PR Anwendungsentwicklung i.d. Medieninformatik

EventJunkie

Requirements and Design

WS 2014/15 - Gruppe PK2 Team: PK2

Contributors:

MatNr.	Name	e-mail
a1125503	Oliver Guggenberger	a1125503@unet.univie.ac.at
a1207617	Philipp Hierman	a1207617@unet.univie.ac.at
a1151917	Sandra Markhart	a1151917@unet.univie.ac.at

November 9, 2014

Contents

1 Application Overview and Objectives

1.1 Application Overview

EventJunkie ist ein Eventkalender welcher mit Facebook (www.facebook.com), Goabase (www.goabase.de), Flickr (www.flickr.com) und Twitter (www.twitter.com) verknüppft ist. Die Hauptseite enthält eine Goolge-Karte welche die Adressen der Events anzeigt, die direkt unter der Karte angezeigt werden. Die Events werden direkt auf EventJunkie erstellt und anschließend mit Facebook, Flickr oder Twitter verlinkt. So könnten etwa Bilder von Flickr oder Kommentare von Facebook oder Twitter zusätzlich im Event angezeigt werden. Auch wenn noch keine Parties im Kalender eingetragen sind, werden aktuelle Parties von Goabase im Kalender angezeigt.

Man kann Informationen zu den Events auch ohne einen Account sehen. Jedoch können Events nur dann erstellt werden, wenn man einen gültigen Account bei EventJunkie hat. Zu diesem Zweck kann man sich jederzeit mit seiner Email Adresse registrieren.

EventJunkie soll ein breites Publikum ansprechen. Die Zielgruppe sind Leute die gerne auf Parties gehen und auf einer eigenen Seite, gesammelte Informationen zu Parties erhalten wollen. Diese Event Plattform soll kein soziales Netzwerk darstellen, weshalb Kommunikation zwischen den einzelnen Benutzern nicht möglich ist.

1.2 Objectives

1.2.1 Funktionale Anforderungen

1. Die Plattform soll die Möglichkeit bieten, Events effektiv zu managen. Dazu zählt:
   - Events erstellen.
   - Events übersichtlich darstellen.
   - Events mit Twitter, Facebook und Flickr verlinken.

1.2.2 Nichtfunktionale Anforderungen

1. Es soll ein Responsive Design verwendet werden.
2. Realisiert soll das Projekt mit dem PHP-Framework Yii2 werden, welches mindestens die PHP-Version 5.4 voraussetzt. Als Datenbank soll eine MySQL-Datenbank eingesetzt werden.

2 Use Cases

2.1 Roles and Actors

2.1.1 Administrator

Der Administrator hat das Recht, Events zu löschen, verändern kann er sie nicht.

2.1.2 Registrierter User

Der registrierte User kann Events erstellen, updaten und löschen. Er kann seinen erstellten Event anschließend mit einem oder mehreren sozialen Netzwerk/en verbinden oder nicht. Sollte das Event verbunden werden, wird dieser gekennzeichent.

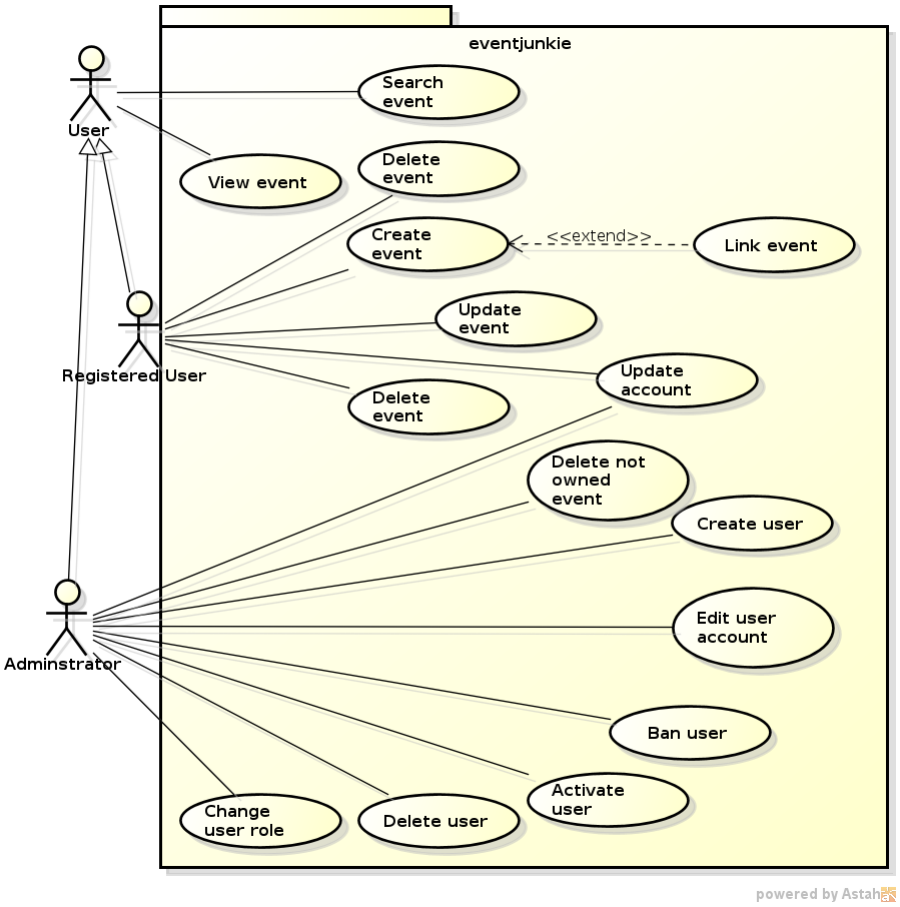
2.1.3 Unregistrierter User

Der unregistrierte User hat nur die Möglichkeit, die Event-Wall einzusehen, auf der alle von den registrierten Usern erstellten Events zu sehen sind.

2.1.4 System

Das System ist verantwortlich für die Aktualisierung der Events, die mit einem oder mehreren sozialen Netzwerk/en verbunden sind. Wenn sich zum Beispiel die Location bei einer auf Facebook geposteten Veranstaltung ändert, soll dies auf unserer Plattform angezeigt werden.

2.2 Use Case Diagrams



2.3 Textual Use Case Descriptions

User

2.3.1 RegisterUser

Use Case ID: 1

Title: RegisterUser

Scope: EventJunkie

Actors: User

Preconditions: Ein nicht eingeloggter User möchte einen Account auf EventJunkie erstellen.

Success Guarantees: Der User ist registriert und kann sich mit seinen Usernamen oder seiner Email Adresse einloggen.

Trigger: Ein User klickt in der Navigationsleiste auf Registrieren. Er wird zur Registrieungsseite weitergeleitet.

Main Success Scenario:

1. Ein User klickt in der Navigationsleiste auf Registrieren.
2. Der User muss seine Email und Usernamen eingeben und ein Passwort auswählen.
3. Der User bekommt eine Email mit einen Activierungslink.
4. Wenn der User den Link folgt ist der Registrierungprozess beendet und der User ist nun ein RegisteredUser.

Exceptions: 1. Die Email Adresse oder der Username des Users ist bereits vergeben.
2. Die Email Adresse ist Fehlerhaft.

Version: 0.1

Last updated: 30.10.2014

Issues/Comments:

2.3.2 CreateEvent

Use Case ID: 2

Title: CreatEvent

Scope: EventJunkie

Actors: RegisteredUser

Preconditions: Ein registrierter und eingeloggter User möchte ein Event erstellen.

Success Guarantees: Der RegisteredUser wird durch eine Erfolgsmeldung darauf hingewießen das, dass Event erfolgreich erstellt worden ist.

Trigger: Der RegistredUser klickt in der Navigationsleiste auf "Create event".

Main Success Scenario:

1. Der Registrierte und eingeloggte User klickt in der Navigationsleiste auf "Create event".
2. Der RegisteredUser gibt einen Namen, Adresse und ein Datum für ein Event ein. Optional kann der RegisteredUser einen End-Datum für ein Event (z.b ein Festival) eingeben. Auserdem kann auch optional eine Beschreibung und ein Eventbild hinzugefügt werden.

3. Der RegisteredUser wird über die erfolgreiche Erstellung des Event informiert.

Exceptions:

1. Startzeitpunkt ist in der Vergangenheit.
2. Endzeitpunkt ist vor dem Startzeitpunkt.
3. Angegebene Links sind ungültig.
4. Fehlerhafte Eingabe. Obligatorische Felder wurden ausgelassen.
5. Adresse ist keine Zeichenkette die in Koordinaten umgewandelt werden kann.
6. Dateiendung des Bildes ist ungültig (Nur jpg, gif oder png).

Version: 0.1

Last updated: 30.10.2014

Issues/Comments:

2.3.3 ViewEvent

Use Case ID: 3

Title: ViewEvent

Scope: EventJunkie

Actors: User, RegisteredUser oder Administrator

Preconditions: Der User, RegisteredUser oder Administrator befindet sich auf der Event-Wall Seite.

Success Guarantees: Der User, RegisteredUser oder Administrator sieht Informationen zu dem gewünschten Event auf einer eigenen Seite.

Trigger: Der User, RegisteredUser oder Administrator klick auf der Event-Wall auf das anzuzeigende Event.

Main Success Scenario:

1. Der User, RegisteredUser oder Administrator möchte nähere Informationen zu einen Event erhalten. Dazu klickt er direkt auf das Event bzw. auf das Eventbild.
2. Der User, RegisteredUser oder Administrator wird auf eine eigene Seite weitergeleitet wo er nähere Informationen zu den Event sieht.

Exceptions:

1. Das Event existiert nicht mehr.

Version: 0.1

Last updated: 30.10.2014

Issues/Comments:

2.3.4 UpdateEvent

Use Case ID: 4

Title: UpdateEvent

Scope: EventJunkie

Actors: RegisteredUser

Preconditions: Ein registrierter und eingeloggter User möchte sein eigenens Event bearbeiten und befindet sich auf der Seite seines Events.

Success Guarantees: Der RegisteredUser wird durch eine Erfolgsmeldung darauf hingewießen das, dass sein Event erfolgreich bearbeitet worden ist.

Trigger: Der RegistredUser sieht als Ersteller des Events einen Edit-Button. Wenn dieser Button gedrückt wird, sieht der RegistredUser ein Formular wo er das Event bearbeiten kann.

Main Success Scenario:

1. Der Registrierte und eingeloggte User klickt auf der Event-Wall auf ein von ihm erstelltes Event.
2. Der Ersteller des Events sieht auf der Ansichtsseite des Events einen Edit-Button.
3. Nachdem dieser Button gedrück wurde, wird der Registrierte und eingeloggte User auf eine Seite mit einem Formular geleitet wo dieser sein Event bearbeiten kann.

Exceptions:

1. Startzeitpunkt ist in der Vergangenheit.
2. Endzeitpunkt ist vor dem Startzeitpunkt.
3. Angegebene Links sind ungültig.
4. Dateiendung des Bildes ist ungültig (Nur jpg, gif oder png).
5. Bild ist zu groß.

Version: 0.1

Last updated: 30.10.2014

Issues/Comments:

2.3.5 SearchEvent

Use Case ID: 5

Title: SearchEvent

Scope: EventJunkie

Actors: RegisteredUser

Preconditions: Ein User, RegistredUser oder ein Administrator befindet sich auf der Event-Wall und möchte ein Event suchen.

Success Guarantees: Der User, RegistredUser oder der Administrator sieht als Suchergebniss das gewünschte Event oder eine ensprechende Meldung das, dass gesuchte Event nicht gefunden wurde.

Trigger: Im Suchfeld auf der rechten Seite der Event-Wall wird eine Suchanfrage eingeben und mit dem Search-Button bestätigt.

Main Success Scenario:

1. Der User, RegistredUser oder der Administrator will einen Event suchen.
2. Dazu gibt man entweder den Namen des zu suchenden Events ein oder die Adresse des Events. Die Suchanfrage wird mit dem Search-Button bestätigt.
3. Als Erebniss werden die zu der Suchanfrage passende Events angezeigt oder eine enpsrechende Meldung ausgegeben das kein passendes Event gefunden wurde.

Exceptions:

1. Es wir nach einen Ort/Adresse gesucht die nicht vorhanden ist.

Version: 0.1

Last updated: 30.10.2014

Issues/Comments:

2.3.6 Update account

Use Case ID: 7

Title: Update account

Scope: EventJunkie

Actors: RegisteredUser, Administrator

Preconditions: Der User ist registriert und eingeloggt.

Success Guarantees: User wurde erfolreich geändert.

Trigger: Der User möchte seine Benutzerdaten ändern.

Main Success Scenario:

1. Der RegisteredUser ruft seine Profilseite mit einem Klick auf seinen Usernamen auf.
2. Die zu änderten Felder werden bearbeitet.
3. Mit einem Klick auf "Update" wird der User gespeichert.

Exceptions:

1. Der Benutzername ist bereits vorhanden.
2. Die Email ist bereits vorhanden.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.7 Delete event

Use Case ID: 8

Title: Delete event

Scope: EventJunkie

Actors: RegisteredUser

Preconditions: Der User ist als User eingeloggt und will einen eigenen Event löschen.

Success Guarantees: Der Event wurde erfolgreich gelöscht.

Trigger: Der User will seinen Event löschen.

Main Success Scenario:

1. User ruft "ViewEvent" auf.
2. User klickt auf "Löschen".

Exceptions:

1. Der Event existiert nicht mehr.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.8 Delete not owned event

Use Case ID: 9

Title: Delete not owned event

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: Der Event wurde erfolgreich gelöscht.

Trigger: Ein Event verstößt gegen die Regeln und soll deswegen gelöscht werden.

Main Success Scenario:

1. Der Administrator ruft die Auflistung der Events oder die Ansicht eines einzelnen Events auf.
2. Der Adminstrator klick auf "Löschen" oder den Papierkorb.
3. Der Löschvorgang wird bestätigt.

Exceptions:

1. Der Event existiert nicht mehr.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.9 Create user

Use Case ID: 10

Title: Create user

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: User wurde erfolreich erstellt.

Trigger: Ein neuer Account soll händisch angelegt werden. Der Administrator ruft die User-Ansicht auf.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.

2. Dann wird das "Create User"-Formular aufgerufen.
3. Der Administrator gibt alle benötigten User-Daten in die Textfelder ein.
4. Die Aktion wird mit "Create" bestätigt.

Exceptions:

1. Der Benutzername ist bereits vorhanden.
2. Die Email ist bereits vorhanden.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.10 Edit user account

Use Case ID: 11

Title: Edit user account

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: User wurde erfolreich geändert.

Trigger: Das Passwort des Users soll zurückgesetzt oder der Benutzername oder die Email geändert werden, da diese gegen die Regeln verstoßen.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.
2. Der Administrator klickt auf das Editieren-Symbol des jeweiligen Users.
3. Die zu änderten Felder werden bearbeitet.
4. Mit einem Klick auf "Update" wird der User gespeichert.

Exceptions:

1. Der Benutzername ist bereits vorhanden.
2. Die Email ist bereits vorhanden.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.11 Ban user

Use Case ID: 12

Title: Ban user

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: User wurde erfolreich gebannt.

Trigger: Der User soll gebannt werden, da dieser mehrmals negativ aufgefallen ist.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.
2. Der Administrator klickt auf das Editieren-Symbol des jeweiligen Users.
3. Es wird eine Bann-Grund eingegeben.
4. Das Häckchen zum Bannen wird gesetzt.
5. Mit einem Klick auf "Update" wird der User gespeichert.

Exceptions:

Der User ist bereits gebannt.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.12 Activate user

Use Case ID: 13

Title: Activate user

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: User wurde erfolreich aktiviert.

Trigger: Der User soll aktiviert werden, da dies nicht durch eine Bestätigungsmail möglich ist.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.
2. Der Administrator klickt auf das Editieren-Symbol des jeweiligen Users.
3. "Activate" wird ausgewählt.
4. Mit einem Klick auf "Update" wird der User gespeichert.

Exceptions:

1. Der User ist bereits aktiviert.

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.13 Change user role

Use Case ID: 14

Title: Change user role

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: Dem User wurde erfolreich eine andere Rolle zugewiesen.

Trigger: Dem User soll entweder die Rolle Administrator oder User zugewiesen werden.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.
2. Der Administrator klickt auf das Editieren-Symbol des jeweiligen Users.
3. Die zuvergebene Rolle wird ausgewählt.
4. Mit einem Klick auf "Update" wird der User gespeichert.

Exceptions:

Version: 0.1

Last updated: 31.10.2014

Issues/Comments:

2.3.14 Delete user

Use Case ID: 15

Title: Delete user

Scope: EventJunkie

Actors: Administrator

Preconditions: Der User ist als Administrator registriert und eingeloggt.

Success Guarantees: User wurde erfolreich gelöscht.

Trigger: Das Passwort des Users soll zurückgesetzt oder der Benutzername oder die Email geändert werden, da diese gegen die Regeln verstoßen.

Main Success Scenario:

1. Der Administrator ruft die Administrations-Interface der User-Ansicht auf.
2. Der Administrator klickt auf das Löschen-Symbol des jeweiligen Users.
3. Der Löschvorgang wird bestätigt.

Exceptions:

1. Der User existiert nicht.

Version: 0.1

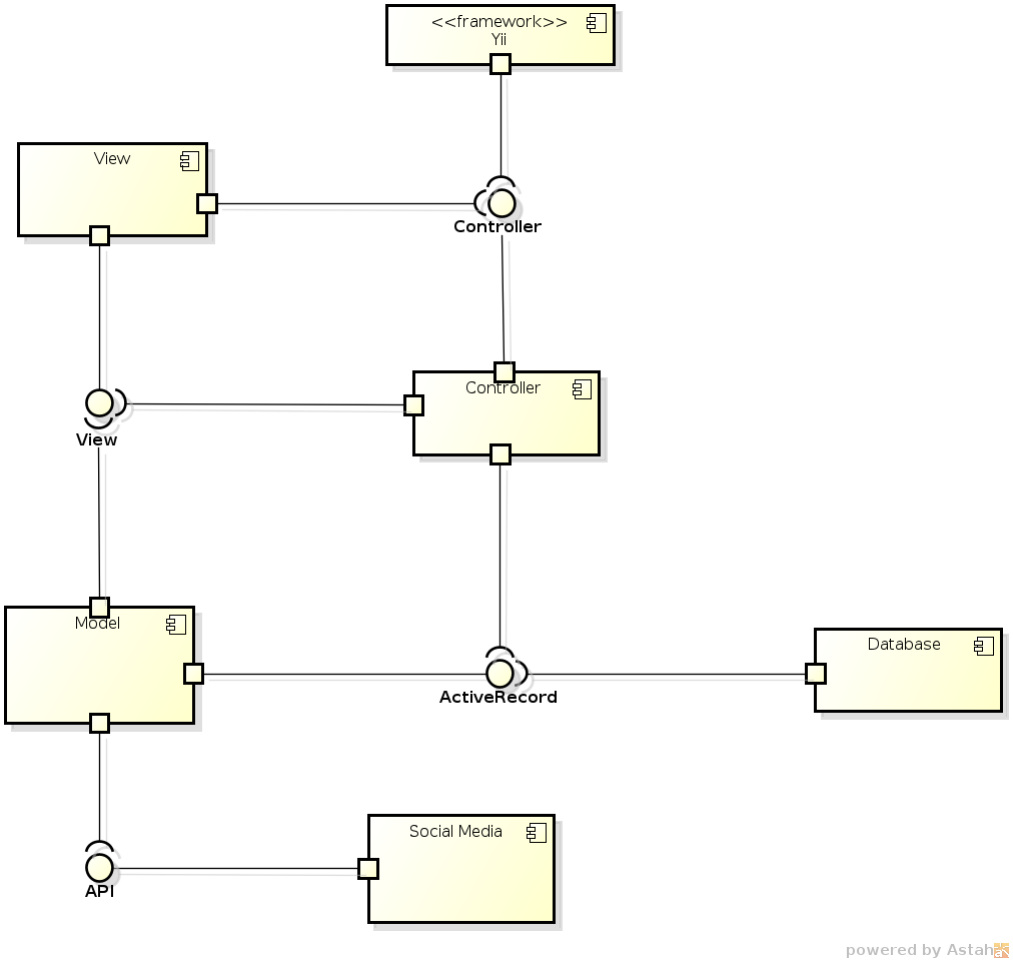
Last updated: 31.10.2014

Issues/Comments:

3 Architecture

3.1 Overview

Da unser Projekt eine Online-Eventplattform ist, haben wir uns für eine Model-View-Controller Architektur entschieden. Um dieses Modell effizient umzusetzen, verwenden wir das Yii-Framework in der Version 2.0. Yii ist ein MVC-Framework für php. Detaillierte Informationen finden Sie unter http://www.yiiframework.com.



3.2 Architecture for Component Model

3.2.1 Design Considerations

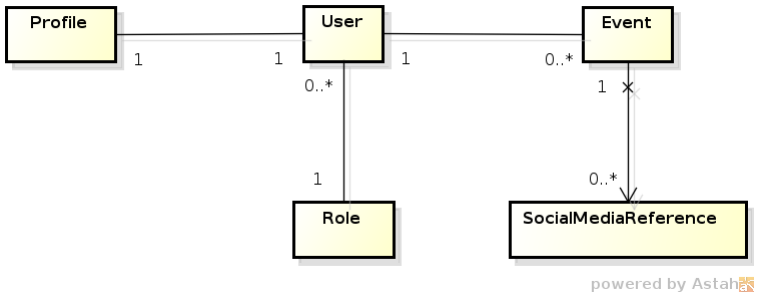
SocialMediaReference - Nach der API-Abfrage werden die Objekte (.json-Objekt) zerlegt und in ein neues Array geschrieben mit folgender Struktur:

```
socialmedia = [
    ['images' => [
        'thumbnail',
        'original' ,
    ]],
    ['comments' => [
        'title',
        'date',
```

```
            'text',
            'author',
            'origin',
        ]],
];
```

Diese Array wird nicht in der Datenbank gespeichert, sondern verbleibt in einem FileCache.

3.2.2 Overview of data design



3.2.3 Interfaces to/from internal and external components

Der Zugriff auf das Model erfolgt über das angebotene Controller-Interface. Das Model selbst greift auf die Datenhaltung über das Interface ActiveRecord zu, welches jedes Element im Model implementiert. Auf die Daten in den Social-Media Plattformen wird über die API der Anbieter zugegriffen.

3.2.4 Cross References

Diese Komponente wird beim Ausführen jedes Use-Cases verwendet.

3.3 Architecture for Component Controller

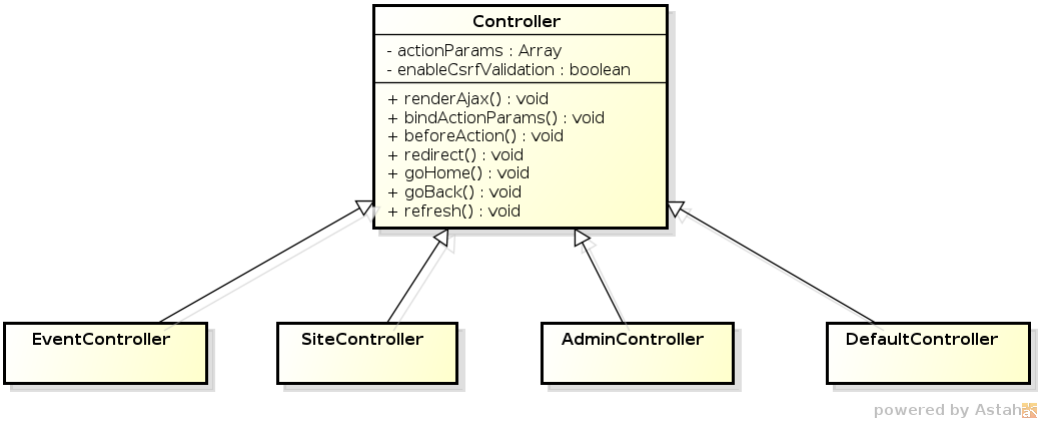
3.3.1 Design Considerations

Das Design ist durch das Yii-Framework vorgegeben.

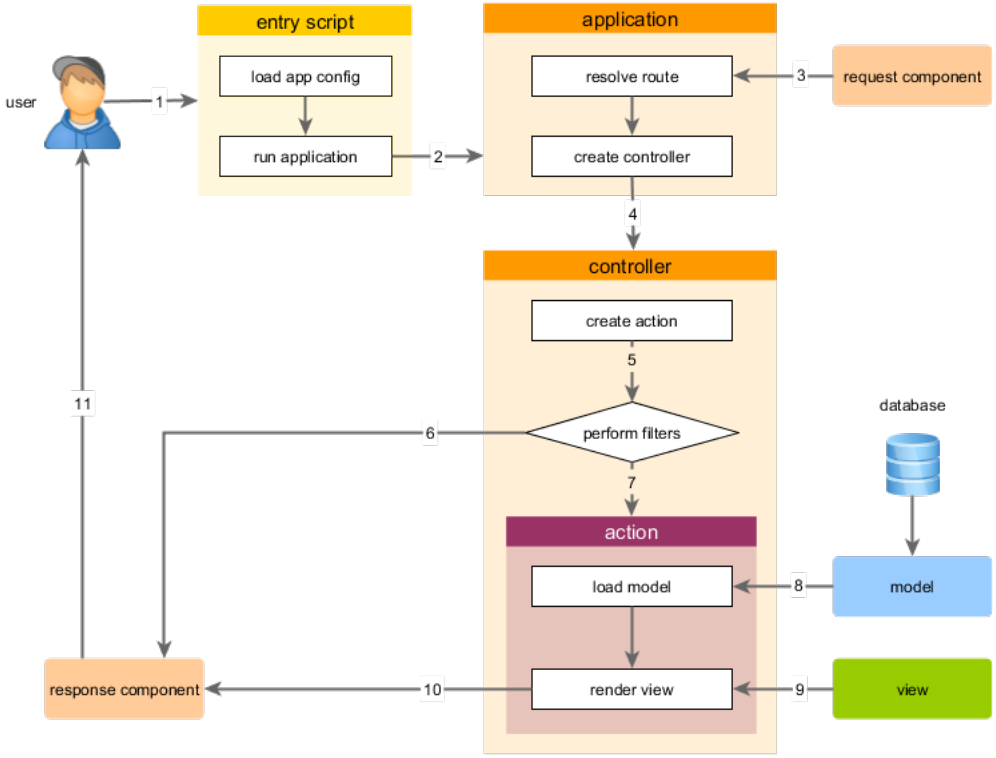
3.3.2 Overview of data design

Da das Request-Response Handling durch das Framework vorgegeben ist, haben wir hier zusätzlich ein Diagramm eingebunden, dass von der Homepage des Frameworks (URL: http://www.yiiframework.com/doc-2.0/guide-runtime-overview.html) stammt. Dies beschreibt das Request-Response Handling des Frameworks.

1. Die Controller, die für unsere Anwendung erstellt wurden



2. Das oben erwähnte Diagramm



3.3.3 Interfaces to/from internal and external components

Die Controller verwenden das View- und das ActiveRecord-Interface. Angeboten wird das Controller-Interface.

3.3.4 Cross References

Diese Komponente wird beim Ausführen jedes Use-Cases verwendet.

3.4 Architecture for Component View

3.4.1 Design Considerations

Auch hier ist anzumerken, dass das Request-Response Handling vom Framework vorgegeben ist.

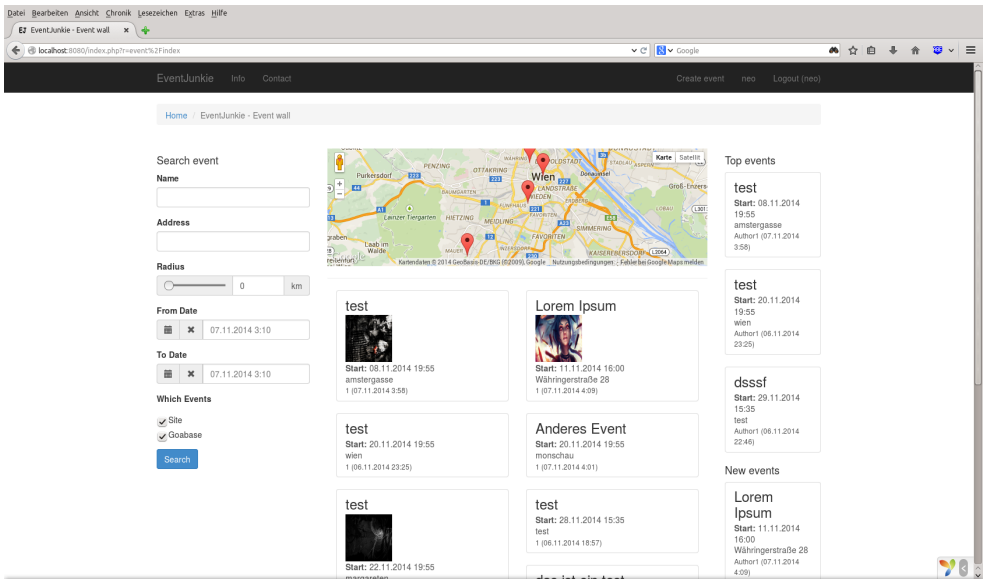
3.4.2 Interfaces to/from internal and external components

Views greifen auf die Model-Komponente über das Controller-Interface zu. Von View-Komponente angeboten wird das gleichnamige View-Interface.

3.4.3 User Interface

Für unser gesamtes Interface verwenden wir Bootstrap und halten uns daher an dessen Konventionen. Dadurch verwenden wir auch "Responsive Design". Für Google Maps und Validierungen von Eingaben werden "rules"-Validierungen des Yii2-Frameworks mittels Ajax und zusätzlich serverseitig, sowie selbstgeschriebene Validierungsfunktionen verwendet.

Event wall



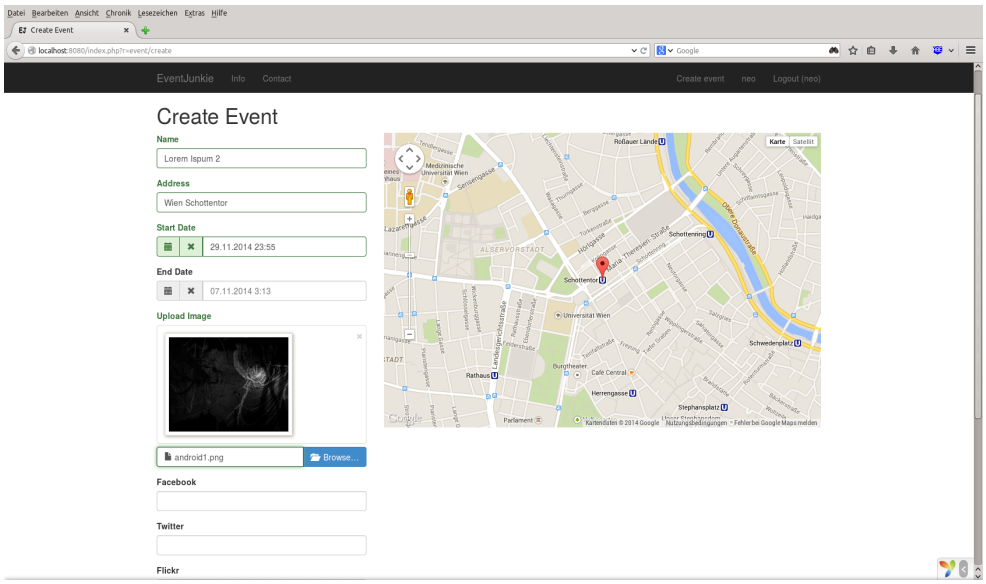
Use cases:

- Search event: Wird durch das Formular auf der linken Seite umgesetzt.

Actions:

- event/actionIndex: Generiert eine Auflistung aller Events inklusive Such-Formular. Top-Events werden nach Klicks ausgewählt und neue Events nach Erstellungsdatum. Die Events die bald stattfinden werden in der Mitte aufgelistet, nach Datum sortiert und auf der Google-Karte angezeigt.
- event/actionSearch: Events können nach Name, Datum, Adresse und dem Umkreis zur Adresse mit dem Suchformular gefiltert werden. Auch können nur Events der Seite oder von Goabase angezeigt werden.

Create event



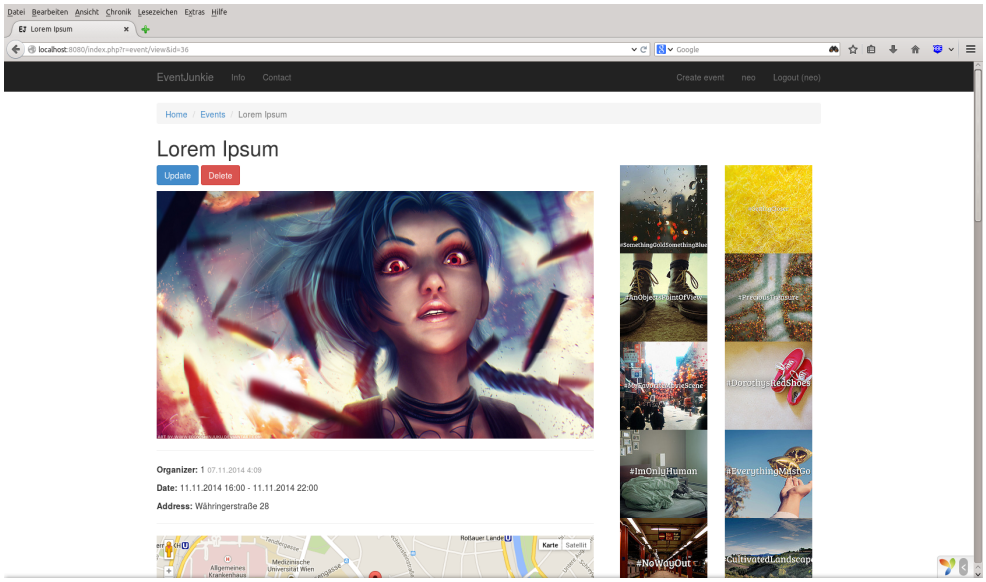
Use cases:

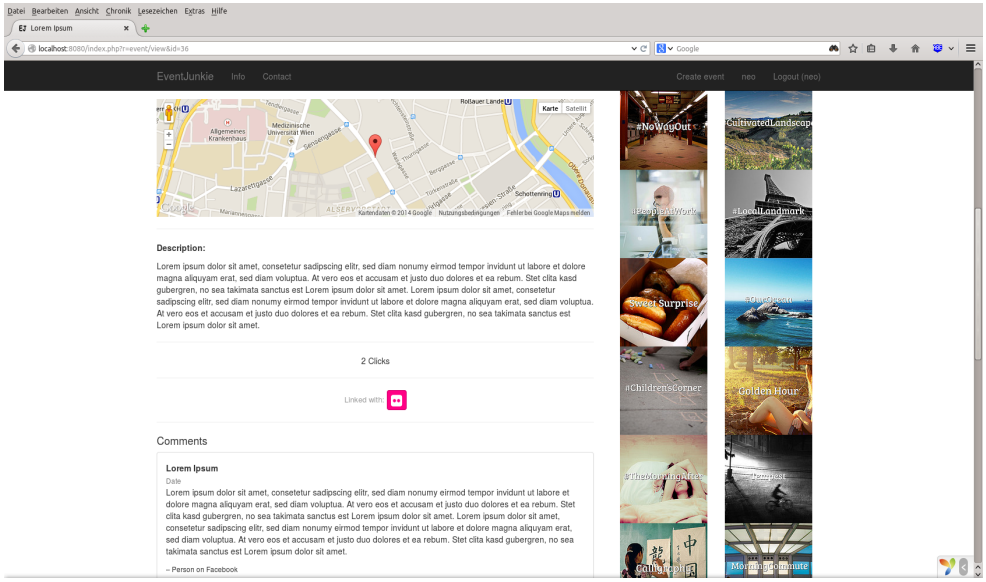
- Create event: Wird durch das Formular auf der linken Seite umgesetzt.
- Link event: Durch die Felder Twitter, Facebook und Flickr kann zu Events, Gallerien und Tweets verlinkt werden.

Actions:

- event/actionCreate: Es wird ein Event-Model erzeugt, wodurch alle Eingabedaten validiert werden. Name, Adresse, Start-Datum sind dabei Pflicht. Nachdem das Formular abgeschickte wurde, werden die Koordinaten über die Google API abgefragt und inklusive der angegebenen Daten abgespeichert. Auch sollen die Koordinaten direkt "onChange" eines Textfeldes über die JavaScript API von Google abgefragt werden und auf der Karte angezeigt werden.

Event





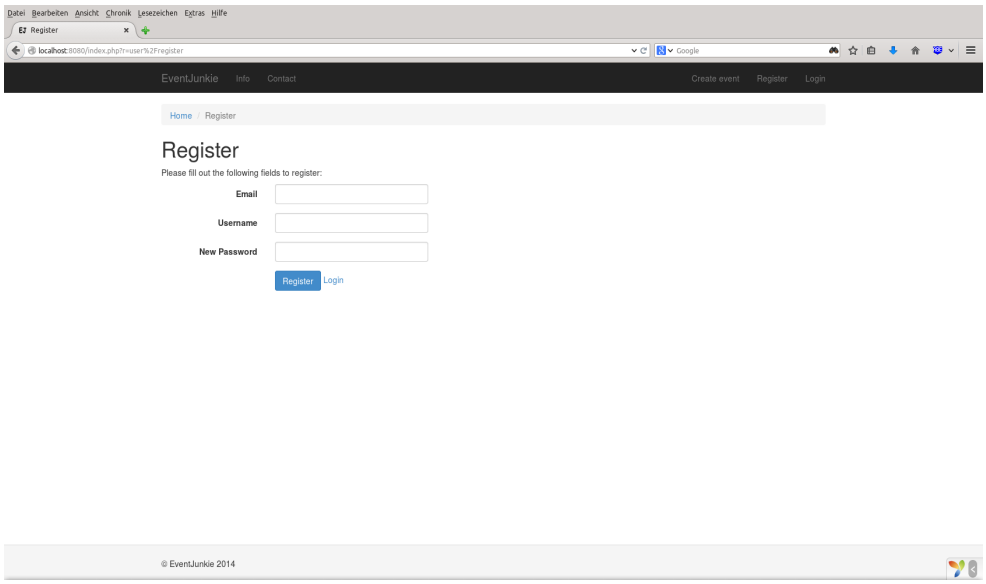
Use cases:

- View event
- Update event: Update-Button.
- Delete event: Delete-Button.

Actions:

- event/actionView: Das Event-Model zur jeweiligen Event-Id wird auf der Seite angezeigt. Die Verlinkungen zu Twitter, Flickr und co. werden überprüft. Die Daten werden über die jeweiligen APIs in ein socialmedia-Objekt geladen und dann die Inhalte auf der Seite ausgegeben.
- event/actionUpdate: Das Event-Model kann über dasselbe Formular wie bei createEvent geupdated und wieder abgespeichert werden.
- event/actionDelete: Das Event wird nach einer Abfrage gelöscht.

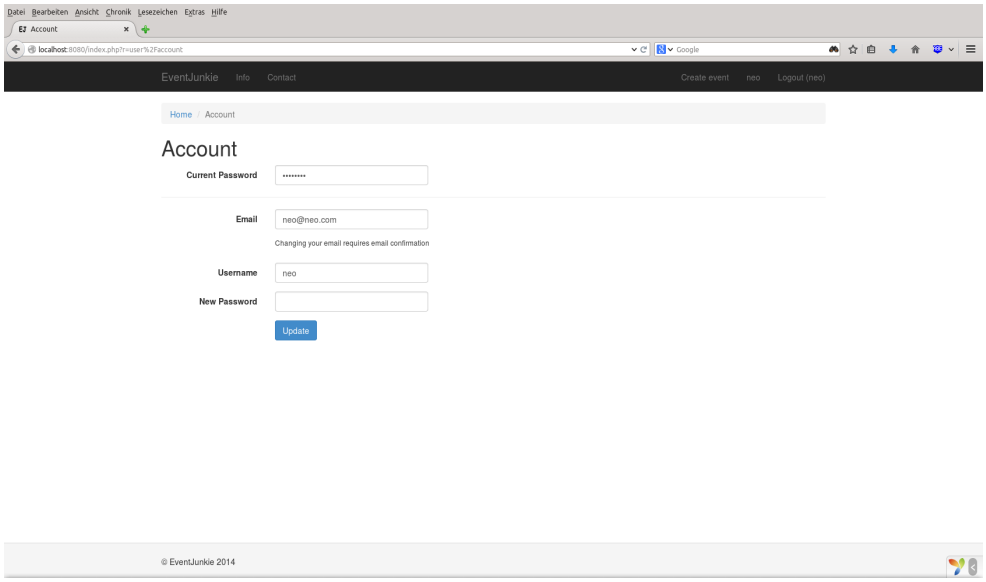
Register



Use cases:

- Register user

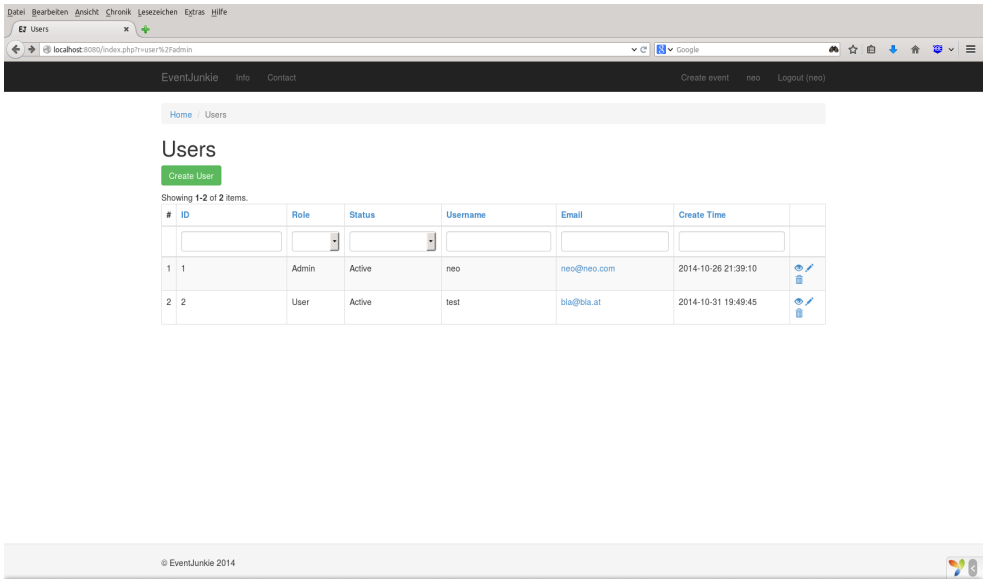
Own account



Use cases:

- Update account

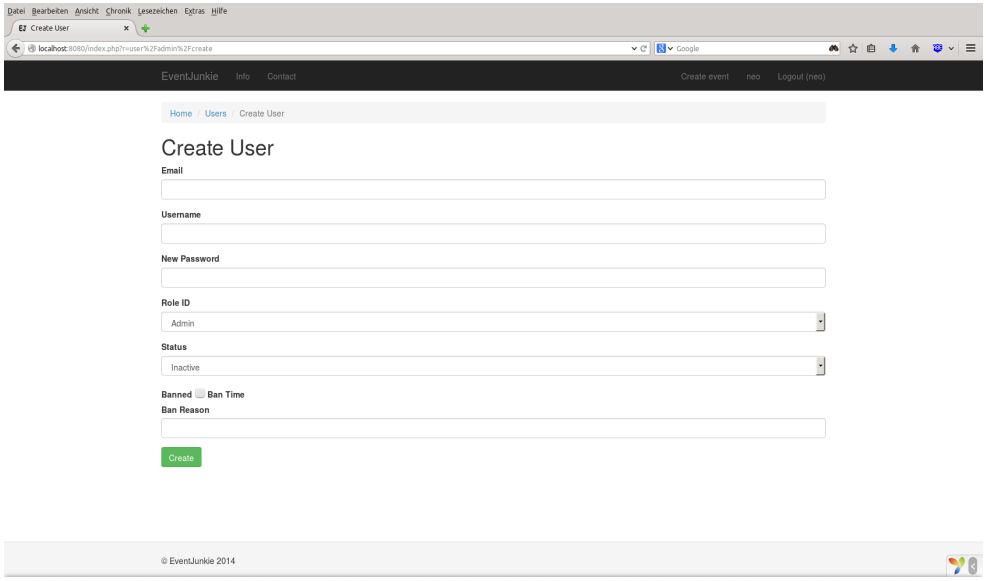
Administrator Interface



Use cases:

- Create user
- Update user
- Delete user

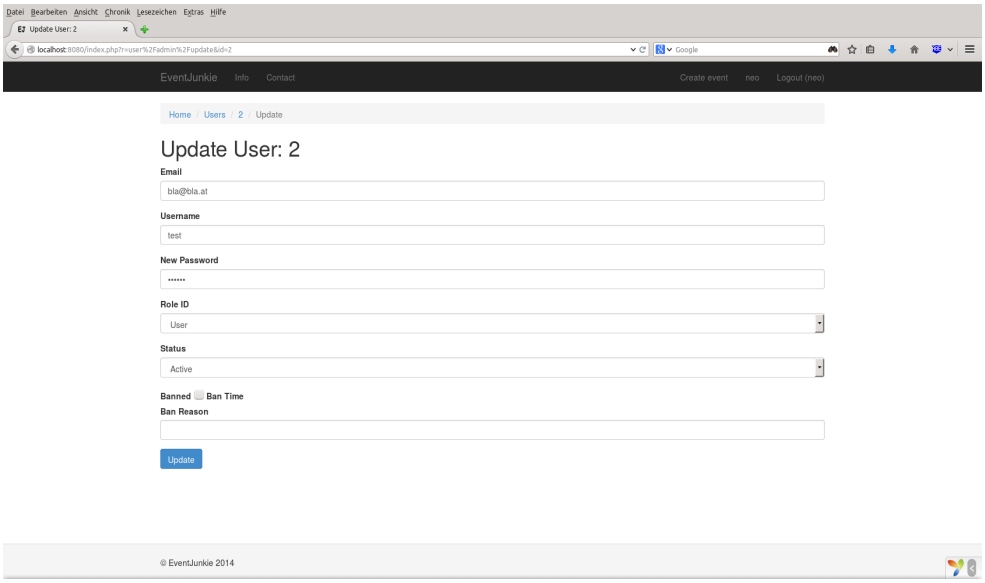
Create User



Use cases:

- Create user
- Change user role
- Activate user
- Ban user

Update User



Use cases:

- Update user
- Change user role
- Activate user
- Ban user

3.5 Architecture for Component Database

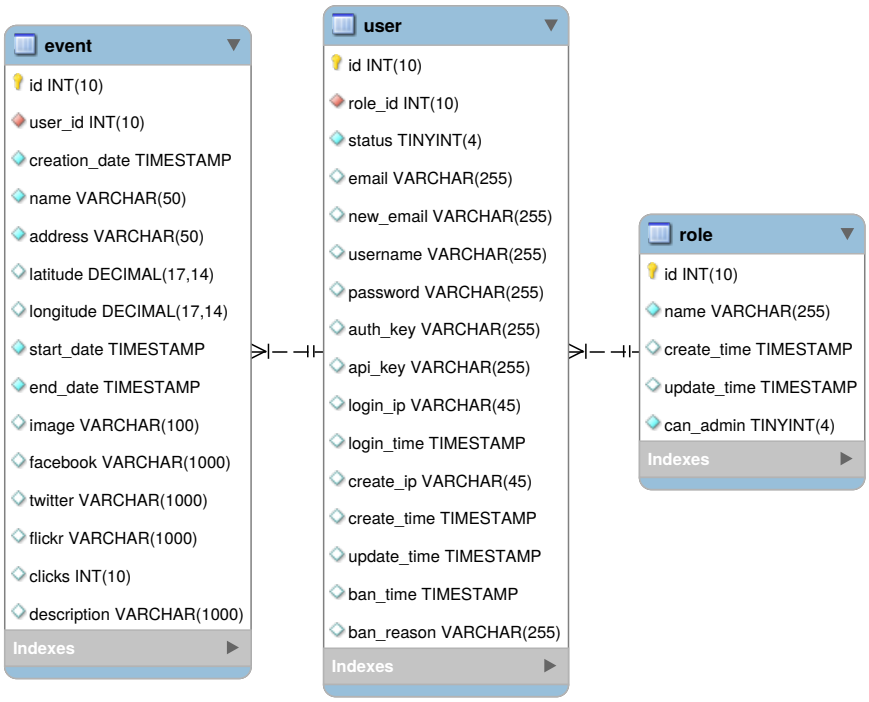
Als Datenbank wird eine MySQL-Datenbank verwerden. Alle zur User-Tabelle zugehörigen Tabellen wurden vom Yii2-User-Modul "Amnah" erstellt.

3.5.1 Design Considerations

Events sollen in der Datenbank gespeichert werden. Diese benötigen eine id um eindeutig identifizierbar zu sein, einen Namen und eine Description um vom User zuordbar zu sein. Start und End Datum werden als Zeitinformation benötigt, Clicks um Top Events auflisten zu können. Das Event-Image, wenn vorhanden wird in der Auflistung der Events anzeigt. Aus der Adresse werden Latitude und Longitude mit der Google Geolocating API geholt. Diese werden benötigt um Orte auf der Karte anzeigen zu können. Die Felder Facebook, Twitter und Flickr werden benötigt um von der jeweiligen Webseite jeweils einen Link zuweisen zu können. Bei Twitter sind dies Tweets und deren Bildern, bei Facebook Kommentare und Bilder von Veranstaltungen und Gruppen, bei Flickr Fotogallerien und Fotosets. Die User-Tabelle dient dazu User abzuspeichern und

die Rollen-Tabelle jeweils eine Administrator-Rolle oder User-Rolle zuzuweisen und somit die Berechtigungen zu setzen. User werden benötigt um unbrauchbare Events zu löschen.

3.5.2 Overview of data design



4 Project Management

4.1 Milestones and Schedules

Meilenstein	Datum
1 Dokumentation der Anforderungen und des Designs	**9.11.2014**
1.1 Anforderungen in Form eines Use Case Diagramm	
1.2 Uses Cases dokumentieren	
1.3 Softwaredesign dokumentieren	
2 Prototyp, Statusbericht	**14.12.2014**
2.1 Implementierung der Basisfunktionalitäten	
- Events erstellen/updaten/löschen/anzeigen/suchen	
- Validierung von Benutzereingaben	
- Anzeigen der Events auf GoogleMaps	
- Mash-up mit Facebook, Flickr, Twitter	
- Goabase-Events anzeigen/suchen	
- Löschen von Events als Admin	
2.2 Anfallende Designabänderungen dokumentieren	
2.3 Basisfunktionalitäten testen	
2.4 Bugs beheben	
2.5 Statusbericht erstellen	
3 Präsentation, Endbericht	**25.01.2014**
3.1 Präsentation erstellen	
3.2 Endbericht erstellen	
4 Finale Demo	**31.01.2014**
4.1 Implentierung eines Caches für die Mash-up-Seiten	
4.2 Überarbeitung des Prototyps	
4.3 Anfallende Designabänderungen dokumentieren	
4.4 Softwarekomponenten testen	
4.5 Bugs beheben	
5. Präsentation der Demo	**31.01.2014**

4.2 Planned Effort per Person

Die Lehrveranstaltung hat 6 ECTS. Pro Person sind das ungefähr 150 Stunden. Zieht man die erste Einheit und die Präsentationen der anderen Gruppen zum Endtermin ab sind es ungefähr 145 Stunden. Da wir aus einem 3er-Team bestehen sind dies daher ungefähr 435 Stunden, welche für die Lehrveranstaltung, bzw. das Projekt aufgewendet werden.

Person	Stunden
Oliver Guggenberger	**145**
1 (Designüberlegungen, Use Case Diagram, Use Cases User, Application Overview, Objectives)	20
2 (Events anzeigen/suchen, Goabase-Events anzeigen/suchen - Integration mit EventJunkie, Administration der Events, Testen, Bugs beheben, Statusbericht erstellen)	65
3 (Präsentation und Endbericht erstellen)	20
4 (Überarbeitungen, Testen, Bugs beheben)	59
5 (Präsentation)	1
Philipp Hiermann	**145**
1 (Designüberlegungen, Komponentendiagramm, Design der Komponenten Model, View, Controller, Objectives)	20
2 (Mashup mit Facebook, Basis-Caching der Mashup-Seiten)	65
3 (Präsentation und Endbericht erstellen)	20
4 (Überarbeitungen, Testen, Bugs beheben, Caching der Mashup-Seiten)	59
5 (Präsentation)	1
Sandra Markhart	**145**
1 (Designüberlegungen, Datenbankkompontente, Use Case Diagram, Use Cases Administrator, User-Interfaces, Projektmanagement)	20
2 (Events anzeigen/suchen/erstellen/updaten/löschen, Validierung der Benutzereingaben, Mashup mit Flickr, Twitter, Anzeigen der Event auf Google Maps, Testen, Bugs beheben, Statusbericht erstellen)	65
3 (Präsentation und Endbericht erstellen)	20
4 (Überarbeitungen, Testen, Bugs beheben)	59
5 (Präsentation)	1

Die Dokumentation und die Erstellung der Berichte wird von jedem Teammitglied gleichermaßen übernommen.