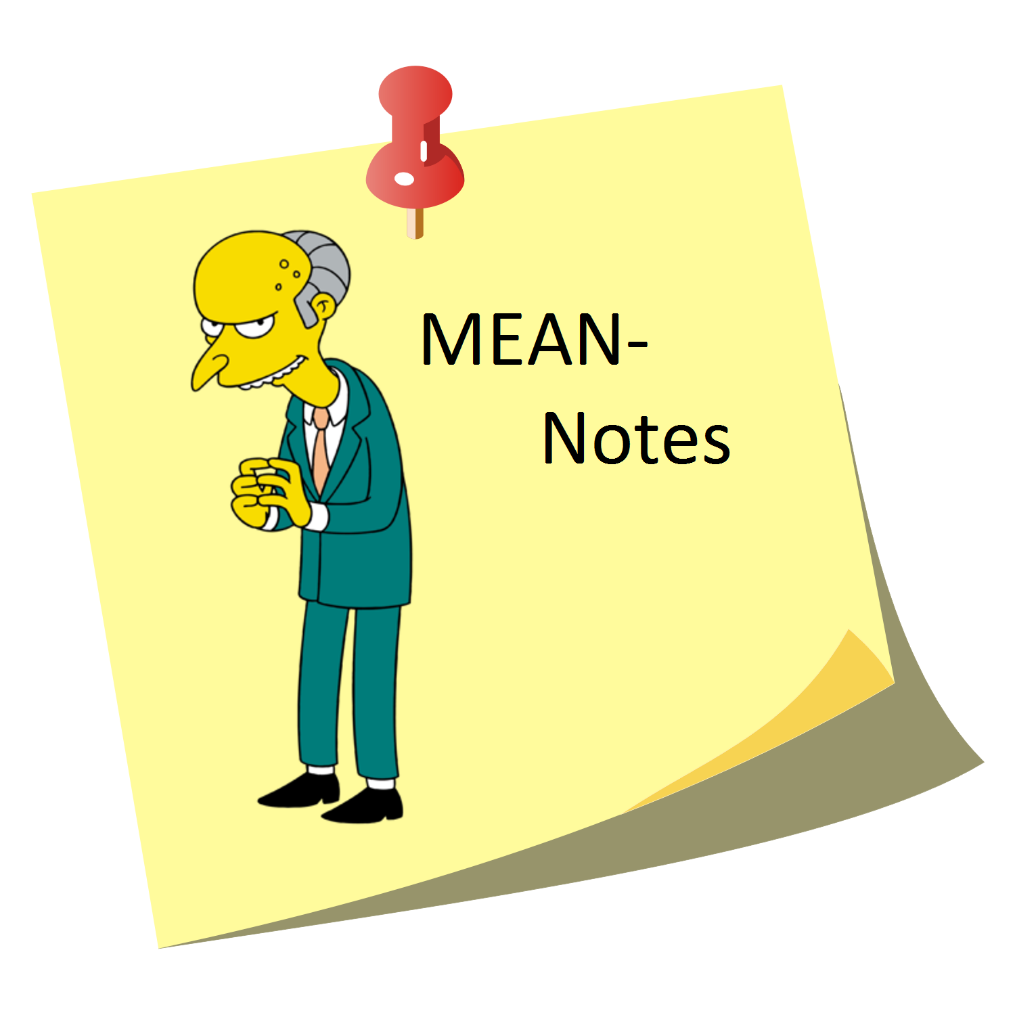
MEAN Notes

Im Fach Projektarbeit I

SRH Heidelberg – Wintersemester 2015/2016

Fakultät für Informatik - Angewandte Informatik



Dozent:

Prof. Dr. Mohammed Yass

Team:

Phillip Beck – Jovan Gerodetti – Lloyd Niebel

Inhaltsverzeichnis

[1. Aufgabenstellung und Grundidee 3](#_Toc442900813)

[I. Ideenfindung 3](#_Toc442900814)

[II. Projektstruktuplan, Zeitplanung und Arbeitpakete 4](#_Toc442900815)

[2. Konzeptionierung und Planung 7](#_Toc442900816)

[I. Mockups 7](#_Toc442900817)

[II. Technologien 10](#_Toc442900818)

[III. Datenmodelle 12](#_Toc442900819)

[3. Umsetzung 13](#_Toc442900820)

[I. Screenshots 13](#_Toc442900821)

[II. REST-Befehle 14](#_Toc442900822)

[4. Zukünftige Features 16](#_Toc442900823)

[5. Quellen 17](#_Toc442900824)

# Aufgabenstellung und Grundidee

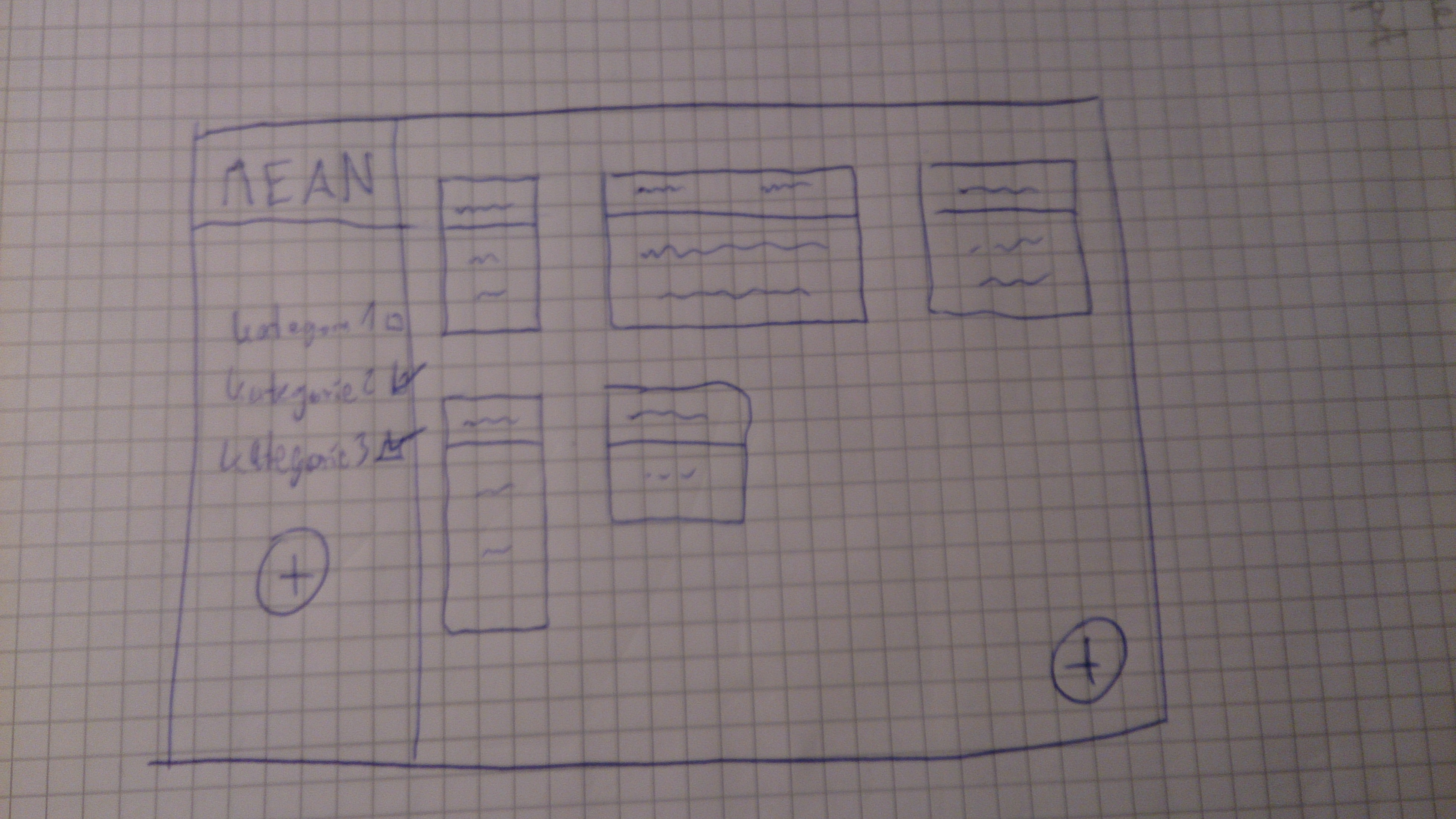
1. Ideenfindung

Die Aufgabenstellung im Fach Projektarbeit I bestand darin, ein Projekt auszuwählen, welches es innerhalb der fünf Wochen umzusetzen galt. Es waren keine Grenzen bei der Ideenfindung oder der Technologieauswahl gesetzt.

Nachdem wir einige Projektideen innerhalb der Gruppe besprochen hatten, haben wir uns schnell für ein Projekt entschieden, über das wir bereits zuvor geredet hatten.

Wir wollten eine einfache Plattform in Form eines Webservices erschaffen, in der der Nutzer Notizen anlegen, abheften, sortieren und archivieren (löschen) können. Die Sortierung soll über Kategorien erfolgen und kann individuell angepasst werden. Beispielsweise wären Kategorien wie „Haushalt“, „Familie“ und „Arbeit“ denkbar.

Erste Ideen bezüglich der Optik wurden schnell ent- und wieder verworfen bis wir eine erste Skizze hatten, die uns gefiel:



1. Projektstruktuplan, Zeitplanung und Arbeitpakete

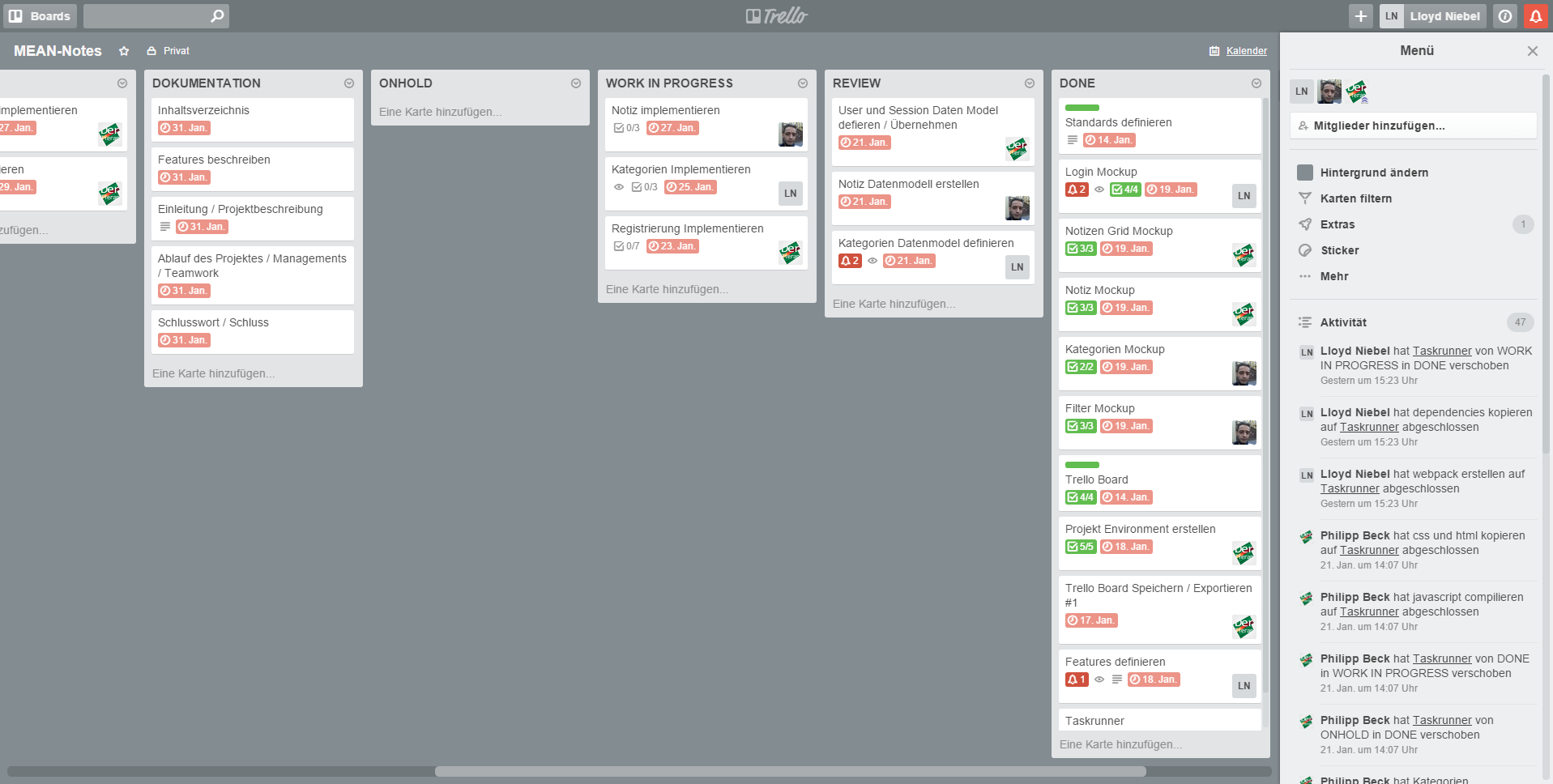
Für die Organisation, Planung und Koordination innerhalb des Teams haben wir uns für ein Trello Board entschieden. Dort wurden die Arbeitspakete erstellt und anschließen gemeinsam im Team bezüglich Arbeitsaufwand und Komplexität bewertet. Zusätzlich haben wir versucht ungefähr festzulegen wann welches Arbeitspaket abgeschlossen sein muss. Dies war insbesondere schwer, da wir nicht wussten wie lange wir für welche Features brauchen. Zwar haben wir einen erfahrenen Javascript-Entwickler im Team, die anderen beiden Mitglieder waren allerdings Anfänger.

Unsere grobe Zeitplanung war die folgende:

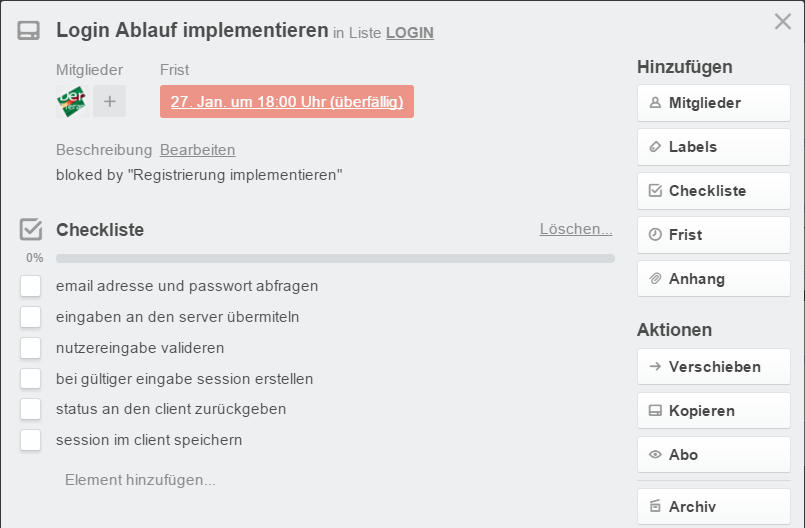
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Woche 1** | **Woche 2** | **Woche 3** | **Woche 4** | **Woche 5** |
| Planung und Skizzierung | Interaktive Mockups | Verwendbare GUI | Grundlegende Funktionalität | Funktionalität, Dokumentation |

In der **ersten Woche** wurde, nachdem wir uns für dieses Projekt entschieden hatten, die restliche Zeit für die Planung und Skizzierung des Projektes verwendet. In der **zweiten Woche** haben wir jeweils Mockups erstellt und verbessert. Diese wollten wir hochwertig und interaktiv, sodass man bereits ein gutes Bild bekommt, wie das Projekt am Ende aussieht. In **Woche drei** haben wir die Mockups umgesetzt und die GUIs erstellt. Diese konnte man zwar verwenden, es war allerdings keine Funktionalität dahinter. Trotzdem hatte man etwas Brauchbares zum Vorzeigen. In **Woche vier** haben wir die Funktionalitäten zu den GUIs implementiert. In der **fünften und letzten Woche** haben wir ebenfalls noch an den Funktionalitäten gearbeitet und währenddessen an diesem Dokument und an der PowerPoint gearbeitet. Wir haben bis zum letzten Tag an der Funktionalität gearbeitet, insbesondere der Login und die Kategorienverwaltung waren am Ende zeitlich knapp.

Unser Trello Board sah nach der Mockup-Erstellung - Woche 2 - folgendermaßen aus:



Wie man sieht haben wir uns Reiter erstellt mit den jeweiligen Zuständen der Arbeitspakete. Wenn beispielsweise jemand an einem Paket arbeiten möchte, schaut er ins Trello-Board und überprüft ob sich das Paket nicht im Reiter „WORK IN PROCESS“ befindet. Sollte es frei sein, so schiebt derjenige, der daran arbeiten möchte es in eben genannten Reiter und fängt mit der Arbeit an. Auf diese Weise ist das Arbeitspaket „blockiert“. Wenn man nun fertig ist schiebt man das Paket entweder in „REVIEW“, um es noch einmal von jemand anders überprüfen zu lassen oder man schiebt es zurück auf „ONHOLD“ – das Paket wäre in diesem Fall noch nicht fertig. Zusätzlich kann man noch ein Kommentar schreiben, sodass die anderen wissen wie der Status des jeweiligen Paketes ist.



Hier sieht man ein Paket in der Einzelansicht. Wir haben für jedes Arbeitspaket eine Checkliste mit Features erstellet, sodass einzelne bereits bearbeitete Pakete abgehakt werden können, ohne dass das ganze Paket als fertig bezeichnet werden muss. In diesem Fall sieht man, dass das Paket überfällig ist, bei diesem Paket hatten wir also eine frühere Fertigstellung ästimiert.

# Konzeptionierung und Planung

* 1. Mockups

Unsere Mockups wollten wir in diesem Projekt hochwertig und wenn möglich interaktiv bereitstellen. Die ersten haben wir mit [Ninjamock](https://ninjamock.com/) erstellt, sind dann allerdings schnell zu [Proto.io](https://proto.io/) gewechselt. Das Gute an Proto ist, dass man ein Mockup mit genau derselben Optik erstellen kann, wie die Technologie die man später benutzen möchte. In unserem Fall ist das Angular Material und mit Proto konnten wir so ein Mockup erstellen, welches fast genau wie die fertige Seite aussehen könnte.

Unsere Mockups sind unter folgendem [Link](https://free-tieran.proto.io/editor/index.cfm?id=e5154370-d283-4068-a2be-dbcd99e290fd) mit diesem Login-Daten anzusehen:

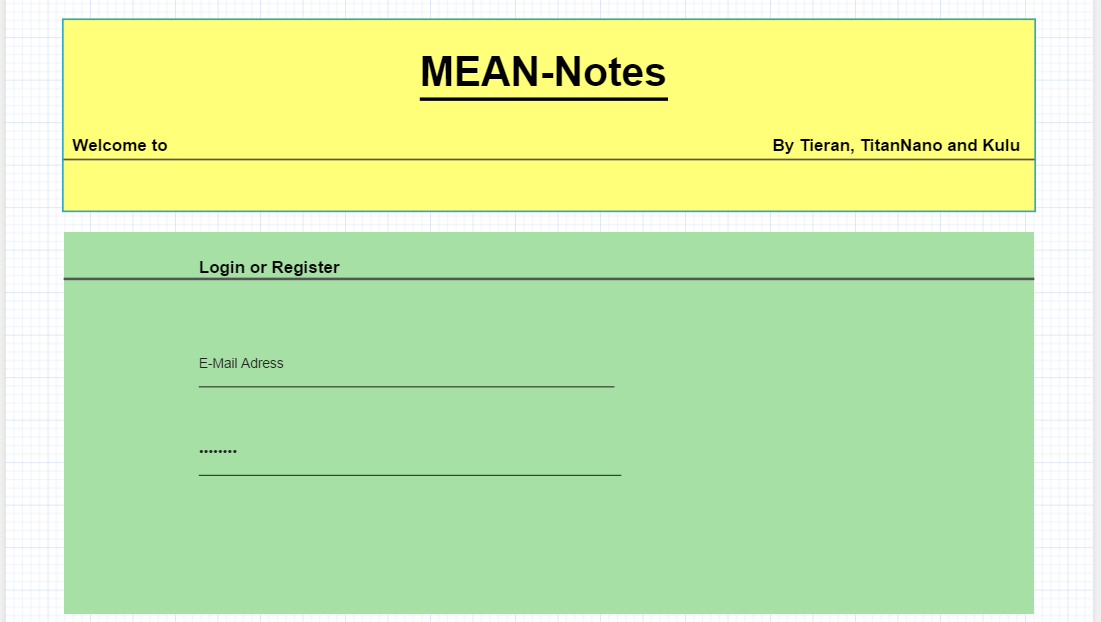
Nutzername j2237963@trbvm.com

Passwort Passwort

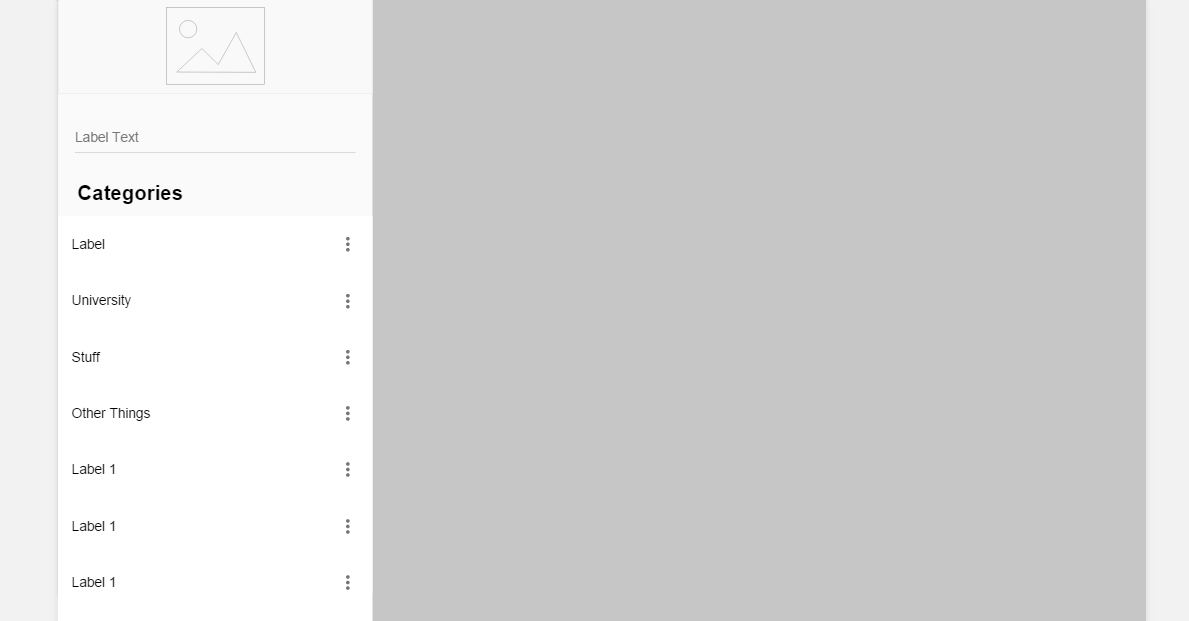
Das Mockup unserer Hauptseite sieht folgendermaßen aus – die Sidebar mit dem Menü ist hier nicht zu sehen. Über die genaue Anordnung der Buttons „Abgehakt“ und „Löschen“ hatten wir uns länger Gedanken gemacht und diese noch mehrere Male verworfen bzw. geändert.



Die Login Seite ist bisher äußerst rudimentär gehalten. Die Eingaben des Nutzers werden in dem Sinne nicht überprüft, dass keine E-Mail versendet wird, in der der Nutzer einen Link bestätigen muss. Dafür fehlt uns schlicht der SMTP-Server – und die Zeit. Die Registrierung funktioniert also mit E-Mail bzw. Nutzername und Passwort. Es wird lediglich bei der Registrierung überprüft, ob es sich um eine E-Mail Adresse handelt und ob diese bereits vorhanden ist. Die eigentliche Login Seite wurde am Schluss nur per Dialog umgesetzt.



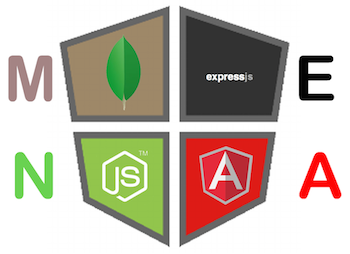
Hier ist ein Bild der Sidebar mit den Kategorien zu sehen. Oben hatten wir überlegt unser Logo bzw. einfach unseren Schriftzug einzufügen, und drunter das Feld um eine neue Kategorie hinzuzufügen. Dies haben wir allerdings später in der Umsetzung in ein Dialog Fenster ausgelagert, da es wohl nicht andauernd vorkommt, dass jemand eine neue Kategorie erstellt und somit das Feld eher unnötig stark präsent ist.



* 1. Technologien

Bei den Technologien haben wir uns – wie der Projektname bereits verrät – für den MEAN-Stack entschieden. Zwar hatten zwei der drei Teammitglieder nur Grunderfahrungen mit Javascript und somit wäre der traditionellere LAMP-Stack (Linux – Apache – MySQL – PHP) wohl die naheliegendere Entscheidung gewesen, aber wir wollten den neueren Stack ausprobieren um damit Erfahrungen zu sammeln und sicherer zu werden. Ebenso war die Kenntnis des MEAN-Stacks eine Voraussetzung für eines der Praktika, welches ein Teammitglied im Praxissemester abhalten würde.

Der MEAN-Stack hat die folgenden Bestandteile:



* MongoDB in der Persistenzschicht
* Express.js als Webserver
* Angular.js im Frontend
* Node.js auf der Serverseite



Als Entwicklungsumgebung haben wir uns für [Atom](https://atom.io/) mit diversen Erweiterungen entschieden.



Zur Versionsverwaltung haben wir Git bzw. GitHub.com im speziellen verwendet. Unser Projekt ist [hier](https://github.com/DerTieran/MEAN-Notes) in einem öffentlichen Repository zu finden. Git haben wir per Kommandozeile bzw. Git-BASH bedient.



Und wie bereits erwähnt haben wir ein Trello-Board für die interne Kommunikation und die Aufteilung der Aufgaben verwendet. Das Trello-Board ist [hier](https://trello.com/b/5fCwGgai/mean-notes) zu finden.



Zur allgemeinen Kommunikation, für kurze Fragen oder Terminabsprachen haben wir den WhatsApp Konkurrenten [Telegramm](https://telegram.org/) verwendet.

* 1. Datenmodelle

Die Datenmodelle sind bei MongoDB einfach gehalten. Dadurch, dass man jedwedes Objekt in der Datenbank speichern kann, ohne diese vorher durch Tabellen darauf vorzubereiten gestaltet sich der Aufwand vor der eigentlichen Nutzung der Datenbank als gering.

Folgende Datenmodelle wurden von uns verwendet:

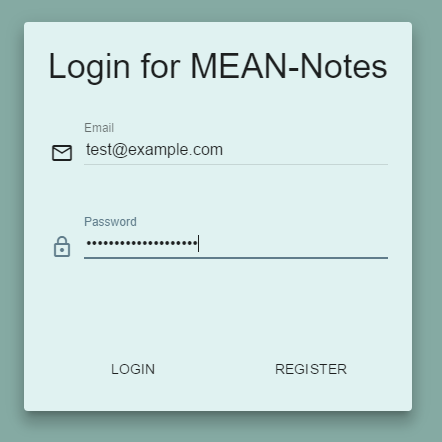
\* = unique

* tokens
  + \_id\* : {ObjectID}
  + userID : {ObjectID}
  + token : {String}
  + lastModifiedDate : {Date}
* users
  + \_id\* : {ObjectID}
  + Email\* : {String}
  + password : {String}
  + categories : [ ]
    - name : {String}
    - color : {String}
* notes
  + \_id\* : {ObjectID}
  + user : {ObjectID}
  + title : {String}
  + content : {String}
  + startdate : {Number}
  + enddate : {Number}
  + categories : [ ]
    - name : {String}
    - color : {String}

# Umsetzung

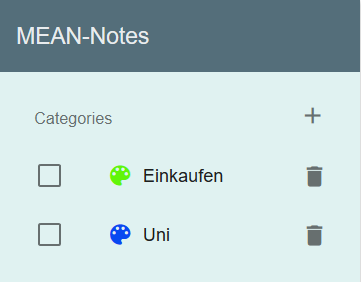
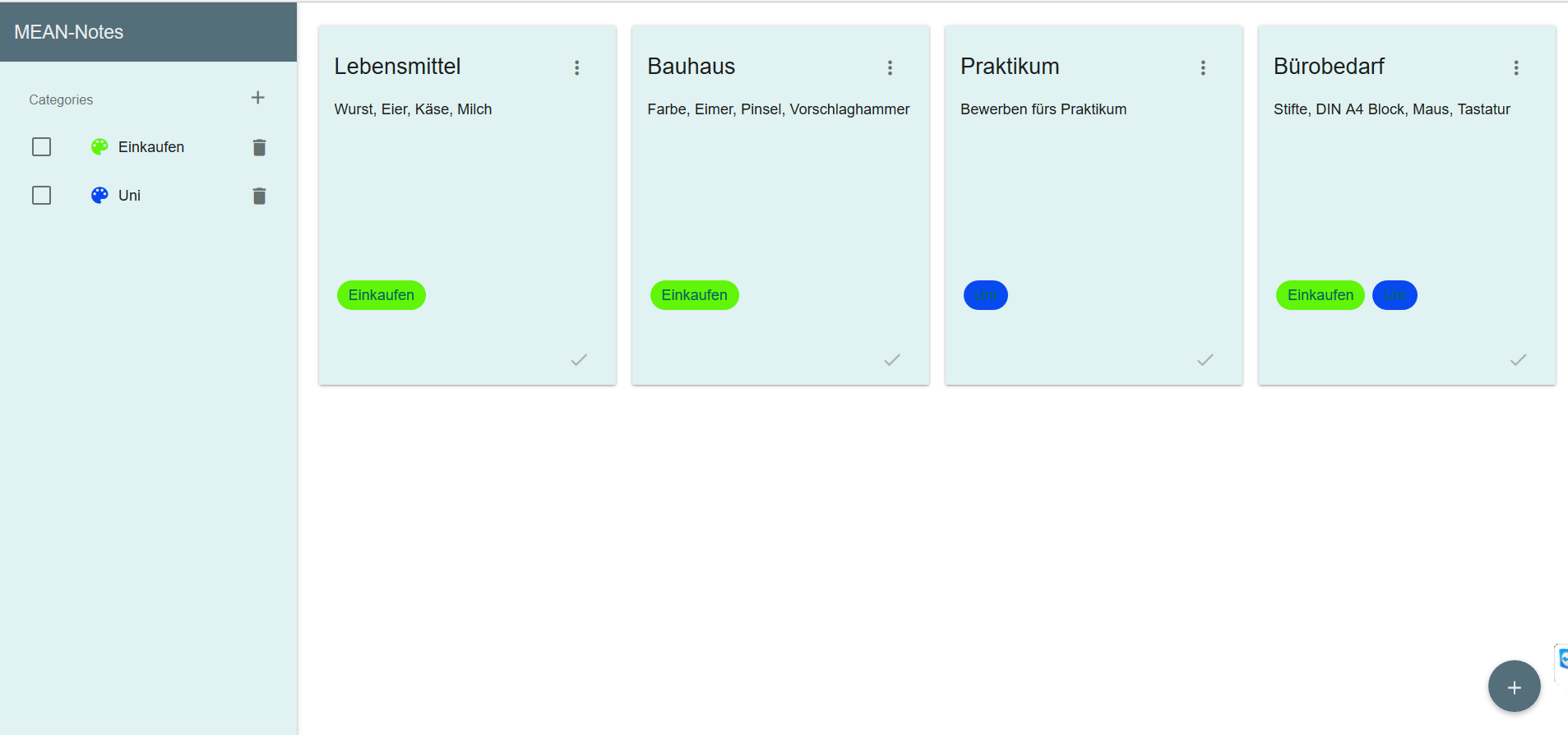
1. Screenshots

Nachfolgend zeigen wir Screenshots des MEAN-Notes Projektes zum Zeitpunkt der Abgabe bzw. Vorstellung.

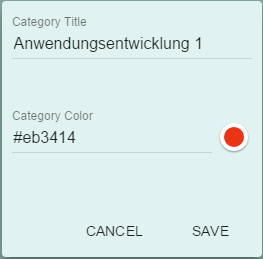


Der Login bzw. Registrierungs-Screen. Hier kann sich der User einloggen oder neu registrieren.

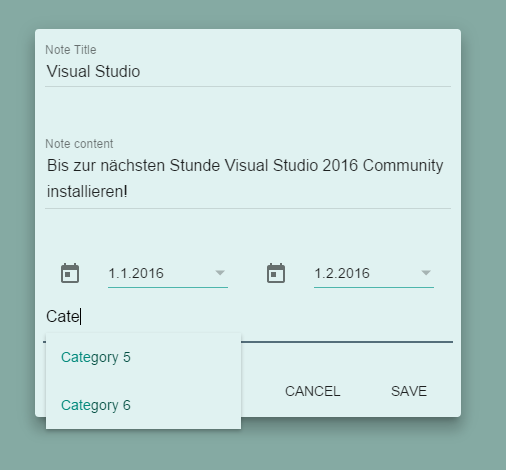
Nach dem einloggen gelangt man zu seiner Notizen Übersicht.



Die Kategorien befinden sich auf der linken Seite. Eine neue Kategorie kann über das Plus-Symbol angelegt werden.



Das Anlegen einer neuen Kategorie geschieht über ein Dialog Fenster. Hier können Name und Farbe der Kategorie festgelegt werden. Die Farbe lässt sich über einen Colorpicker oder direkt per hexadezimalem HTML-Farbcode angeben.



Eine neue Notiz wird ebenfalls per Dialog erstellt. Hier können Titel, Inhalt, Anfangs- und Enddatum und die Kategorien angegeben werden. Es können auch Notizen ohne Kategorien angelegt werden.

1. REST-Befehle

Nachfolgend eine Auflistung aller Befehle die abgesetzt werden können.

**Notiz erstellen:**

/api/v1/user/:userId/note - POST  
Erstellt neue Notiz, der request body beinhaltet die zu speichernde Notiz.

**Notiz abrufen:**  
/api/v1/user/:userId/note/:id - GET  
Gibt die Notiz passend zur ID zurück.

**Alle Notizen abrufen:**  
/api/v1/user/:userId/note/all - GET  
Gibt eine Liste aller Notizen zurück. Optionale Parameter: categories - enthält eine Liste aller Kategorien die in einer Notiz vorhanden sein müssen.

**Notiz updaten:**  
/api/v1/user/:userId/note/:id - PUT   
Updatet eine Notiz, die neue Notiz muss im request body enthalten sein.

**Notiz löschen:**  
/api/v1/user/:userId/note/:id - DELETE  
Löscht eine Notiz anhand ihrer ID.

**Nutzer registrieren:**  
/api/v1/user - POST  
Registriert einen neuen User. Die Daten des Users sind im body enthalten.

**Sitzung erzeugen:**  
/api/v1/session - POST  
Erzeugt eine neue Sitzung. Email und Passwort müssen im request body übergeben werden.

**Sitzung validieren:**  
/api/v1/session - GET  
Validiert eine Sitzung anhand der request Authentifizierung. Der Status der Sitzung wird zurückgegeben.

**Kategorien eines Users abrufen:**  
/api/v1/user/:userId/category - GET  
Gibt eine Liste alle Kategorien dieses Benutzers zurück.

**Kategorien aktualisieren:**  
/api/v1/user/:userId/category - PUT  
Aktualisiert die Liste aller Kategorien für diesen Benutzer.

# Zukünftige Features

1. Registrierung ausbauen

Ein zukünftiges Feature wäre es, die Registrierung auszubauen bzw. auf den allgemeinen Stand der Technik zu heben. Somit würde eine E-Mail an den User gesendet werden, woraufhin dieser einen Link in der E-Mail bestätigen muss. Ebenso fehlt bisher die Funktion „Passwort vergessen“.

1. „Multiplayer“

Bereits am Anfang des Projekts hatten wir die Vorstellung, dass man seine Notizen mit jemandem teilen kann. Dies wäre insofern interessant, als das man bestimmte Kategorien mit anderen Usern teilen kann. So könnte es in einer Wohngemeinschaft die Kategorie „Einkaufen“ geben, bei welcher alle Mitbewohner Schreibrechte besitzen.

Eine andere Möglichkeit wäre es ein eigenes Brett mit extra Kategorien und Notizen zu erstellen, welches dann geteilt werden kann.

Beide genannten Möglichkeiten sind allerdings softwareergonomisch nicht einwandfrei, es müsste unbedingt auf eine klare Trennung von privaten und geteilten Notizen geachtet werden.

1. Archivierung von Notizen

Wir hatten uns vorgenommen eine Archivierung der Notizen zu implementieren - im Moment ist es nur möglich eine Notiz zu löschen.

(Siehe Notiz Mockup: Es gibt ein „Haken“ und ein „Papierkorb“ Symbol)

Geplant war es, eine Kategorie „Archiv“ zu erstellen, in der dann archivierte Notizen gelagert werden. Somit müsste die abgehakte Notiz allerdings aus allen anderen Kategorien entfernt werden – und man könnte nicht mehr nachvollziehen in welchen Kategorien die Notiz einmal war. Das Archiv wäre in dieser Form auch nichts anderes als ein riesiger Papierkorb.

Das Feature haben wir aus Zeitgründen hintenangestellt und nicht fertig gestellt.

1. Drag & Drop

Eine weitere Idee war es, dass der Nutzer per Drag & Drop die Notizen individuell anordnen kann. Dies haben wir allerdings nicht weiter verfolgt.

# Quellen

1. Bildquellen

* Alle Screenshots wurden von uns selbst erstellt.

Kein Link

* MEAN-Stack Logo:

<http://adrianmejia.com/blog/2014/10/03/mean-stack-tutorial-mongodb-expressjs-angularjs-nodejs/>

* Atom.io Logo:

<https://www.pubnub.com/blog/2015-01-21-building-realtime-chat-plugin-atom-io-text-editor/>

* GitHub Logo:

<http://orcid.org/sites/default/files/ckfinder/userfiles/images/github-logo.jpg>

* Trello Logo:

<http://www.thebluecollarmarketer.com/wp-content/uploads/2015/07/Trello-Logo.gif>

* Telegram Logo:

<https://telegram.org/img/t_logo.png>

Alle Bilder wurden am 09.02.2016 zwischen 09:30 und 12:30 herausgesucht.