google Apps Setup, 2016) .....	81
Abb. 58 Google Sheets – Logo (Google, Google Apps Setup, 2016) .....	81
Abb. 59 Google Slides – Logo (Google, Google Apps Setup, 2016).....	82
Abb. 60 HTML 5 – Logo (W3, 2016) .....	82
Abb. 61 JavaScript – Logo (voodootikigod, 2016).....	82
Abb. 62 jQuery – Logo (jQuery, 2016) .....	82
Abb. 63 Laravel – Logo (Laravel, 2016) .....	83
Abb. 64 MailChimp – Logo (MailChimp, 2016).....	83

Abb. 65 Microsoft Project – Logo (Pepsi9072, 2016) .....	83
Abb. 66 MySQL – Logo (MySQL, 2016).....	84
Abb. 68 Node.js – Logo (Node.js, 2016) .....	84
Abb. 67 NPM – Logo (npm, 2016).....	84
Abb. 69 PHP – Logo (PHP, PHP: Download Logos, 2016) .....	84
Abb. 70 PHPStorm – Logo (JetBrains, 2016).....	84
Abb. 71 SourceTree – Logo (Atlassian, 2016) .....	85
Abb. 72 Trello – Logo (Trello, 2016).....	85
Abb. 73 Vagrant – Logo (Fco.plj, 2016) .....	85
Abb. 74 VirtualBox – Logo (Oracle Corporation, 2016) .....	86
Abb. 75 Visual Paradigm – Logo (Visual Paradigm, 2016) .....	86
Abb. 76 WBStool – Logo (Schoder, 2016) .....	86
Abb. 77 XMind – Logo (XMind, 2016) .....	86

## 13 Tabellenverzeichnis

Tab. 1 Projektauftrag.....	6
Tab. 2 Projektorganisation .....	9
Tab. 3 Meilensteinplan .....	14
Tab. 4 Risikoanalyse.....	15
Tab. 5 Evaluierung von Eventplanern .....	19
Tab. 6 Vergleich zwischen WordPress und Laravel .....	24
Tab. 7 Designbibliotheken.....	34
Tab. 8 Beschreibung der Unterordner des “database” Ordners .....	41
Tab. 9 Beschreibung Artisan Befehle.....	43
Tab. 10 Benutzerrollen .....	49
Tab. 11 REST Befehle .....	72
Tab. 12 Glossar.....	90
Tab. 13 Abkürzungsverzeichnis .....	95



## 14 Abkürzungsverzeichnis

<b>AJAX</b>	Asynchronous JavaScript and XML
<b>API</b>	Application Programming Interface
<b>CLI</b>	Command Line Interface
<b>CMS</b>	Content Management System
<b>CRUD</b>	Create, Read, Update und Delete
<b>CSS</b>	Cascading Style Sheets
<b>EMS</b>	Email Marketing Service
<b>ER</b>	Entity Relation
<b>HTML</b>	Hypertext Markup Language
<b>JS</b>	JavaScript
<b>PHP</b>	Hypertext Preprocessor
<b>PSP</b>	Projektstrukturplan
<b>PTP</b>	Projektterminplan
<b>PL</b>	Projektleiter
<b>PM</b>	Projektmanagement
<b>REST</b>	Representational State Transfer
<b>Sass</b>	Syntactically Awesome Style Sheets
<b>SCSS</b>	Sassy CSS
<b>SQL</b>	Standard Query Language
<b>UAT</b>	User-Acceptance-Test
<b>UML</b>	Unified Modeling Language
<b>XML</b>	Extended Markup Language

Tab. 13 Abkürzungsverzeichnis

## 15 Literaturverzeichnis

- Atlassian. (2. April 2016). *Atlassian Design Guidelines*. Von <https://design.atlassian.com/brand/logo/> abgerufen
- Bootstrap. (30. März 2016). *Wikimedia Commons*. Von [https://commons.wikimedia.org/wiki/File:Bootstrap\\_logo.svg](https://commons.wikimedia.org/wiki/File:Bootstrap_logo.svg) abgerufen
- Disqus. (30. März 2016). *Brand & Logos | Disqus*. Von <https://disqus.com/brand/> abgerufen
- Fco.plj. (30. März 2016). *Vagrant - Vagrant (Software) - Wikipedia*. Von [https://de.wikipedia.org/wiki/Vagrant\\_\(Software\)](https://de.wikipedia.org/wiki/Vagrant_(Software)) abgerufen
- Google. (30. März 2016). *Google Analytics*. Von <https://analytics.google.com/analytics/web/s/logo-ga.png> abgerufen
- Google. (30. März 2016). *Google Apps Setup*. Von <https://apps.google.com/setup/resources/logos/> abgerufen
- Jason Long. (2. April 2016). *Wikimedia Commons*. Von <https://commons.wikimedia.org/wiki/File:Git-logo.svg> abgerufen
- JetBrains. (30. März 2016). *JetBrains :: Press Room*. Von <https://www.jetbrains.com/company/press/> abgerufen
- jQuery. (30. März 2016). *Logos | jQuery Brand Guidelines*. Von <https://brand.jquery.org/logos/> abgerufen
- Laravel. (30. März 2016). *Laravel*. Von <https://laravel.com/assets/img/laravel-logo.png> abgerufen
- MailChimp. (30. März 2016). *Brand Assets | MailChimp*. Von <http://mailchimp.com/about/brand-assets/> abgerufen
- MySQL. (30. März 2016). *MySQL Logo Downloads*. Von <https://www.mysql.com/about/legal/logos.html> abgerufen
- Node.js. (30. März 2016). *Logos and Graphics | Node.js*. Von <https://nodejs.org/en/about/resources/> abgerufen
- npm. (30. März 2016). *npm/logos: official logos for npm, Inc.* Von <https://github.com/npm/logos> abgerufen
- Oracle Corporation. (30. März 2016). *File:Virtualbox logo.png - Wikimedia Commons*. Von [https://commons.wikimedia.org/wiki/File:Virtualbox\\_logo.png](https://commons.wikimedia.org/wiki/File:Virtualbox_logo.png) abgerufen

- Pepsi9072. (30. März 2016). *Microsoft Project - Logopedia - Wikia*. Von [http://logos.wikia.com/wiki/Microsoft\\_Project](http://logos.wikia.com/wiki/Microsoft_Project) abgerufen
- PHP. (30. März 2016). *PHP: Download Logos*. Von <http://php.net/download-logos.php> abgerufen
- PHP. (31. März 2016). *Traits*. Von <http://php.net/manual/de/language.oop5.traits.php> abgerufen
- Sass. (30. März 2016). *Sass: Brand Guidelines*. Von <http://sass-lang.com/styleguide/brand> abgerufen
- Schoder, D. (30. März 2016). *File:WBStool logo.png - Wikimedia Commons*. Von [https://commons.wikimedia.org/wiki/File:WBStool\\_logo.png](https://commons.wikimedia.org/wiki/File:WBStool_logo.png) abgerufen
- ToWa. (07. April 2016). *Digital Agentur ToWa*. Von <http://www.towa.at/> abgerufen
- Trello. (30. März 2016). *Trello Brand Guide*. Von <https://trello.com/about/branding> abgerufen
- Visual Paradigm. (30. März 2016). *Visual Paradigm Media Kit*. Von <https://www.visual-paradigm.com/aboutus/mediakit.jsp> abgerufen
- voodootikigod. (30. März 2016). *A community logo for JS*. Von <https://github.com/voodootikigod/logo.js> abgerufen
- W3. (30. März 2016). *W3C HTML5 Logo*. Von <https://www.w3.org/html/logo/> abgerufen
- Wikipedia. (28. März 2016). *Ajax (Programmierung)*. Von [https://de.wikipedia.org/wiki/Ajax\\_\(Programmierung\)](https://de.wikipedia.org/wiki/Ajax_(Programmierung)) abgerufen
- Wikipedia. (28. März 2016). *Framework*. Von <https://de.wikipedia.org/wiki/Framework> abgerufen
- Wikipedia. (28. März 2016). *Hypertext Markup Language*. Von [https://de.wikipedia.org/wiki/Hypertext\\_Markup\\_Language](https://de.wikipedia.org/wiki/Hypertext_Markup_Language) abgerufen
- Wikipedia. (1. April 2016). *Laravel*. Von Laravel: <https://en.wikipedia.org/wiki/Laravel> abgerufen
- Wikipedia. (28. März 2016). *OAuth*. Von <https://de.wikipedia.org/wiki/OAuth> abgerufen
- Wikipedia. (27. März 2016). *PHP*. Von <https://de.wikipedia.org/wiki/PHP> abgerufen
- Wikipedia. (31. März 2016). *Responsive Webdesign*. Von [https://de.wikipedia.org/wiki/Responsive\\_Webdesign](https://de.wikipedia.org/wiki/Responsive_Webdesign) abgerufen

- Wikipedia. (28. März 2016). *Sass (Stylesheet-Sprache)*. Von [https://de.wikipedia.org/wiki/Sass\\_\(Stylesheet-Sprache\)](https://de.wikipedia.org/wiki/Sass_(Stylesheet-Sprache)) abgerufen
- Wikipedia. (31. März 2016). *Unified Modeling Language*. Von [https://de.wikipedia.org/wiki/Unified\\_Modeling\\_Language](https://de.wikipedia.org/wiki/Unified_Modeling_Language) abgerufen
- WizardCat. (30. März 2016). *Composer*. Von <https://getcomposer.org/img/logo-composer-transparent3.png> abgerufen
- XMind. (30. März 2016). *XMind*. Von <http://www.xmind.net/blog/de/index.php/xmind-2013-steht-zur-verfugung/> abgerufen