



Benutzerbezogenes online  
Dokumentenverwaltungssystem mit integrierter Such-  
und Sortierfunktion –  
**EasyDocs**

Diplomarbeit der höheren Abteilung für  
Wirtschaftsingenieurwesen mit dem Schwerpunkt  
Betriebsinformatik an der HTL Anichstraße in Innsbruck

Mitglieder:

Thomas Kerber

Verena Gurtner

Sara Hindelang

Betreuungslehrer:

DI. Andreas Holzmann

vorgelegt am

**4.April 2018**

An der HTL Anichstraße



## Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt durch meine eigenhändige Unterschrift, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht. Die vorliegende Arbeit wurde bisher in gleicher oder ähnlicher Form noch nicht als Magister-/ Master-/Diplomarbeit/Dissertation eingereicht.

Thomas Kerber: \_\_\_\_\_

Verena Gurtner: \_\_\_\_\_

Sara Hindelang: \_\_\_\_\_



## Kurzfassung

Im Alltag entstehen viele elektronische Dokumente, insbesondere während der Schulzeit. Über diese kann im Laufe der Zeit leicht der Überblick verloren gehen. Das Ziel der Diplomarbeit war es, eine Webanwendung zu schaffen, welche es ermöglicht Dokumente in einer Datenbank zu speichern und diese mit Hilfe von Volltextsuche wieder zu finden. Des Weiteren soll es dem Benutzer möglich sein auf seine Dokumente von verschiedenen Geräten zuzugreifen und diese für andere Benutzer öffentlich zu stellen. Die hierbei verwendeten Kerntechnologien waren PostgreSQL mit dessen implementierter Volltextsuche, Java - Libraries welche das Extrahieren des Textes und der Metadaten von diversen hochgeladenen Dokumenten auslesen können und eine Weboberfläche, welche es dem Benutzer ermöglicht dynamisch die Datenbank verwalten zu können. Mit Hilfe von Projektmanagement - Techniken wurde der Arbeitsablauf der Diplomarbeit koordiniert. Im Laufe des Projektes hat sich herausgestellt, dass die Volltextsuche eine sehr präzise Möglichkeit für die Durchsuchung und Wiederfindung von Dokumenten bietet und eine zusätzliche, zu Beginn geplante Beschlagwortung redundant und somit nicht weiter hilfreich ist.



## Abstract

In everyday life, many electronic documents are created, especially during school career. Over time this can easily lead to the loss of track of things over time. The aim of the diploma- thesis was to create a web application, which makes it possible to save documents in a database and find them again with the help of full-text search. Furthermore, it should be possible for the user to access his documents from different devices and make them public for other users. The core technologies used were PostgreSQL with its implemented full - text search, Java libraries that can extract the text and the metadata from various uploaded documents, and a web interface that allows the user to dynamically manage the uploads. The workflow of this was coordinated with the help of project management techniques. During the process of the project, it has been found that the full-text search offers a very precise possibility for the search and recovery of documents and an additional, initially planned indexing of words was redundant and thus not very helpful.



## Gendererklärung

Zur besseren Lesbarkeit werden auf dieser Website personenbezogene Bezeichnungen, die sich zugleich auf Frauen und Männer beziehen, generell nur in der im Deutschen üblichen männlichen Form angeführt, also z.B. "Teilnehmer" statt "TeilnehmerInnen" oder "Teilnehmerinnen und Teilnehmer".

Dies soll jedoch keinesfalls eine Geschlechterdiskriminierung oder eine Verletzung des Gleichheitsgrundsatzes zum Ausdruck bringen.



## Inhaltsverzeichnis

Eidesstattliche Erklärung .....	1
Kurzfassung .....	2
Abstract .....	3
Gendererklärung.....	4
Inhaltsverzeichnis.....	5
Vorwort / Danksagung .....	10
Aufgabenbeschreibung Thomas - Frontend: .....	11
Unterstützte Browser .....	11
Verwendete Technologien und Libraries .....	11
HTML.....	11
CSS.....	13
JAVASCRIPT .....	14
Kommunikation zwischen Webbrowser und Server .....	14
AJAX .....	17
Vorteile von JSON in JavaScript .....	18
JQuery.....	18
JQuery DataTable .....	19
Möglichkeiten:.....	19
DropzoneJS .....	20
Möglichkeiten.....	20
Bootstrap.....	21
Bootstrap Modal Plug-In .....	22
Bootstrap Validator Plug-In.....	22
Cookie Nachricht mit Cookie Consent.....	23
Allgemeine Designentscheidungen .....	24
Aufbau des Konzepts .....	24
Zielgruppe: .....	24
Anspracherichtung:.....	25
Design für Endgeräte mit einer Bildschirmauflösung-Breite unter 768px .....	25
Verwendung der Navigationsleiste.....	25
Design des Logos: .....	28
Kreativitätstechnik 6-3-5 .....	28



Elemente der Website .....	30
Startseite .....	30
Konzept .....	30
Aufgabe einer Startseite - Botschaft: .....	30
Ausarbeitung des relevanten Angebotes für den Kunden .....	31
Beschreibung des Angebotes auf der Website: .....	32
Design .....	34
Aufbau der Startseite: .....	34
Gegenüberstellung des Konzeptes mit dem Resultat.....	37
Mobiles Design .....	38
Das Team- Seite .....	38
Konzept .....	38
Aufgabe einer Team- Seite, Botschaft.....	38
Gestaltung des persönlichen Klappentextes.....	39
Design .....	39
Aufbau der Team- Seite .....	39
Registrier- Seite .....	41
Konzept .....	41
Aufgabe einer Registrierseite - Botschaft .....	41
Ablauf des Registrier-Zykluses .....	41
Design .....	42
Aufbau der Registrierseite .....	42
Gegenüberstellung des Konzeptes mit dem Resultat.....	44
.....	44
Mobiles Design .....	44
Technische Umsetzung .....	45
Clientseitige Validierung.....	45
Darstellen des Passworts in Klartext .....	46
Login- Seite .....	47
Konzept .....	47
Aufgabe der Login- Seite .....	47
Ablauf des Login- Zyklus .....	47
Design .....	48



Aufbau der Website.....	48
Gegenüberstellung des Wireframes mit dem Resultat .....	48
Mobiles Design .....	49
Technische Umsetzung .....	49
Dokument - Seiten .....	49
Konzept .....	49
Vorwort – gelöschte Dateien - Seite .....	49
Aufgabe der Dokumentseite .....	49
Ablauf der Dokumentseite.....	50
Design .....	51
Aufbau der Dokumentseite.....	51
Vergleich Konzept mit Resultat .....	51
Mobiles Design .....	53
Technische Umsetzung .....	54
Implementierung der Dropzone .....	54
Implementierung des jQuery DataTables.....	56
Lizenzen.....	61
Ausblick in die Zukunft Thomas: .....	61
Aufgabenbeschreibung Sara - Controller-Teil .....	62
Extraktion der Dokumente Durch Libraries.....	63
PDFs mittels PDFBox.....	63
PoiLibrary für Word Dokumente .....	66
TXT Format: Simple Extraktion ohne Library.....	68
Strategy- Pattern: .....	69
Benutzer Registrierung und Login .....	71
Das Registrieren .....	71
Der Login.....	74
Passwort-Vergessen Feature .....	75
Sessions: .....	79
Servlets .....	81
UploadServlet.....	85
DownloadServlet .....	86
JSP in EasyDocs .....	86



Tomcat.....	88
Ausblick in die Zukunft Sara .....	89
Aufgabenbeschreibung Verena - Datenbankprogrammierung .....	89
Verwendete Technologien und Prinzipien.....	90
Postgres .....	90
Relationales Datenbankmanagementsystem .....	93
Warum Postgres.....	96
pgAdmin .....	96
Normalisierung einer Datenbank .....	96
SQL.....	98
JDBC .....	100
Volltextsuche von Postgres.....	104
JSON.....	109
JSON .....	110
Speicherung der Dokumente durch BLOBs.....	111
Github .....	111
Umsetzung.....	115
Datenbankdesign und Realität .....	115
Klassendesign .....	117
BLOB Umsetzung: .....	118
Volltextsuche Umsetzung .....	120
Weitere Features: .....	121
1.     Sotierung der Daten .....	121
2.     Daten löschen und wiederherstellen.....	121
3.     Daten nach öffentlichen und privaten Daten unterscheiden.....	121
4.     Passwort zurücksetzen .....	121
Testen.....	121
Ausblick in die Zukunft Verena .....	121
Anhang .....	122
Glossar/ Stichwortverzeichnis – Wortdefinition: .....	122
Quellenverzeichnis .....	123
Literatur: .....	124
Tabellenverzeichnis .....	125



Abbildungsverzeichnis .....	125
Arbeitszeiten Thomas Kerber.....	127
Arbeitszeiten Verena Gurtner .....	129
Arbeitszeiten Sara Hindelang .....	131



## Vorwort / Danksagung

Nach dem wir uns intensiv den Kopf darüber zerbrochen haben, was wir in unserer Diplomarbeit machen wollen, haben wir uns mit unseren alltäglichen Problemen auseinandergesetzt. Diese Überlegungen führten uns schlussendlich zu der Idee Schülern eine Möglichkeit zu bieten elektronische Dokumente, wie Laborberichte und Mitschriften online zu speichern und untereinander auszutauschen. Vor allem war uns aber auch wichtig einen Weg zu schaffen diese so schnell und unkompliziert wie möglich wieder zu finden, nachdem sie einmal abgelegt wurden.

Weiteres möchten wir uns ganz besonders bei Herr Professor Holzmann bedanken, der immer ein offenes Ohr für uns hatte und uns mit Rat und Humor zur Seite stand.

Nicht zu vergessen ist auch unser Umfeld, welches uns in dieser herausfordernden Zeit immer unterstützt und aufgeheizt hat.

Schließlich danke ich meinen Freunden während der Schulzeit für fünf schöne und unvergessliche Jahre.



## Aufgabenbeschreibung Thomas - Frontend:

Der Aufgabenbereich von Herrn Kerber ist die Webschnittstelle. Dies umfasst die Planung und Gestaltung einer modernen Weboberfläche als Benutzeroberfläche. Des Weiteren gilt es die Kommunikation zwischen Client und Server zu programmieren. Zum Schluss umfasst er noch die Ausarbeitung von clientseitigen Skripten innerhalb der Website für eine dynamische Oberfläche.

## Unterstützte Browser

Die Website unterstützt die meisten gängigen Browser. Eine genaue Bestandsaufnahme ist in der unteren Liste vorhanden. Diese Browser werden vollständig unterstützt:

Tabelle 1 Browser

Browser	Version
Chrome	65.0.3325.181 und vorherige Version
Edge	41.16299.15.0 und vorherige Version
Firefox	59.0 und vorherige Version
Internet Explorer	9+
Safari	Version 11
Opera	52.0.2871.3

Für andere Versionen und Browser kann eine Kompatibilität möglich sein, aber sie ist nicht gewährleistet.

## Verwendete Technologien und Libraries

### HTML

HTML steht für Hypertext Markup Language. Dies könnte man auf Deutsch vereinfacht als Auszeichnungssprache beschreiben. Sie bildet die Struktur einer Website und kann von Webbrowsers statisch angezeigt werden. HTML dient lediglich zur Strukturierung des Inhaltes und nicht zur Formatierung.

### Verwendete Version

In der Diplomarbeit verwenden wir die aktuellste Version HTML5.(W3C, <https://www.w3.org/TR/2014/REC-html5-20141028/>, 1.1.2018) Diese hat abgesehen von der Aktualität den Vorteil, dass sie eine Drag&Drop API hat sowie SVG – Dateien unterstützt.

SVG – Dateien sind Scalable Vector Graphics (englisch für skalierbare Vektorgrafik). Sie sind eine Möglichkeit wie man Grafiken auf einer Website anzeigen lassen kann. Ihr großer Vorteil ist, dass die Formen in der Grafik im



Gegensatz zu herkömmlichen Bildern nicht aus einzelnen Pixeln bestehen, sondern aus mathematischen Funktionen, welche den Verlauf angeben. Dadurch werden bei einer Vergrößerung des Bildes keine Pixel unnatürlich groß dargestellt, sondern die Funktionen skalieren nach oben mit. Der zweite große Vorteil von SVG – Bildern ist, dass sie wesentlich weniger Speicherplatz brauchen als herkömmliche Formate, die jeden Pixel speichern müssen.

## Grundstruktur

```
<!DOCTYPE html>
<html>
<head>
<meta>
<title>Page Title</title>
<script/>
<link/>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>
```

Zuerst wird angegeben, dass es sich um eine HTML – Datei handelt, damit der Webbrowser auch weiß, wie er sie zu verarbeiten hat.

### head - Element

Im head Element wird der Titel angegeben, welcher der Browser in der Titelleiste darstellen soll. Des Weiteren erfolgt die Angabe der Meta – Daten im head Element und die Einbindung von Skripts und Stylesheets, welche nicht direkt auf der Seite sind.

### Meta – Daten

Mit den Metadaten kann man zum einen zusätzliche Informationen in den http – Kopfdatensatz einfügen. Hierbei ist besonders wichtig das charset, welches die Charaktercodierung bestimmt. Dadurch wird eine korrekte Darstellung der Zeichen in der HTML – Datei im Webbrowser gewährleistet. Auf unserer Website verwenden wir UTF-8 Charaktercodierung, da diese dank Multi-byte Charakter-Encoding den Großteil des Unicodes und definitiv alle für unsere Website relevanten Zeichen problemlos darstellen kann. Durch Multibyte Encoding werden Zeichen mit bis 4 Bytes in UTF-8 dargestellt. Auf unserer Website werden jedoch nur die ersten 2 Bytes verwendet, da der ASCII – Zeichensatz von 0-127 mit 1 Byte übernommen wurde und Umlaute mit 2 Bytes dargestellt werden.

```
<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">
```



Zum anderen kann man seine Seite mit zusätzlichen Daten für Suchmaschinen optimieren.

```
<meta name="description" content="Speichere und finde deine Dokumente schnell und einfach wieder!" />

<meta name="keywords" content="WORD, DOC, DOCX, PDF, DROP, DRAG, TXT, SAVE, CLOUD, MANAGEMENT, FILE, SYSTEM" />
```

## Körperteil

Im body Element des HTML – Dokuments befindet sich die eigentliche Struktur und der Inhalt des Dokuments. Dieser wird zum Teil mit vorgegebenen Elementen strukturiert. Des Weiteren können die Elemente Attribute bekommen, welche sie im CSS editierbar machen. (siehe CSS)

```
<h1>Ich bin ein Standart-HTML Element</h1>

<div class="custom">Ich kann im CSS angepasst werden</div>
```

## CSS

CSS steht für Cascading Style Sheet. Dies könnte man auf Deutsch stufenartiges Gestaltungsdokument nennen. Es dient zur Formatierung der Inhalte im HTML-Dokument.

### **Verwendete Version:**

In der Diplomarbeit wird CSS3 verwendet, da dieses die neueste Version ist und mehrere Features bereitstellt, welche für modernes und flexibles Design benötigt werden.

### **Gestaltungsmöglichkeiten mit CSS**

- Platzierung der Inhalte auf der Seite
- Darstellung von Farben
- Darstellung von Schriftarten
- Darstellung von Animationen
- Dynamische Darstellung von Inhalten abhängig von der Bildschirmauflösung



## JAVASCRIPT

JavaScript ist eine Skriptsprache, welche es ermöglicht Inhalte auf der Webseite dynamisch zu verändern. Dies beinhaltet Darstellung von Multimedia, Animation und Veränderung von Inhalten und Speichern von Daten lokal auf der Seite des Clients. Des Weiteren ist es möglich mit Hilfe von Ajax mit dem Server zu kommunizieren und dessen Inhalte an den Client weiterzugeben und darzustellen ohne die Seite neu zu generieren müssen. Eine genauere Abhandlung von Ajax erfolgt unter *Ajax*. Die Ausführung von JavaScript hängt sehr stark vom jeweiligen Browser ab.

### Verwendete Version

Unsere Website verwendet JavaScript nach dem Standard von ECMAScript 5, (ECMA International, <http://www.ecma-international.org/publications/files/ECMA-ST-ARCH/ECMA-262%205th%20edition%20December%202009.pdf>, 1.1.2018), welche auf allen gängigen Browsern, sowie mobilen Betriebssystemen unterstützt wird. (Refsnes Data, [https://www.w3schools.com/js/js\\_versions.asp](https://www.w3schools.com/js/js_versions.asp), 1.1.2018)

### Einbettung des Scriptes

Laut dem Standard des W3C ist es möglich seinen JavaScript Code an einer beliebigen Stelle des Dokumentes zu platzieren. Der Hauptfaktor, welcher berücksichtigt werden muss, für die Platzierung ist der Einfluss auf die Ladegeschwindigkeit der Seite. Denn ein Webbrower arbeitet beim Aufbauen der Seite das HTML-Dokument von oben nach unten ab. Außerdem sind einige Scripts abhängig von anderen, weswegen diese darunter platziert werden müssen. So ergibt sich als Lösungsansatz, dass relevante Scripts für den Seitenaufbau im Header platziert werden und weniger relevante Scripts ganz unten im Body – Teil. Zusätzlich stellt JQuery noch eine Funktion zur Verfügung, welche mit dem Ablauf eines Scripts beginnt, sobald die Website vollständig geladen ist. Dadurch werden NullPointerExceptions verhindert und der Aufbau der Seite steht im Vordergrund.

### Kommunikation zwischen Webbrower und Server

Das HTTP Protokoll wurde in den 1990er Jahren entwickelt und regelt die Kommunikation zwischen Client und Server. Unterteilt werden kann eine HTTP Nachricht in eine Request und Response Nachricht.



## Request – Nachricht

```
▼ Request Headers    view parsed
POST /DataTableServlet HTTP/1.1
Host: default-environment.8adp2pb2sf.us-west-2.elasticbeanstalk.com
Connection: keep-alive
Content-Length: 2278
Accept: application/json, text/javascript, */*; q=0.01
Origin: http://default-environment.8adp2pb2sf.us-west-2.elasticbeanstalk.com
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/65.0.3325.162 Mobile Safari/537.36
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Referer: http://default-environment.8adp2pb2sf.us-west-2.elasticbeanstalk.com/LoginServlet
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: JSESSIONID=EE2849142F67F487B70476F5DABA9997
```

Abbildung 1 DataTableServlet Request (siehe DataTableServlet)

Der Request –Teil dient dazu eine Anfrage an den Server zu senden. Hierbei müssen mindestens die Serveradresse(URL), die Sendemethode (GET bzw. POST) und das Protokoll angegeben werden.

**URL:** Bezeichnet eine weltweit gültige Internetadresse mit zusätzlicher Angabe des Dokuments.

**Methode:** Insgesamt gibt es 8 Methoden, wobei nur 2 in unserer Webapplikation relevant sind. (Fielding & Reschke, <http://tools.ietf.org/html/7231#section-4>, 1.1.2018) Dies sind die GET und die POST Methode.

### GET:

Bei der GET Methode wird der Body des Requests leer gelassen und alle Daten werden in die URL geschrieben. Dies ist nur für unsensible und sehr kleine Informationen zu empfehlen, da zum einen potenzielle Angreifer die URL abfangen und dadurch ebenfalls an die Informationen kommen könnten und zum anderen sollte die URL – Länge nicht 8 Byte überschreiten, da ansonsten ältere Browser Probleme bereiten könnten.

### POST:

Bei der POST – Methode werden sämtliche Daten in den Body des Requests gespeichert, wodurch zum einen größere Daten geschickt werden können und zum anderen die Daten verschlüsselt sind.



## Optionale Headerfelder:

Als optionale Headerfelder kann man noch zusätzliche Informationen angeben, welche für den Server relevant sein können. In unserem Fall sind dies sogenannte Cookies, welche die Informationen zum eingeloggten Benutzer speichern, um sicher zu gehen, dass der Benutzer auch wirklich angemeldet ist. Des Weiteren wird noch angegeben, welche Informationen als Antwort erwartet werden. Hierdurch soll das Risiko, dass ein potentieller Angreifer eine falsche Antwort schickt, minimiert werden.

## Response – Nachricht

Der Response Teil der Nachricht stellt die Antwort des Servers auf eine Anfrage dar. Die Antwort enthält den Erfolgsstatus der Anfrage. Zusätzlich wird noch das Headerfeld Content-Type mitgegeben, welches die Zeichenkodierung und den Inhaltstyp der Antwort festlegt.

```
▼ Response Headers      view parsed
HTTP/1.1 200 OK
Content-Encoding: gzip
Content-Type: application/json; charset=UTF-8
Date: Sun, 18 Mar 2018 13:46:58 GMT
Server: Apache-Coyote/1.1
Vary: Accept-Encoding
Transfer-Encoding: chunked
Connection: keep-alive
```

Abbildung 2 Response des DataTableServlet (siehe DataTableServlet)

Der Body Teil der Antwort enthält die eigentliche Nachricht.

```
x Headers Preview Response Cookies Timing
1 {"draw": "1", "recordsTotal": 0, "recordsFiltered": 0, "data": []}
2
```

Abbildung 3 Inhalt Response DataTableServlet

## AJAX

Ajax steht für Asynchronous JavaScript and XML. Mit Hilfe von Ajax ist es möglich, eine Datenübertragung zwischen Server und Browser zu starten, ohne eine Seite neu laden zu müssen. Auch wenn Ajax keine eigene Technologie ist, so ist es laut James Garrett ein neuer Ansatz einer Kombination von bestehenden Technologien. (James Garrett, <http://adaptivepath.org/ideas/ajax-new-approach-web-applications/>, 1.1.2018)

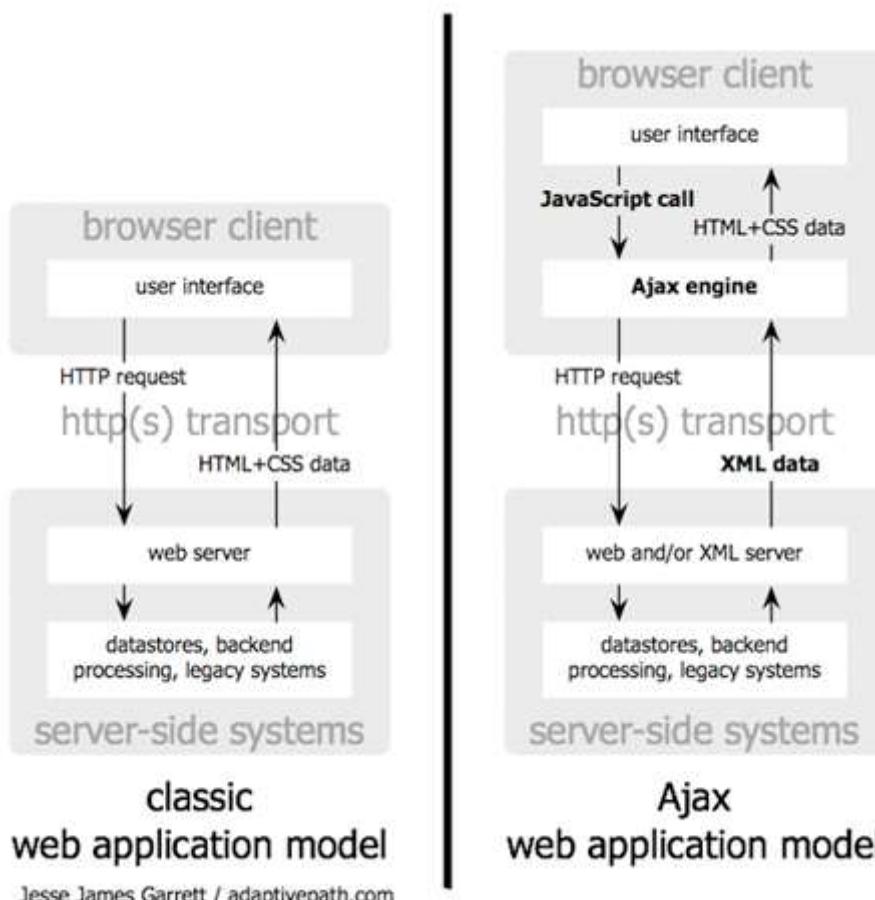


Abbildung 4 Ajax

Die Antwort kann vom Server an Ajax entweder im XML oder JSON Format gesendet werden. Auf unserer Website verwenden wir ausschließlich JSON zur Kommunikation.



## Vorteile von JSON in JavaScript

- JSON ist kürzer und verbraucht dadurch weniger Speicherplatz.
- JSON verwendet Arrays.
- JSON kann von JavaScript direkt in ein JavaScript Objekt umgewandelt werden.
- JSON kann am Server in Java mit einer Library leicht wieder umgewandelt werden. (*siehe JSON*)

## JQuery

jQuery ist eine freie JavaScript – Library, welche die Benützung von JavaScript vereinfacht. Es verkürzt viele JavaScript Befehle, kümmert sich um Kompatibilitätsunterschiede der Browser und stellt neue Befehle zur Manipulation des Inhaltes und Aussehens der Seite, welche es in nativen JavaScript nicht gibt, zur Verfügung. Um jQuery auf einer Webseite zu verwenden, muss man lediglich die JavaScript Bibliothek mittels eines <script> - Tags einbinden. Hierbei kann entweder die Library lokal in der Webapplikation eingebunden werden oder man greift auf eine öffentliche Version des Skriptes(CDN) zu. Auf unserer Website haben wir die CDN von Google eingebettet. Dies hat den Vorteil, dass wenn ein Nutzer auf einer anderen Seite bereits einmal die Library gedownloadet hat, dann muss er sie nicht erneut herunterladen, sondern kann sie aus dem Cache seines Webbrowsers abrufen.

## jQuery Ajax Anfrage

```
$.ajax({  
    method: "POST",  
    url: "MeinServlet",  
    data: {Benutzer: "User1"},  
})  
.done(function( msg ) {  
});
```

Mittels "\$.ajax(...)" wird ein Ajax Objekt erstellt. Das übergebene Objekt spezifiziert die Anfrage, sowie eine Methode zur Verarbeitung des Resultates.



## JQuery DataTable

jQuery DataTables ist ein jQuery Plugin, welches es ermöglicht, eine Tabelle in HTML mit JavaScript zu bearbeiten.

DateiTyp	Name	Uploader	Autor	UploadDatum	DokumentDatum	Download
	AA05-NAT-Kerber.pdf	thomas	Thomas Kerber	16.03.2018	01.12.2017	
	AA15-wifi.docx	neu	c102wm	18.03.2018	22.02.2018	
	Abgabe.pdf	verena	Verena Gurtner	01.03.2018	11.12.2017	
	Abgabe.pdf	maxi	Verena Gurtner	01.03.2018	11.12.2017	
	Cooler text.pdf	verena	Verena Gurtner	01.03.2018	30.01.2018	
	Das Skelett des Menschen und seine Knochen.pdf	verena	Verena Gurtner	01.03.2018	27.07.2017	
	Klassische Krautfleckerl.pdf	verena	Verena Gurtner	01.03.2018	11.12.2017	
	Referat_Leitfaden.docx	neu	Schueler User	18.03.2018	01.02.2017	
	Thunfisch Nudeln.pdf	verena	Verena Gurtner	02.03.2018	11.12.2017	
	Thunfisch Nudeln.pdf	holzmann	Verena Gurtner	02.03.2018	11.12.2017	

1 bis 10 von 11 Einträgen

12

Abbildung 5 DataTable

Möglichkeiten:

- 1) Seitennummerierung: Es ist möglich die Anzahl an Einträgen pro Seite zu definieren und wenn diese überschritten wird, den Wechsel zwischen mehreren Seiten anzubieten.
- 2) Suchfunktion: Standardmäßig ist eine Durchsuchung der Texte nach Titel über den Client möglich. Dies haben wir jedoch überschrieben und stattdessen serverseitig eine Kombination aus Volltextsuche und der bestehenden Suche eingebaut.
- 3) Spaltensortierung: Mit einem Klick auf die Spaltenüberschrift kann man die Tabelle bezüglich dieser Spalte alphabetisch bzw. numerisch ordnen.
- 4) Dynamisches Design: Die Größe der Tabelle passt sich an die Bildschirmauflösung an.



- 5) Sprachunterstützung: Das Plug-In ist sprachlich komplett anpassbar. Auch die einzelnen Funktionen sind überschreibbar und erweiterbar beziehungsweise deaktivierbar.
- 6) Open Source: Der Code ist frei lesbar und veränderbar nach der MIT Lizenz. (*siehe Lizenz*)
- 7) Rendern: Abhängig von den Daten ist es möglich eigenen Symbole zur Anzeige zu generieren. Dies erfolgt im oben gezeigten Beispiel bei den Symbolen für den DateiTyp, sowie bei den Download- Buttons.

## DropzoneJS

DropzoneJS ist eine Open Source JavaScript Library (Matthias Meno, <http://www.dropzonejs.com/>, 1.1.2018), welche eine einfache Konfiguration von Drag & Drop ermöglicht. Lizenziert ist es unter der MIT- Lizenz (*siehe Lizenz*).

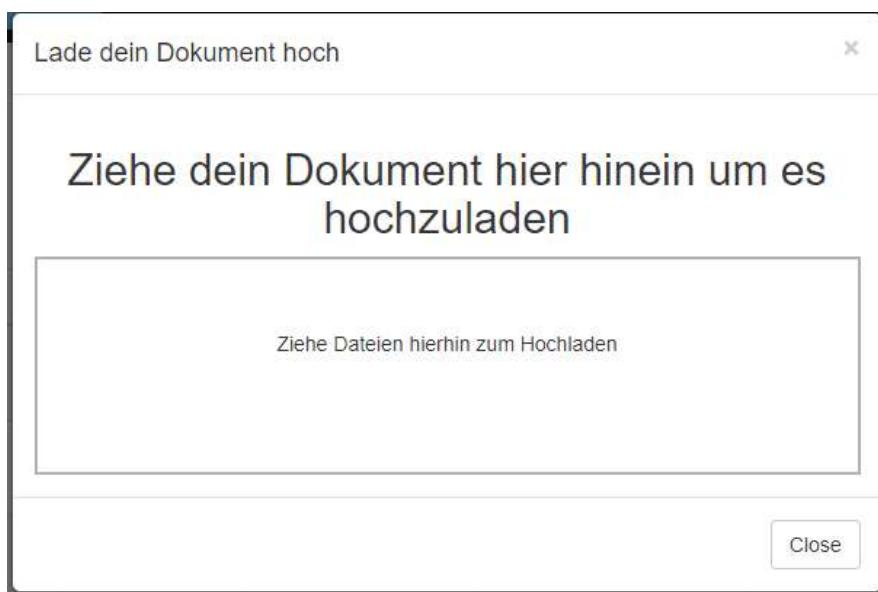


Abbildung 6  
Dropzone

## Möglichkeiten

- 1) Clientseitige Validierung: DropzoneJS ermöglicht es möglich bereits im Browser die Größe und den Dateityp zu überprüfen und falls gewünscht abzuweisen.
- 2) Sprachunterstützung: Die Library ist sprachlich komplett anpassbar. Auch die einzelnen Funktionen sind überschreibbar, erweiterbar beziehungsweise deaktivierbar.
- 3) Ereignisverarbeitung: Die Library sendet für fast jede Aktion wie Start, Verlauf und Abbruch des Uploads ein Event, welches von anderen JavaScript Funktionen erkannt und verwendet werden kann.



- 4) Abwärtskompatibilität: DropzoneJS unterstützt Chrome 7+, Firefix 4+, IE 10+, Opera 12+ und Safari 6+. Bei älteren Browsern ersetzt es seine Dropzone mit einem Input Element.

## Bootstrap

Bootstrap ist ein Open-Source<sup>1</sup> Frontend – CSS, HTML, JS – Framework. Es verwendet ein Grid – System, welches einen einfachen Aufbau einer Webseite ermöglicht. Die Einbettung des CSS – Stylesheets, sowie der JS – Library erfolgt wieder mit einer CDN von Google.

### Verwendete Version

Auf unserer Webseite wird Bootstrap 3 verwendet (Bootstrap, <http://getbootstrap.com/docs/3.3/>, 1.1.2018). Dies hat den Grund, dass ich mit der Version 3 am meisten Erfahrung habe und die Dokumentation meiner Meinung nach am besten ist. Im Vergleich zu Bootstrap 4 sind die Neuerungen für unser Projekt nicht relevant.

### Vorgefertigte CSS – Klassen

Bootstrap bietet für die meisten Elemente in HTML ein eigenes CSS – Design, welche die Optik wesentlich aufwertet und einem die Arbeit abnimmt, diese selbst zu gestalten. Ein weiterer wesentlicher Vorteil dieser vorgefertigten Klassen ist, dass der Benutzer sie wahrscheinlich bereits auf einer anderen Website, auf Grund der Popularität von Bootstrap, gesehen hat. Dadurch fühlt sich der Benutzer automatisch auf unserer Website wegen ihm unterbewusst bekannter Elemente wohl.

### Grid – Layout

Zur Gestaltung von Webseiten stellt Bootstrap ein Gitterlayout zur Verfügung. Dieses besteht aus beliebig vielen Reihen und 12 Spalten. Gestaltet wird dieses System mit Hilfe von CSS – Klassen. In einen <div> Element mit der Klasse „row“ für Reihe werden mehrere <div> Elemente mit den Klassen für die entsprechende Kolumnenbreite angegeben. Hierbei gilt zu beachten, dass insgesamt immer genau 12 Kolumnen in einer Reihe sein müssen. Da man oft andere Layouts abhängig von der Bildschirmgröße haben möchte, bietet Bootstrap die Möglichkeit Spaltenklassen zu parametrisieren:

```
<div class="row">
  <div class="col-xs-4 col-lg-0"><768px|33% , >1200px | 0%</div>
  <div class="col-xs-4 col-lg-12"> 768px|33% , >1200px | 100%</div>
  <div class="col-xs-4 col-lg-0"> 768px | 33% , >1200px | 0% </div>
</div>
```



In diesem Fall wäre der Bildschirm auf Geräten mit einem Bildschirm unter 768 Pixel in 3 Inhalte geteilt und bei einem Bildschirm mit mehr als 1200 Pixel in 1 Inhalt.

## Bootstrap Modal Plug-In

Bootstrap Modal ist ein Plug-In, welches zusätzlich als JavaScript implementiert werden muss.

Ein modaler Dialog ist ein Fenster, welches auf der Seite erscheint. Hierbei wird der Rest der Seite verdunkelt und das Fenster steht klar im Vordergrund.

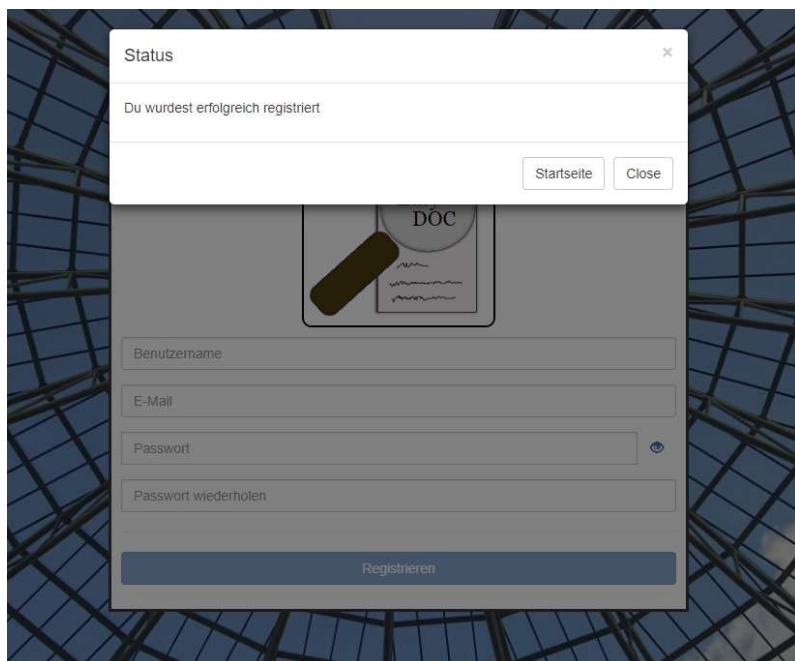


Abbildung 7 Modal

## Vorteile des modalen Dialogs gegenüber eines Alerts

- Ein Alert ist eine native JavaScript – Methode, welche jeder Browser abhängig von seiner JavaScript Engine anders darstellt. Mit einem Modal wird sichergestellt, dass die Kommunikation browserunabhängig ist.
- Das Modal ist mit CSS anpassbar und dadurch optisch wesentlich ansprechender.
- Das Modal ist mit HTML – Elementen besetzbar, welche zum Beispiel eine Navigation auf eine andere Seite anbieten können.

## Bootstrap Validator Plug-In

Bootstrap Validator ist ein JavaScript Plugin. (Calpico Papi, <http://1000hz.github.io/bootstrap-validator/>, 1.1.2018)



Es bietet eine Echtzeitvalidierung von Formularen an. Hierbei kann dann ein Fehlerhinweis ausgegeben werden und die Absendefunktion gesperrt werden. Zusätzlich zu bereits standardmäßig implementierten Überprüfungen wie Mindestlänge, können auch noch eigene Überprüfungen geschrieben werden.

The screenshot shows a form with two password input fields. The first field contains "passwort1" and the second field contains "pafseasdf". Below the fields, the text "Passwörter stimmen nicht überein" is displayed in red, indicating a validation error.

Abbildung 8 Bootstrap Validator

### Cookie Nachricht mit Cookie Consent

Cookie Consent ist eine freie JavaScript Library, welche eine Nachricht an den Benutzer schickt, wenn er die Seite aufruft. (Silktide Ltd., <https://cookieconsent.insites.com/>, 1.1.2018) Laut der EU – Richtlinie RL 2009/136/EG vom 25. November 2009 ist eine Website verpflichtet seine Nutzer über die Nutzung von Cookies zu informieren. Unsere Website verwendet für eine Vielzahl von Diensten Cookies (*siehe Session*). Umgesetzt habe ich diese Nachricht mit einer freien JavaScript Library von Cookie Consent. Diese Library hat den Vorteil, dass sie sprachlich konfigurierbar ist und einen Link zu ihrer Website bietet, welche eine vollständige Information zu Cookies bietet. Implementiert wird der untenstehende Code auf jeder Website, da die Bestätigung der Nachricht ebenfalls als Cookie gespeichert wird und nur einmal angezeigt wird.

```
<link rel="stylesheet" type="text/css"
      href="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.0.3/cookieconsent.min.css"
      />

<script
      src="//cdnjs.cloudflare.com/ajax/libs/cookieconsent2/3.0.3/cookieconsent.min.js"><
      /script>

<script>

window.addEventListener("load", function(){

window.cookieconsent.initialise({


    "palette": {

        "popup": {

            "background": "#000"
        }
    }
})});
</script>
```



```
},
"button": {
  "background": "#f1d600"
},
},
"content": {
  "message": "Unsere Website verwendet Cookies um Ihnen die bestmögliche Nutzererfahrung zu garantieren.",
  "dismiss": "Verstanden!",
  "link": "Mehr Informationen"
}
});
</script>
```

## Allgemeine Designentscheidungen

### Aufbau des Konzepts

Der Aufbau des Konzepts erfolgt nach dem Prozess der Werbeplanung nach Marketing für Techniker Abschnitt Werbestrategie. (Rupert Erhart, Marketing für Techniker) Wobei in Bezug auf die Website auf die Ausarbeitung eines Werbeetats verzichtet wird. In den allgemeinen Designentscheidungen wird auf die ersten beiden Punkte: Zielgruppe und Anspracherichtung eingegangen, da diese über die ganze Website gleichbleiben. Der Punkt Botschaft wird dann bei jeder Webseite gesondert beschrieben.

### Zielgruppe:

Hierbei gehen wir davon aus, dass Benutzer unsere Seite besuchen, welche auf der Suche nach einem Online - Dateiablagensystem sind. Unser Fokus in Bezug auf die Zielgruppe richtet sich vor allem an Studenten und Schüler, da unser Projekt ursprünglich aus unserem Bedürfnis nach einer Online - Dateiablage entstanden ist. Selbstverständlich sind auch alle anderen Berufssparten auf unserer Website willkommen.



## Anspracherichtung:

Unter der Anspracherichtung versteht man laut Jochen Dohmen die „Rahmenvorgaben für die Gestaltung konkreter Werbemittel in Bild und Text“.  
(Buch Rationales Werben, Werbemedien und Werbeplanung, S. 21). Zur Entscheidung für die Anspracherichtung habe ich nach 2 Kriterien entschieden.

### 1. Welche Anrede passt für unsere Zielgruppe?

Da unsere Zielgruppe zu einem großen Teil aus Schülern und Studenten besteht, jedoch auch Nutzer aus dem Berufsleben beinhaltet, ist sowohl die Anrede „du“ als auch „Sie“ angebracht.

### 2. Welche Atmosphäre will ich auf der Website aufbauen?

Unsere Website soll jung und dynamisch wirken. Der Benutzer soll den Eindruck haben, dass die Website einfach und unkompliziert ist. Deswegen ist das „Du“ besser geeignet, um ihm diesen Eindruck zu vermitteln.

Aus diesen Gründen wird auf der Website ein groß geschriebenes „Du“ verwendet, welches seit der neuen Rechtschreibordnung erlaubt ist und einen Mittelweg zwischen Höflichkeit und Unkompliziertheit bildet. (Quelle Duden, Sprachwissen) Ein Beispiel für eine solche Praxis ist die Website von Spotify (Stand März 2018).

## Design für Endgeräte mit einer Bildschirmauflösung-Breite unter 768px

Solche Geräte sind aktuell in erster Linie Smartphones. Auf Grund ihrer relativ kleinen Bildschirme sind einige Anpassungen des Designs zu treffen.

- Strukturen müssen stark vergrößert werden, damit sie weiterhin erkennbar bleiben.
- Schriftgrößen müssen sich dynamisch anpassen.
- Die Navigationsbar muss von einer horizontalen Ausrichtung zu einer vertikalen Ausrichtung wechseln.
- Inhalte dürfen vertikal nicht über den Seitenrand hinausgehen.

## Verwendung der Navigationsleiste

Da sich die Navigationsleiste auf allen Webseiten befindet wird sie hier nur einmal besprochen.

Sinn der Navigationsleiste ist es, dem Benutzer eine einfache Navigation auf der Website zu ermöglichen.



## Vorteile:

- Vertrautes Element: Da eine Navigationsbar weit verbreitet ist, wissen die meisten Benutzer wie sie zu verwenden ist.
- Konstantheit: Durch die Positionierung auf allen Hauptfunktionsseiten hat der Benutzer stets ein konstantes Element zur Orientierung.

## Aufbau der Navigationsbar

Das 1. Element von links ist immer das Logo, wodurch es sich auf Dauer beim User einprägt und eine Markenzugehörigkeit (brand affiliation) entsteht. Als nächstes folgen dann abhängig von der Webseite die jeweiligen Links zu den anderen Webseiten.

Navigationsbar bei angemeldetem Benutzer:

	Seite 1	Seite 2		Abmelden
--	---------	---------	--	----------

Abbildung 9 Navigationsbar angemeldet

Navigationsbar bei nicht angemeldetem Benutzer

	Seite 1	Seite 2	
--	---------	---------	--

Abbildung 10 Navigationsbar nicht angemeldeter Benutzer

## Auswahl der Symbole auf der Navigationsbar

Zur besseren Orientierung und um eine angenehmere User Experience zu schaffen wurden einige Elemente mit Icons zusätzlich zur Beschriftung versehen. Die Auswahl der Icons erfolgt nach ihrer Bekanntheit. Hierbei wird auf Grund der Bekanntheit die Bibliothek von „Font Awesome“ verwendet. ( Fonticons Inc, <https://fontawesome.com/>, 1.1.2018)

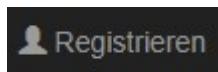


Abbildung 11 Registrierfeld



Abbildung 12 Anmeldefeld

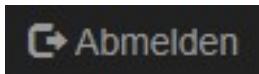


Abbildung 13 Abmeldefeld



Abbildung 14 Private/ Publicfeld



Abbildung 15 Symbol für PDF - Dokumente



Abbildung 16 Symbol für Microsoft Word Dokument



Abbildung 17 Symbol für Textdokument

## Mobiler Aufbau

Beim mobilen Aufbau wird die Navigationsbar bis auf das Logo reduziert und bei einem Klick auf den Knopf fährt sie ihre Elemente nach unten aus.



Abbildung 18 Mobile Ansicht 1

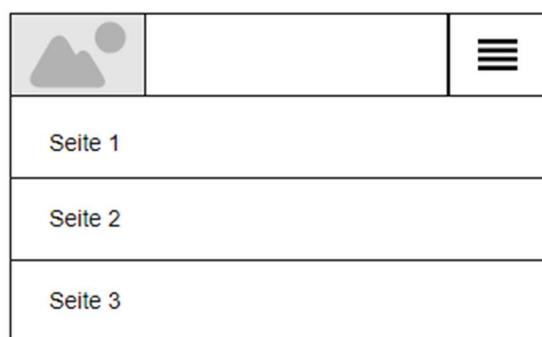


Abbildung 19 Mobile Ansicht 2



## Design des Logos:

Zum Ermitteln des Logos haben wir ein Treffen abgehalten und mit Hilfe von Kreativitätstechniken Ideen gesammelt. Hierbei haben wir die 6-3-5 Methode verwendet, um mehrere Ideen zu sammeln. Diese Ideen haben wir dann in der Gruppe diskutiert und uns dann für ein Logo entschieden.

## Kreativitätstechnik 6-3-5

### *Anwendung:*

Die 6-3-5 Methode wird verwendet, um in kurzer Zeit möglichst viele Ideen zu sammeln. Der Grundgedanke besteht darin, dass Teilnehmer Ideen von anderen aufgreifen und weiterentwickeln beziehungsweise Variationen entwickeln.

### **Vorteile:**

- sofortiges Feedback
- sehr viele Ideen in verhältnismäßig kurzer Zeit
- Ideen werden erst am Ende besprochen → niemand wird in der Ideenfindung durch andere beeinflusst
- Weiterentwicklung von Ideen

### **Nachteile:**

- meist viele redundante Ideen
- Zeitdruck kann störend wirken

### *Ablauf:*

#### **Allgemein:**

Der Moderator erklärt allen Mitgliedern die Regeln der 6-3-5 Technik. Dann wird kurz die Problemstellung besprochen und mögliche Fragen werden geklärt. Als nächstes erhält jeder Teilnehmer einen Zettel. Auf diesen soll er in 5 Minuten alle Lösungen aufschreiben, welche ihm einfallen. Nach 5 Minuten tauscht der Moderator dann die Zettel unter den Teilnehmern aus. Dies wird solange fortgeführt, bis jeder Teilnehmer jeden Zettel erhalten hat.

#### **Spezifisch:**

Die Problemstellung war das Design eines Logos. Hierbei war uns wichtig, dass das Logo nicht zu überladen mit Inhalten und gut skalierbar ist. Dies hat den Grund, dass es sowohl in kleinem Format als Icon verwendet werden soll, als auch in größeren Formaten auf der Website.



*Ergebnis der Kreativitätstechnik:*



Abbildung 20 Logo



## Elemente der Website

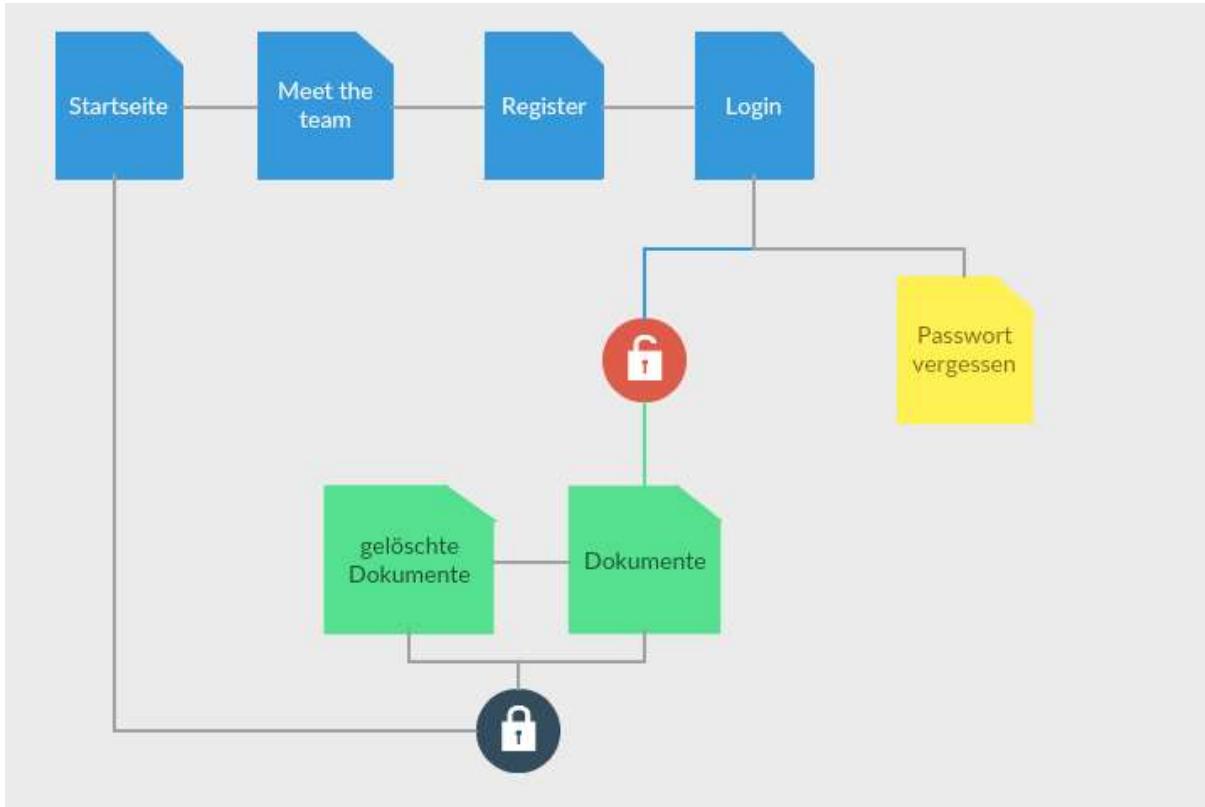


Abbildung 21 Sitemap

Die obige Sitemap stellt die Struktur der Website dar. Zwischen den Seiten, welche in der obersten Hierarchie – Stufe dargestellt werden, kann frei gewechselt werden. Von der Login- Seite aus hat man entweder die Möglichkeit auf seine persönlichen Dokumente zu gehen oder auf eine Unterseite, welche eine Möglichkeit zum Zurücksetzen des Passwortes anbietet.

## Startseite

### Konzept

#### Aufgabe einer Startseite - Botschaft:

Die Startseite ist der erste Kontaktpunkt mit den potenziellen Kunden. Sie sollte den Kunden neugierig machen, kurz erklären was die Seite kann und ihn zu einem Beitritt auf unsere Seite überzeugen.



## Ausarbeitung des relevanten Angebotes für den Kunden

Tabelle 2 Nutzen

Funktion	Grundnutzen	Zusatznutzen	Psychologischer Effekt
1. Dokumente werden in einer Online-Datenbank hinaufgeladen und heruntergeladen	Dokumente können über einen Webbrower hochgeladen und heruntergeladen werden. Dokumente werden nicht lokal gespeichert.	Einfacher Upload und Download von Dokumenten.  Sicherheit der Daten	Eine intelligente, moderne und smarte Lösung verwenden.
2. Dokumente können mit Volltextsuche durchsucht werden.	Schnelle Navigation und Finden von Dokumenten	Zeitersparnis im Gegensatz zur herkömmlichen Suche	Eine intelligente, moderne und smarte Lösung verwenden.
3. Dokumente können öffentlich oder privat gestellt werden.	Zugriff auf fremde Dokumente und die Möglichkeit eigene Dokumente öffentlich zu stellen.	Vernetzung mit anderen Benutzern.  Simples Teilen (Share) von Dokumenten mit Freunden.  Verbreitung von Dokumenten	Teil einer Community/Gemeinschaft.

Ziel ist es die Funktionen der Website als Nutzen für den Kunden darzustellen. Hierbei muss zwischen den 3 Arten von Nutzen, welcher ein Kunde wahrnimmt, unterschieden werden. (*Grundlagen Marketing Werbung S. 8 von Bernhard Ulrich*)



## 1. Grundnutzen –

ist die reine Funktion, welche verwendet werden kann

## 2. Zusatznutzen –

sind darüberhinausgehende Funktionen, welche der Kunde erlebt

## 3. Psychologischer Nutzen –

ist der psychologische Effekt, welchen der Kunde erlebt

Beschreibung des Angebotes auf der Website:

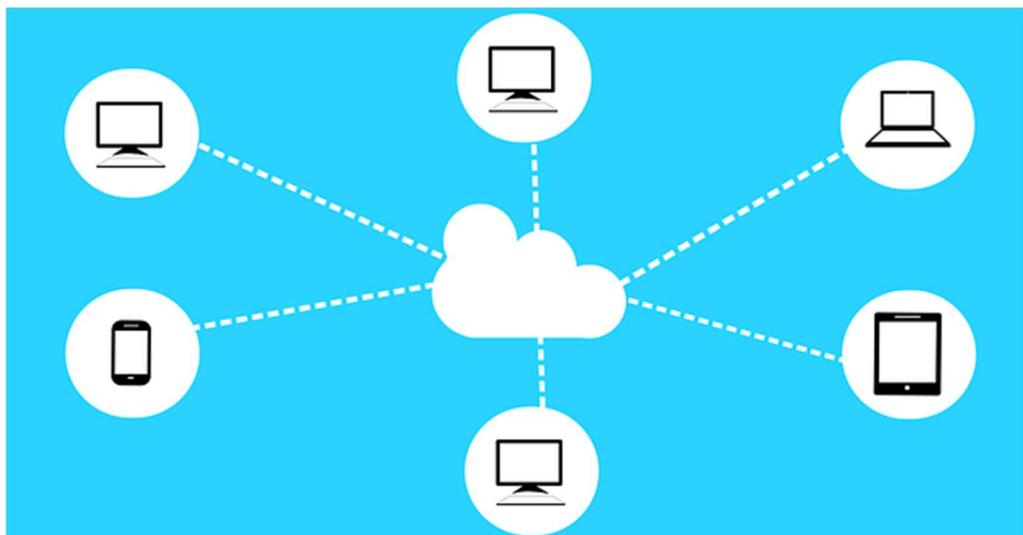
*1. Funktion - Zugriff von überall*

Abbildung 22 Zugriff von Überall

**Bild:** Ziel des Bildes ist es Globalität darzustellen und eine Verbindung mit dem Internet. Das Bild der Cloud im Internet ist ein Sinnbild für Onlinespeicher. Des Weiteren werden Verbindungen von vielen Geräten, sowohl Mobilgeräten, als auch Standgeräten, zu der Cloud in der Mitte gezogen, um darzustellen, dass von überall ein Zugriff möglich ist.

**Unterüberschrift:** Zugriff von überall

**Text:** Mit EasyDoc hast Du einen einheitlichen, kostenfreien Speicherort für alle deine Dokumente. Nie wieder Verlust von Daten auf Grund eines PC-Ausfallen. Die Tage, an denen Du deine Dokumente mühsam von einem Gerät zum Nächsten schicken musstest, sind gezählt.



## 2. Funktion - Volltextalgorithmus

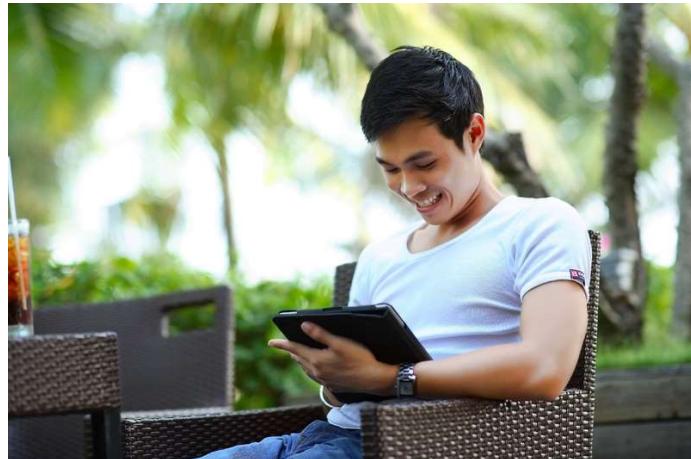


Abbildung 23 Work smart, not hard

**Bild:** Ziel ist es dem Benutzer zu zeigen, dass er mit dieser neuartigen Technik seine Dokumente schneller und durch technologische Mittel einfacher finden kann. Deswegen wird auf dem Bild eine junge fröhliche Person gezeigt, welche entspannt und glücklich ihren Laptop bedient.

**Unterüberschrift:** Work smart, not hard

**Text:**

Vorbei sind die Zeiten in welchen Du nach deinen Aufsätzen, Arbeiten, Briefen, etc. mühsam suchen musstest. Mit unserer innovativen Volltextsuche ist jedes Dokument mit nur einigen Stichwörtern aus dem Text gefunden. Probiere es einfach selbst!

## 3. Funktion - Community



Abbildung 24 Community



**Bild:** Ziel ist es dem Benutzer zu vermitteln, dass er mit unserer Website Teil einer Gemeinschaft ist, welche sich gegenseitig unterstützt und in welcher er auf einfache Art und Weise seine Dokumente teilen und andere Dokumente erhalten kann. Um dies darzustellen wird ein Bild mit mehreren Personen gezeigt werden, welche sinnbildlich für die gemeinsame Arbeit zusammensitzen und auf mehreren Endgeräten arbeiten.

**Unterüberschrift:** Gemeinsam sind wir mehr

**Text:** Teile und empfange Texte aus der ganzen Welt. Mit nur einem Klick kannst Du deine Dokumente für alle zur Verfügung stellen. Lasse dich von fremden Arbeiten inspirieren oder finde endlich wieder diesen einen Aufsatz, von dem du nichts außer dieser einen Überschrift weißt. Tritt jetzt unserer Community bei und erreiche mehr.

## Design

Mit den gesammelten Anforderungen gilt es nun eine ansprechende und moderne Website zu bauen. Hierzu wurde zuerst ein konzeptioneller Entwurf erstellt, welches dann als Vorlage für die tatsächliche Webseite verwendet wurde.

Aufbau der Startseite:

### Teil - Navigationsbar

In der Navigationsbar sind alle Bereiche, welche von der Startseite aus angesteuert werden sollen. Dies sind:

- Anmelden
- Registrieren
- About Us

### Teil – Hero Image



Abbildung 25 Navigationsbar



Das Hero Image ist ein großes Bild, welches im Hintergrund ganz oben auf der Webstartseite platziert wird. Es hat mehrere positive Effekte:

- **Visueller Content:** Durch ein großes Bild wird sofort die Aufmerksamkeit des Besuchers eingefangen und dann automatisch weitergeleitet zu dem Text, welcher im Zentrum des Bildes steht.
- Bilder unterstützen Aussagen, selbst wenn sie nicht einmal direkt mit der Aussage zu tun haben.
- Schaffen einer angenehmen Atmosphäre durch ein ansprechendes und ruhiges Bild.

Unterstützt wird das Hero - Image von einer Überschrift mit einem Slogan darunter, welcher die Neugierde des Besuchers wecken und ihn zum 2. Teil der Website weiterführen soll.

### Bild für das Hero - Image:

#### Kriterien:

- **Kontrast:** Das Bild sollte nicht zu aufgewühlt sein, damit der darüber liegende Text immer noch gut lesbar ist. Außerdem sollte sich die Schrift des Textes gut vom Bild abheben
- **Unterstützend:** Der Inhalt des Bildes soll den Besucher nicht vom Text ablenken, sondern den Text unterstützen.
- **Positive Atmosphäre schaffen:** Der Besucher soll sich durch das Bild auf der Website willkommen fühlen.

Ausgewähltes Hero - Image:

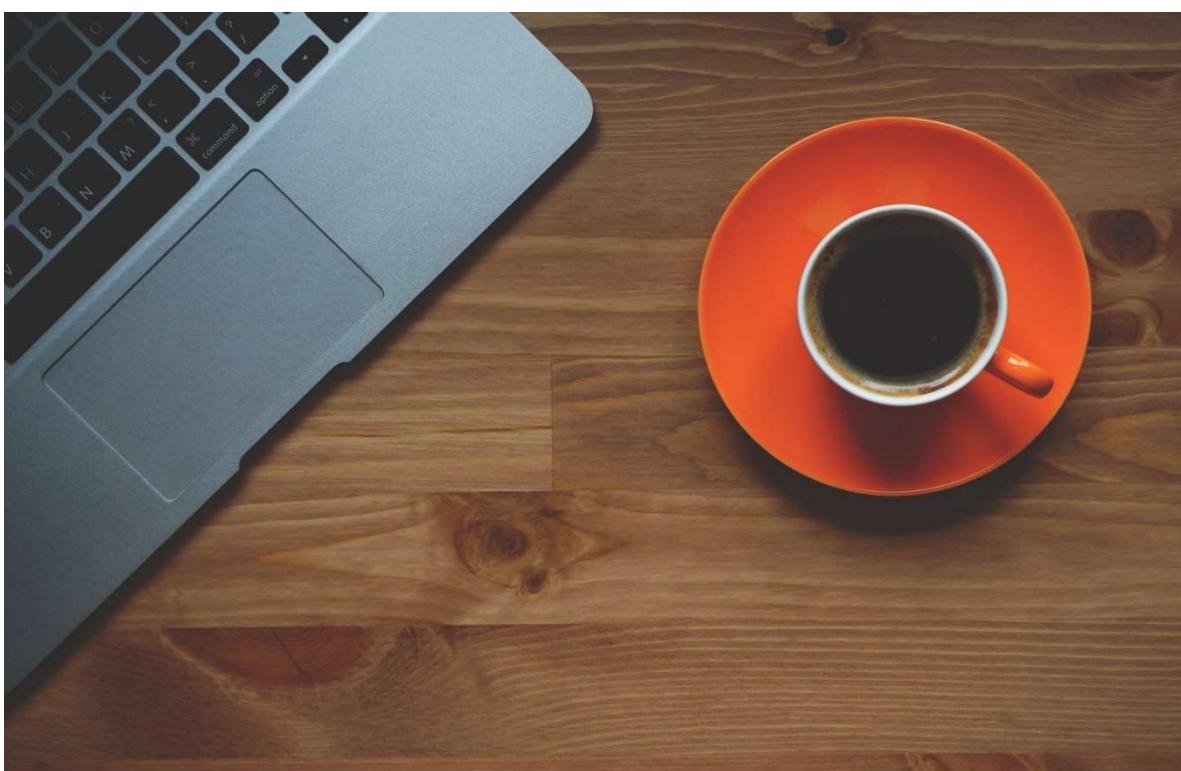


Abbildung 26 Heroimage Startseite



Nachdem das anfängliche Interesse des Kunden geweckt wurde, ist der nächste Schritt ihm im Detail die Funktionen und Vorteile der Website darzulegen. Hierbei wird eine Kombination aus Bildern und Text verwendet. Navigiert wird durch den Inhalt mit einem Scroll – Design. Durch das Scrollen nach unten wird der Besucher Schritt für Schritt in die einzelnen Funktionen eingeführt.

### Parallax - Effekt

#### Erklärung:

Um dem Besucher den Eindruck von Tiefe zu simulieren wird der Parallax - Effekt angewandt. Bei diesem Effekt wird das Hero - Image beim Herunterscrollen von dem kommenden Content schneller überlagert. Zurückführen kann man dies auf den physikalischen Effekt der Bewegungsparallaxe. Dieser Effekt sagt aus, dass wenn ein Beobachter sich parallel zu zwei sich gleich schnell bewegenden Objekten bewegt, dann erscheint ihm das näher gelegene Objekt schneller.

#### Verwendungszweck:

Dies ist ein Trend im Webdesign, welcher die Dynamik und Modernität noch einmal hervorheben soll. Des Weiteren soll durch diesen spielerischen Übergang der Besucher dazu animiert werden nach unten zu scrollen und in der Folge die restliche Webseite zu entdecken.

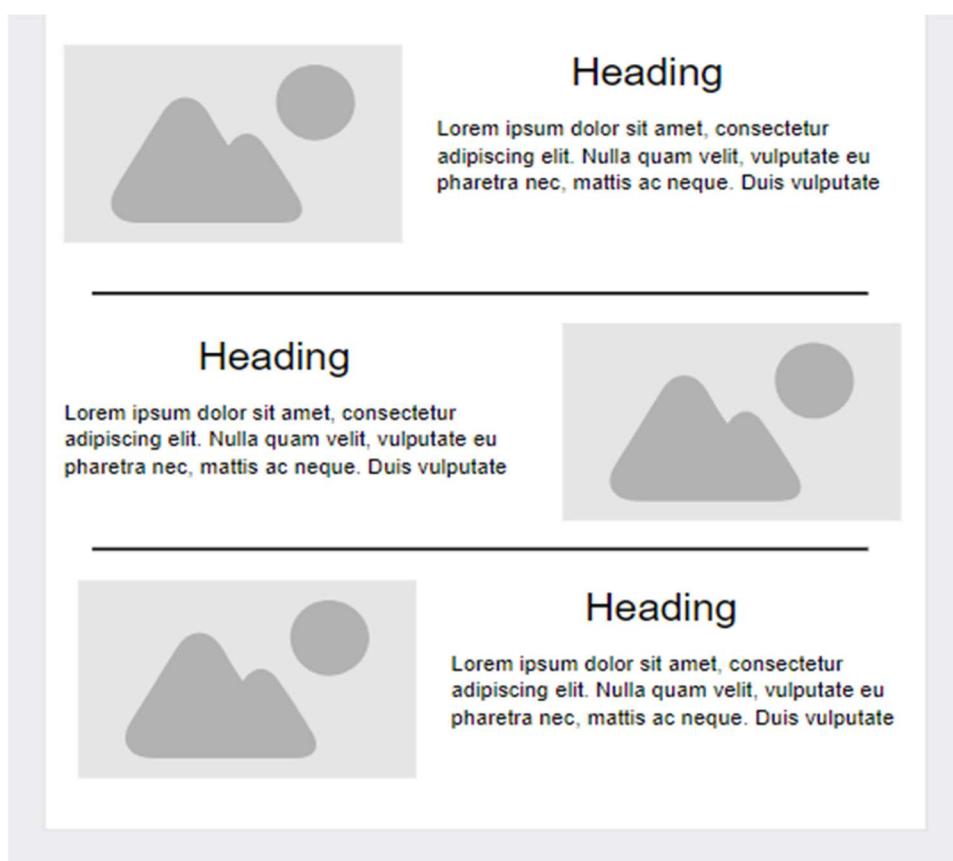


Abbildung 27 Parallaxeffekt



## Gegenüberstellung des Konzeptes mit dem Resultat

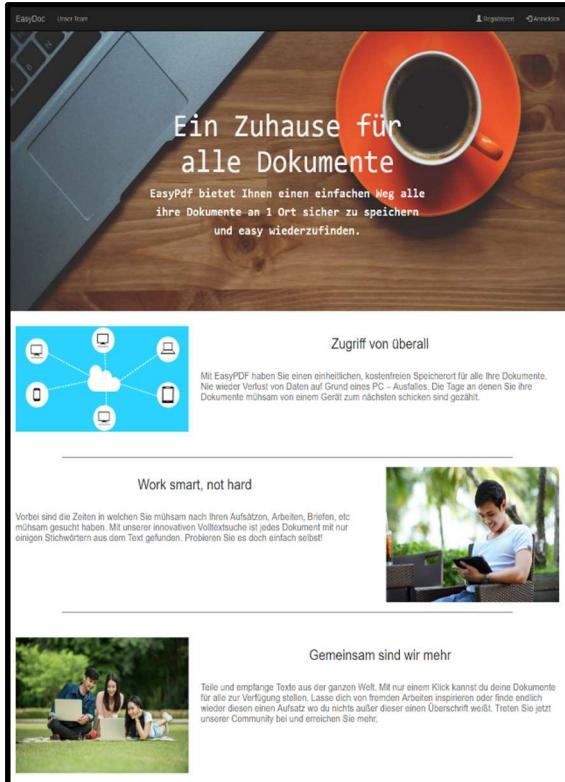


Abbildung 28 Wireframe IST

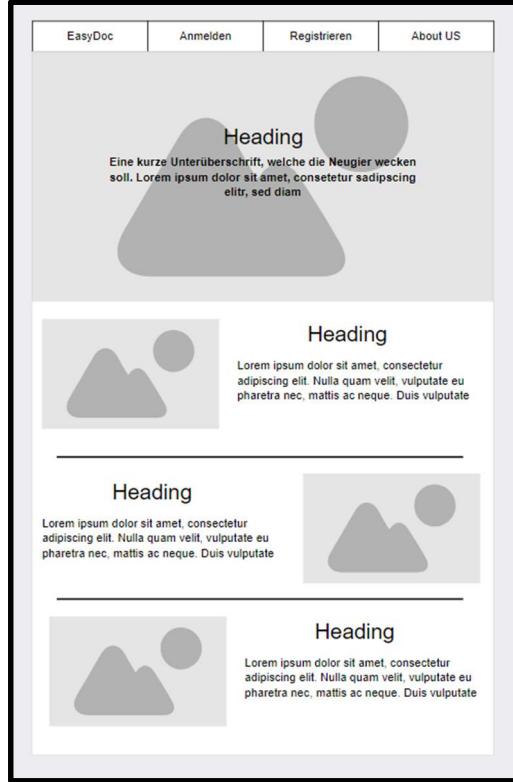


Abbildung 29 Wireframe Plan

Der untenstehende Content schiebt sich über das Hero - Image.

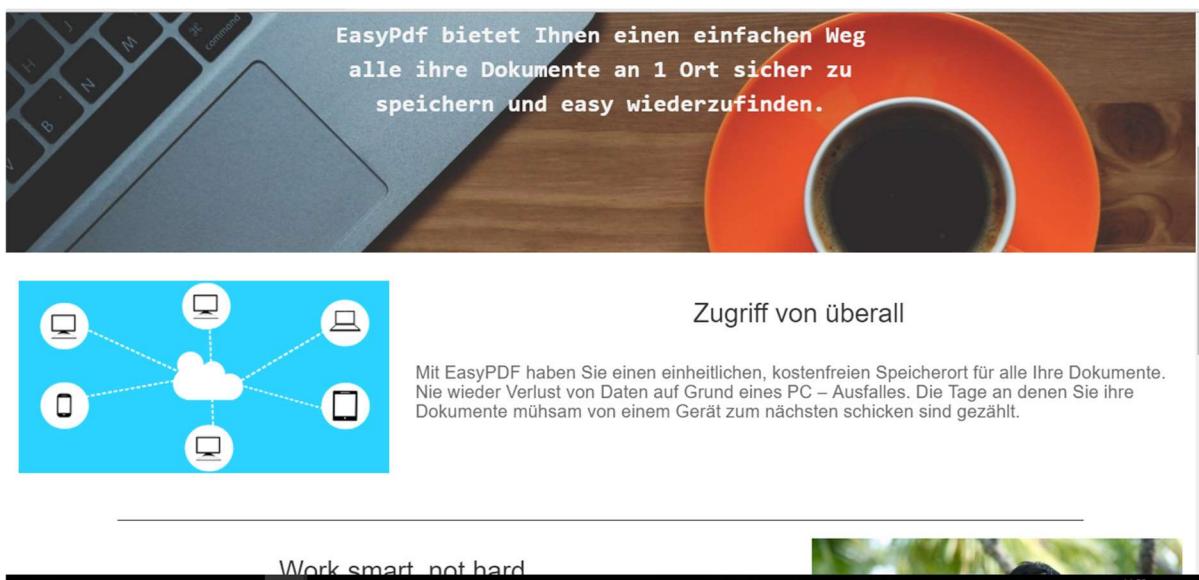


Abbildung 30 Wireframe IST



## Mobiles Design



Abbildung 32 Mobiles Design Startseite 1

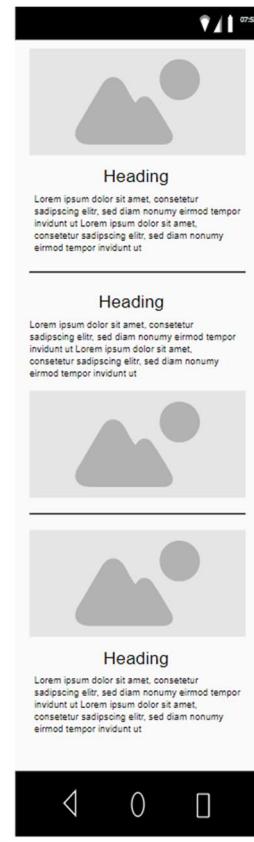


Abbildung 31 Mobiles Design Startseite 2 Menü

Der erste Eindruck der Website bleibt immer noch mit dem Hero - Image erhalten. Beim Scrolling - Teil müssen die Bilder aber abwechselnd über und unter den Text fallen, damit Bild und Text weiterhin gut lesbar bleiben.

## Das Team- Seite

## Konzept

## Aufgabe einer Team- Seite, Botschaft

Die Teamseite ist dazu da dem Benutzer die Möglichkeit zu bieten die Personen kennenzulernen, welche die Betreiber und Entwickler der Website sind. Das Ziel einer solchen Webseite ist es zum einen dem Benutzer die Möglichkeit zu geben uns zu kontaktieren und zum anderen der Aufbau und die Verstärkung der Beziehung zwischen der Website als Produkt und dem Benutzer.



## Gestaltung des persönlichen Klappentextes

Der persönliche Klapptext soll zum einen eine kurze Selbstbeschreibung zur eigenen Person liefern und zum anderen den eigenen Bezug zur Website darstellen. Da unsere Zielgruppe (*siehe Zielgruppe*) in dem Alter der Ersteller ist, ist es durchaus möglich in einem legeren Stil zu schreiben. Prinzipiell wurde der Text von jedem Mitglied selbst verfasst, wobei der obige Text als Richtlinie fungiert.

### **Klapptext Thomas Kerber**

Lachen, Arbeiten, Programmieren, Feiern und Kaffeesucht. Wenn ich mein Leben in 5 Worten beschreiben sollte, dann wären es wohl diese. Abgesehen davon bin ich ein absoluter Serienjunkie. Mein Ziel bei dieser Website ist es einen möglichst simplen und intuitiven Weg gegen das Dokumente-chaos zu bieten.

### **Klapptext Verena Gurtner**

Ein Teelöffel Chaos im Kopf, eine Prise Humor, eine Messerspitze Ordnung, ein bisschen Party und ganz viel Liebe und Geborgenheit für meine Familie ergeben mich. Ich kann den besten strukturierten Plan haben und doch schwirren tausend Gedanken in meinem Kopf umher. Deshalb war mir für unsere Website wichtig schnell, einfach und ohne viel Nachzudenken seine Dokumente wiederzufinden.

### **Klapptext Sara Hindelang**

Wenn man mich mit einem Wort beschreiben sollte, wäre dieses mit Sicherheit verplant. Egal um was es geht, mein Kopf ist eher mehr in den Wolken als sonst wo. Und genau deswegen ist EasyDocs genau das richtige Tool um mir zu helfen, den Überblick über meine Unterlagen zu bewahren um somit meine Zeit mehr in meine Hobbies wie Yoga, lesen & wandern investieren zu können.

## Design

### Aufbau der Team- Seite

Die Teamseite besteht aus einem großen Hintergrundbild und mehreren Karteikarten für die jeweiligen Teammitglieder. Für das Hintergrundbild wurde ein Holzbrett gewählt, da dieses keinen Inhalt hat und der Großteil der Seite von den Karteikarten bedeckt wird.



## Aufbau der Karteikarten

Die Karteikarten sind 4 – geteilt und bestehen aus einem Profilbild, dem Namen, einer Beschreibung der Position innerhalb des Projekts und einem kurzen Klapptext. Bei den einzelnen Profilbildern haben wir uns für Fotos aus unserer Kindheit entschieden, da dies einen gewissen Charme hat und wir als junge Menschen noch mitten in der Veränderung sind.

## Gegenüberstellung des Konzeptes mit dem Resultat

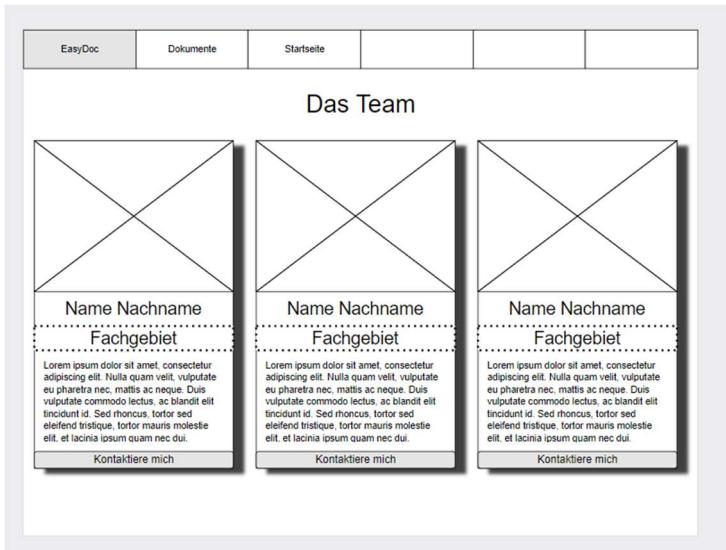


Abbildung 33 Wireframe Meet the Team SOLL

Easy DOCS	Dokumente
--------------	-----------

**Unser Team**

Sara Hindelang	Verena Gurtner	Thomas Kerber
Dokumentanalyse	Datenbankentwicklerin	Frontend Entwickler
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed diam nonummy elmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.		

Abbildung 34 Meet The Team IST



## Registrier- Seite

### Konzept

#### Aufgabe einer Registrierseite - Botschaft

Die Registrierseite dient dazu, dass der Kunde sich auf unserer Website einen Account machen kann. Sie sollte einfach zu bedienen sein und den Kunden rechtzeitig warnen, wenn seine Daten nicht korrekt sind.

#### Ablauf des Registrier-Zykluses

Zuerst gibt der Benutzer seine Daten in das Formular ein. Diese werden von der Website überprüft und erst wenn alle Daten korrekt angegeben werden, wird der Registrier - Knopf freigeschalten. Nachdem der Benutzer die Daten mit dem Knopf abgeschickt hat, wird ihm eine Ladeanimation angezeigt. Abhängig davon, ob die Daten am Server auch als in Ordnung betrachtet werden, erhält der Benutzer die entsprechende Antwort. Wenn die Durchführung in Ordnung ist, wird ihm angeboten auf die Startseite zurückzukehren und wenn nicht, erhält er eine Fehlermeldung mit dem entsprechenden Fehler.



Grafische Veranschaulichung des Ablaufs:

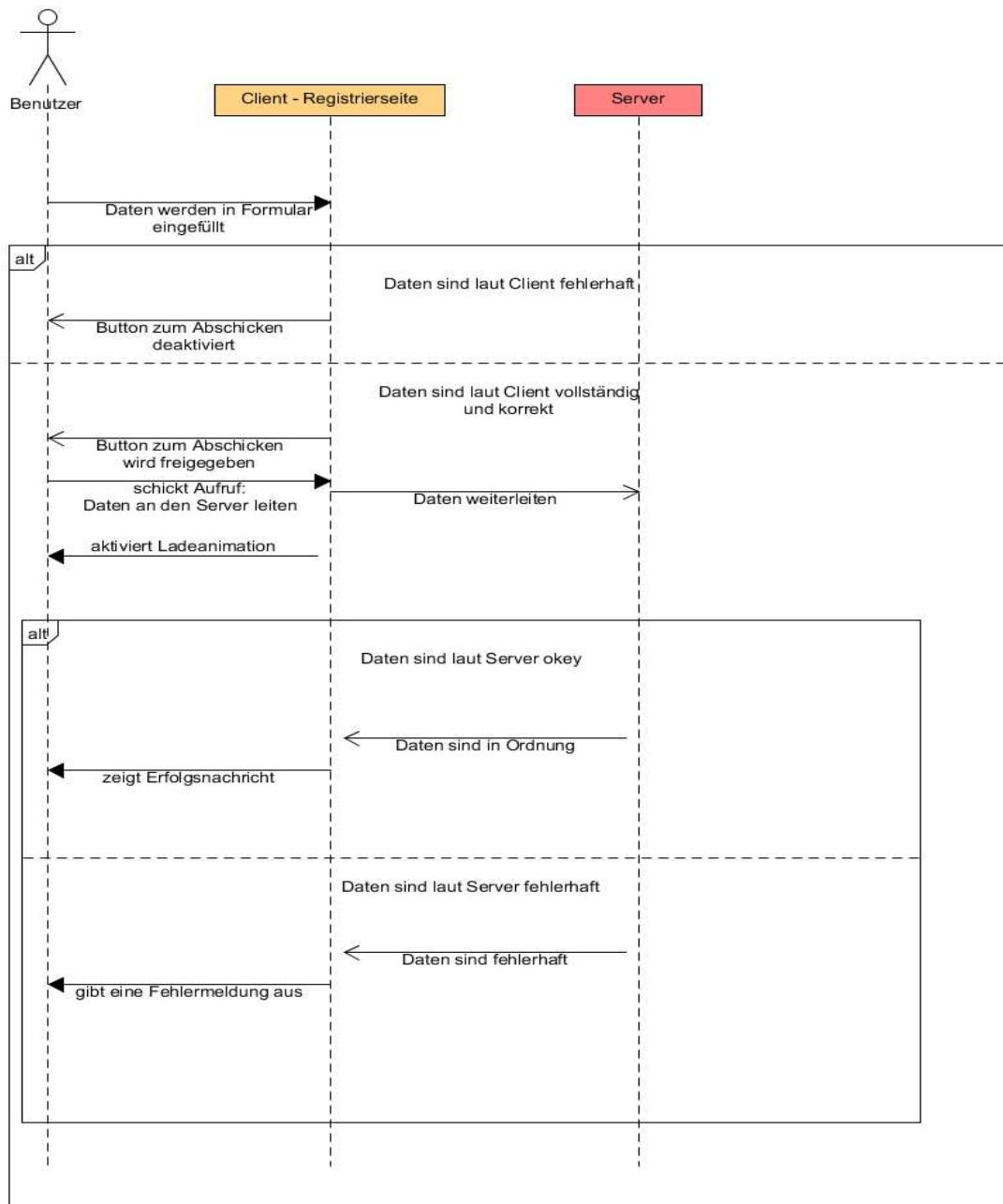


Abbildung 36 Registrieren Clientseitig Sequenzdiagramm

## Design

### Aufbau der Registrierseite

Die Registrierseite besteht aus 2 wichtigen Teilen. Zum einen ist dies das Hintergrundbild, welches wie ein Hero – Image fungiert (*siehe HeroImage*).



## Auswahl des Hero Image

Als Hero Image wurde eine Glaskuppel gewählt, welche in der Mitte ein Loch hat. Durch die Verzweigungen soll eine Verbundenheit zur Technik symbolisiert werden. Im Zentrum des Bildes steht dann auf der Website das 2. Element. Das Bild konzentriert sich klar um das Zentrum herum.



Abbildung 37 Heroimage Login, Registrieren ect..

## Aufbau des Formulars

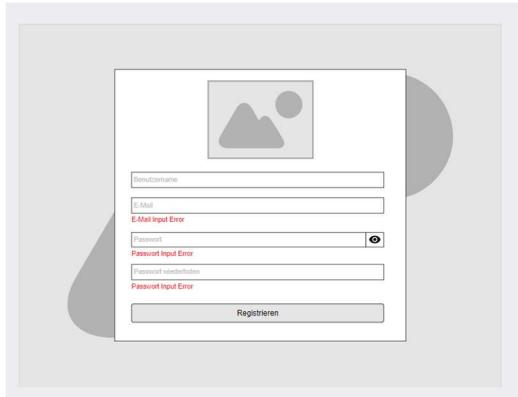
Im Zentrum des Hero – Images steht ein Block, welcher eine Ebene über dem Image platziert ist. Dieser beinhaltet das Formular zur Registrierung. Zuerst ist unser Logo zu sehen, um die Markenverbundenheit zu stärken. Dann erfolgen die einzelnen Inputs:

1. Benutzername
2. E-Mail
3. Passwort
4. Passwort wiederholen

Passwörter werden grundsätzlich nicht im Klartext angezeigt. Um Fehleingaben zu vermeiden, muss das Passwort zweimal eingegeben werden.

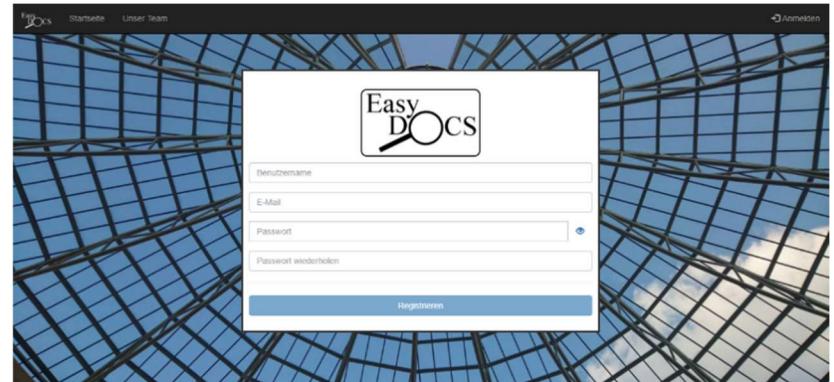


## Gegenüberstellung des Konzeptes mit dem Resultat



A wireframe diagram of a registration form. It features a placeholder image of a person's head and shoulders at the top. Below it are four input fields: 'Benutzername', 'E-Mail', 'Passwort', and 'Passwort wiederholen'. Each field has a red error message below it: 'Email Input Error', 'Passwort Input Error', and 'Passwort Input Error' respectively. A 'Registrieren' button is at the bottom.

Abbildung 38 Wireframe Register Plan



A wireframe diagram of a registration form integrated into a larger website layout. The background shows a photograph of a modern building with a glass and steel structure. The registration form is a white box containing the same fields as the plan: 'Benutzername', 'E-Mail', 'Passwort', and 'Passwort wiederholen'. It also includes a 'Registrieren' button. The top of the page has a navigation bar with links like 'Startseite' and 'Unser Team'.

Abbildung 39 Wireframe Register IST

In der Testphase hat sich ergeben, dass auf jeder Seite eine Navigationsbar benötigt wird, um den User nicht auf eine potenzielle dead – end Seite zu schicken.

## Mobiles Design

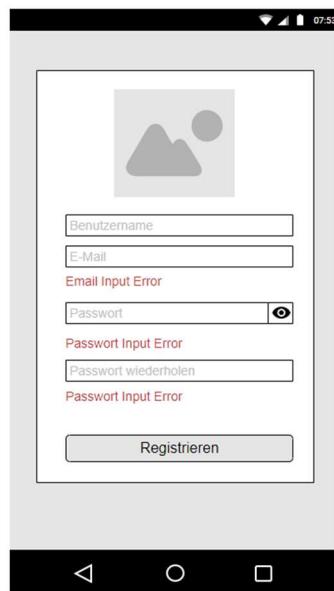


Abbildung 40 Mobiles Registrieren

Auch diese Seite ist bis auf die Navigationsbar so übernommen worden.



## Technische Umsetzung

### Aufbau des Body – Teils

```
<body>

    <div class="heroimage">

        <div class="container">

            <div class="center-content">

            </div>

        </div>

    </div>

</body>
```

Der `<div> „heroimage“` beinhaltet als Hintergrundbild das Hero – Image.

Der `<div> „container“` sorgt für die Skalierung des Inhaltes mit der gesamten Seite

Der `<div> „center-content“` beinhaltet die Box mit dem Formular. Er sorgt dafür, dass diese innerhalb des container-div zentriert ist.

Es wäre möglich den „container“ und „center-content“ zusammenzufassen, jedoch stammt diese Struktur aus einem ausgearbeitetem persönlichem Template, welches für mehrere Webseiten verwendet wird. Des Weiteren ist der „container“ eine vorgefertigte Klasse von Bootstrap, welche eine automatische Skalierung mit der Bildschirmauflösung vornimmt. (*siehe Bildschirmauflösung*)

### Clientseitige Validierung

Die clientseitige Validierung erfolgt mittels des Plug-Ins Bootstrap Validator. (*siehe Bootstrap- Validator*).

Hierbei wird mit den vorgegebenen Validierungsmöglichkeiten die Minimallänge, die Emailgültigkeit, sowie das Ausfüllen des Inputs überprüft. Zusätzlich wurden noch eigene Validierungen geschrieben, welche überprüfen, ob das Passwort eine Zahl hat, keine Sonderzeichen besitzt und nicht länger als 20 Zeichen ist. Um diese eigenen Funktionen hinzuzufügen, ist "data-funktionsname = 'parameter' " als Attribut beim Input anzugeben.



Bsp. Hinzufügen der eigenen maxlength – Funktion zum Input.

```
<input type="password" data-maxlength="20">

<script>
  $("#registerform").validator({
    Custom: {
      Maxlength: function($el) {
        var maxValue = $el.data("maxlength"); // bekommt maxlänge
        if($el.val().length > maxValue) return "Das Passwort ist zu lang."
      }
    }
  })
</script>
```

## Darstellen des Passworts in Klartext

Um das Passwort im Klartext darzustellen, muss theoretisch lediglich der „type“ des Inputs von "password" zu "text" umgestellt werden. Dies könnte theoretisch mit dem JavaScript Befehl Element.type=„text“ passieren. Dies wird jedoch im Internet Explorer erst ab Version 10 unterstützt. Davor ist es nicht möglich ein Attribut eines Input-Elementes zu ändern. Um dies zu umgehen, wird mittels JavaScript der Inhalt des Password-Inputs ausgelesen und dann wird der gesamte Password-Input mit einem Text-Input ausgetauscht, wobei im Text-Input als „value“ das Passwort angegeben wird.

```
if($("#pwinput").attr('type') == 'password'){
  var input = $("#pwinput");
  var pw = input.val(); //Speichert das Passwort
  //Ersetzt das Input-Feld
  input.replaceWith("<input type=\"text\" name=\"password\" data-containsnumber=\"YES\" id=\"pwinput\" data-maxlength=\"16\" placeholder=\"Passwort\" data-minlength=\"8\" data-maxlength-error=\"Das Passwort darf maximal 16 Zeichen lang sein\" data-minlength-error=\"Das Passwort muss mindestens 8 Zeichen lang sein\" class=\"form-control\" value=\""+pw+"\" required>");
  //Verändert den Button in ein geschlossenes Auge
  $(this).html("<span class=\"glyphicon glyphicon-eye-close\"></span>")
}
```



```
// Führt das gleiche auch für das 2. Passwortfeld aus

var input2 = $("#pwinput2");

var pw2 = input2.val();

input2.replaceWith("<input type=\"text\" name=\"password\" id=\"pwinput2\" placeholder=\"Passwort wiederholen\" class=\"form-control\" data-match=\"#pwinput\" data-match-error=\"Passwörter stimmen nicht überein\" value=\""+pw2+"\\" required>");

register.validator('update'); // Validator er muss noch einmal überprüfen

}

else{

// Ersetzt den Input wieder mit dem type password


}
```

## Login- Seite

### Konzept

#### Aufgabe der Login- Seite

Die Login- Seite dient dazu dem Benutzer die Möglichkeit zu geben sich mit seinen Daten anzumelden und zu seinen individuellen Dokumenten zu gelangen. Sie soll einfach und schnell zu bedienen sein. Zusätzlich gibt es auf der Login- Seite noch eine Unterseite, welche es dem Benutzer ermöglicht sich via Mail ein neues Passwort zu setzen. (*siehe Sitemap und Emailsenden*)

#### Ablauf des Login- Zyklus

Beim Login gibt der Benutzer seine Daten an und klickt dann auf den Submit- Button. Sind die Daten korrekt wird er weitergeleitet auf die Dokumentseite, ansonsten bleibt er auf der Login- Seite und erhält eine entsprechende Fehlernachricht als Feedback. Diese Nachricht ist erforderlich, da der Benutzer stets eine Resonanz auf seine Aktionen erleben sollte.

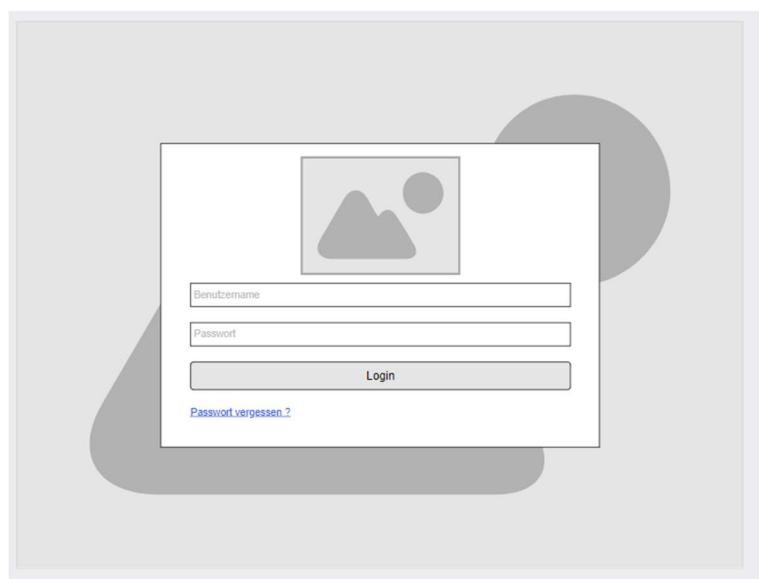


## Design

### Aufbau der Website

Das Design ist nahezu ident von der Konzeption und Aufbau mit der Registrierseite. Der einzige Unterschied liegt in den verschiedenen Parametern in der Form.

Gegenüberstellung des Konzeptes mit dem Resultat



A wireframe diagram of a login interface. It features a central rectangular form with rounded corners. At the top center is a placeholder image of a user profile. Below the image are three input fields: 'Benutzername' (Username), 'Passwort' (Password), and a large 'Login' button. At the bottom of the form is a small blue link labeled 'Passwort vergessen?'. The entire form is set against a light gray background with a large, semi-transparent gray circle centered behind it.

Abbildung 42 Wireframe Login Plan

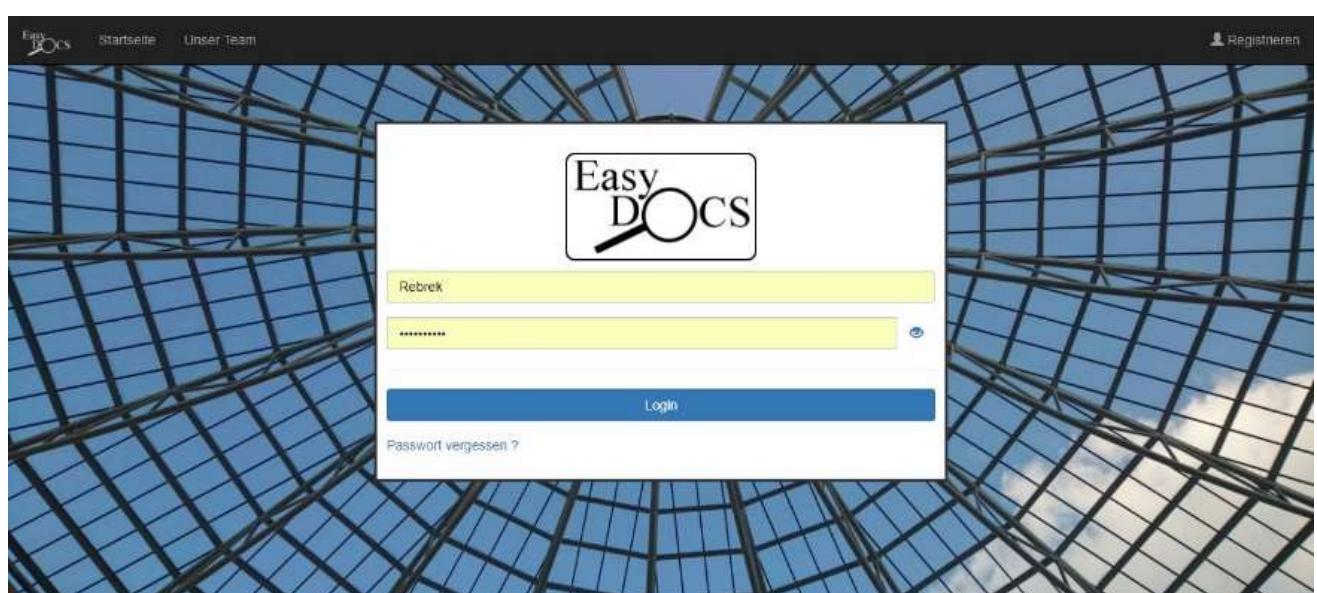


Abbildung 41 Wireframe Login IST



## Mobiles Design



Abbildung 44 Login Mobil IST



Abbildung 43 Login Mobil Plan

## Technische Umsetzung

Die technische Umsetzung erfolgt analog zu der Registrierseite.

## Dokument - Seiten

## Konzept

## Vorwort – gelöschte Dateien - Seite

Da die "gelöschte Dokumente Seite" lediglich eine reduzierte Version der Dokument Seite ist, gelten die Erklärungen der Dokument Seite auch für diese.

## Aufgabe der Dokumentseite

Die Dokumentseite ist das Herzstück unserer Website. Auf ihr kann man seine Dokumente hochladen und sie sich anzeigen lassen. Auch die Sortier- und Suchfunktion ist hier untergebracht.



## Ablauf der Dokumentseite

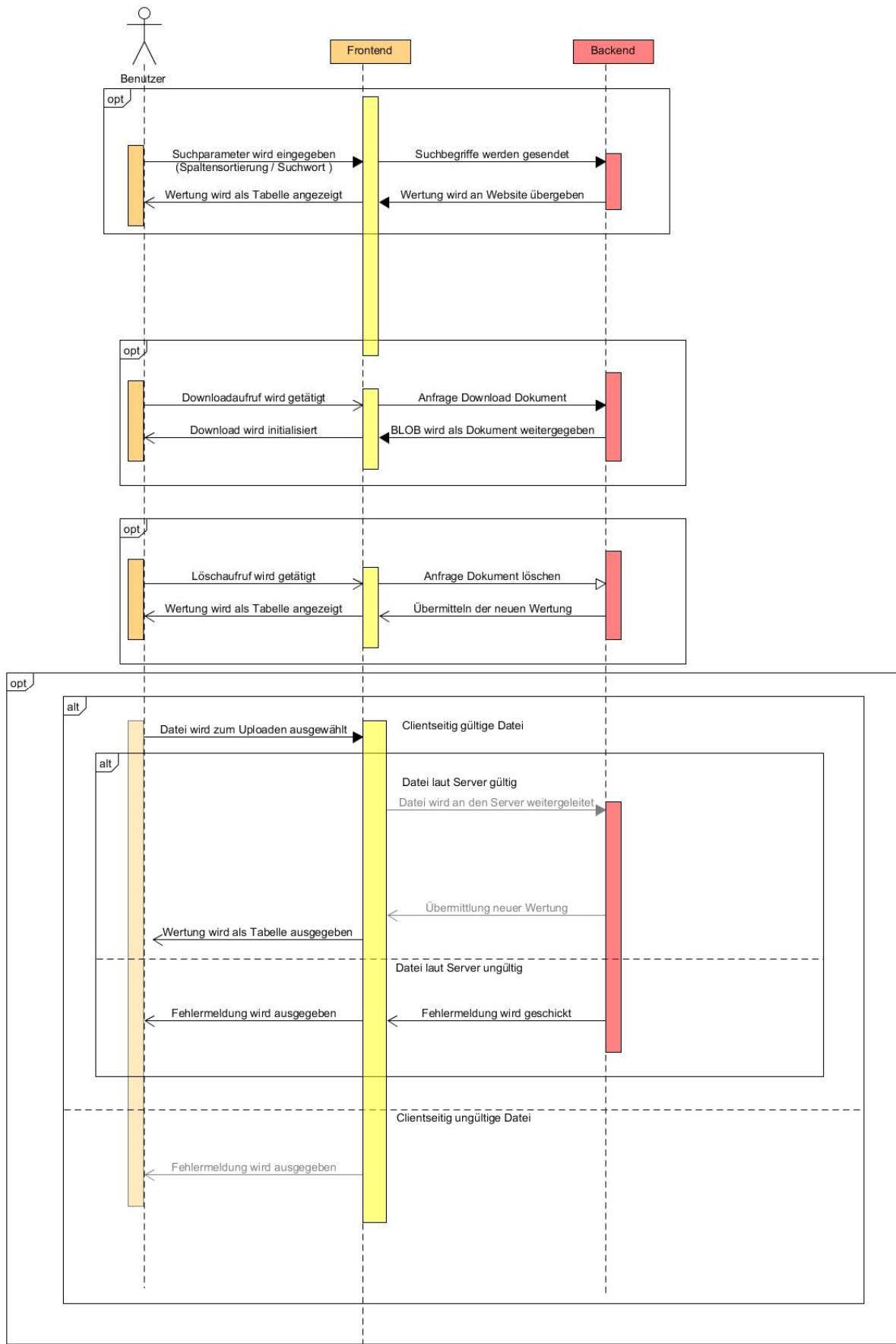


Abbildung 45 Sequenzdiagramm Dokumentseite



Der Benutzer hat auf der Dokumentseite eine Vielzahl an Möglichkeiten. Er kann Dokumente speichern, herunterladen, löschen und sortieren lassen. Bei jeder dieser Möglichkeiten wird der Inhalt der Tabellen neu vom Server angefragt und angezeigt. Zusätzlich werden Dokumente beim Hochladen noch auf ihre Endung überprüft und es wird kontrolliert, ob der Name bereits schon einmal verwendet wurde. In diesem Falle hat der Benutzer die Möglichkeit einen neuen Namen einzugeben.

## Design

### Aufbau der Dokumentseite

Die Dokumentseite besteht aus 2 Kernelementen. Dies ist zum einen ein Bereich zum Hochladen der Dokumente und zum anderen die Tabellen, welche die Dateien der User darstellen. Da der Benutzer hauptsächlich mit den Tabellen arbeitet und um den Tabellen so viel Platz wie möglich zu bieten, habe ich die Speicherung in einen modalen Dialog ausgelagert, welches mittels eines Buttons in der Navigationsbar aufgerufen werden kann. Die Tabellen selbst stehen zentriert auf der Seite mit einer jeweiligen Überschrift, welche eine Information zu den Dokumenten angibt.

### Gegenüberstellung Konzept mit Resultat

Das Design ist vollständig übernommen worden.

The screenshot displays a web-based document management interface. At the top, there is a navigation bar with tabs: 'EasyDoc', 'Das Team', 'Gelöschte Dokumente', 'Upload', an empty space, and 'Abmelden'. Below the navigation bar, there are two main sections:

- Private Dokumente:** This section contains a table with the following data:

DateiTyp	Name	Autor	UploadDatum	DokumentDatum	Zugang	Download	Delete
PDF	Die Verwandlung	Franz Kafka	10.3.2017	1.1.1915	Public	<input type="radio"/>	<input checked="" type="radio"/>
DOC	Faust	Goethe	10.3.2017	19.1.1829	Public	<input type="radio"/>	<input checked="" type="radio"/>
TXT	Laborbericht - NWES	Max Muster	10.3.2017	7.3.2017	Public	<input type="radio"/>	<input checked="" type="radio"/>
DOC	Faust	Goethe	10.3.2017	19.1.1829	Public	<input type="radio"/>	<input checked="" type="radio"/>
DOC	Faust	Goethe	10.3.2017	19.1.1829	Public	<input type="radio"/>	<input checked="" type="radio"/>
- Öffentliche Dokumente:** This section contains a table with the following data:

DateiTyp	Name	Uploader	UploadDatum	DokumentDatum	Download
PDF	Die Verwandlung	User1	10.3.2017	1.1.1915	<input type="radio"/>
DOC	Faust	LiteraturFan1	10.3.2017	19.1.1829	<input type="radio"/>
TXT	Laborbericht - NWES	Schüler98	10.3.2017	7.3.2017	<input type="radio"/>
DOC	Faust	User2	10.3.2017	19.1.1829	<input type="radio"/>
DOC	Faust	User3	10.3.2017	19.1.1829	<input type="radio"/>

Abbildung 46 Dokumentseite Plan



Easy DOCS   Über EasyDoc   Gelöschte Dokumente   UPLOAD   Abmelden

## Meine Dokumente

10 ▾

DateiTyp	Name	Autor	UploadDatum	DokumentDatum	Zugang	Download	Delete
	Allgemeines über Texte	Wikipedia	18.03.2018	01.05.2017	<input type="button" value="Private"/>		
	Kerber-Griesebner.pdf	Thomas Kerber	24.03.2018	07.06.2017	<input type="button" value="Public"/>		
	schule.pdf	Verena Gurtner	18.03.2018	21.01.2018	<input type="button" value="Public"/>		
	Schwäbischer Zwiebelkuchen.pdf	Verena Gurtner	18.03.2018	21.01.2018	<input type="button" value="Private"/>		

Q

1 bis 4 von 4 Einträgen 1

## Öffentliche Dokumente

10 ▾

DateiTyp	Name	Uploader	Autor	UploadDatum	DokumentDatum	Download
	Kerber-Griesebner.pdf	verena	Thomas Kerber	24.03.2018	07.06.2017	
	schule.pdf	verena	Verena Gurtner	18.03.2018	21.01.2018	

Q

1 bis 2 von 2 Einträgen 1

Abbildung 47 Dokumentseite IST



## Mobiles Design

DateiTyp	Name	Autor
WORD	Deutsch-Aufsatz	Thomas Kerber
PDF	Laborbericht	Verena Gurtner
PDF	Laborbericht	Sara Hindelang
WORD	Textanalyse	Max Muster 10

## Öffentliche Dokumente

DateiTyp	Name	Uploader
WORD	Deutsch-Aufsatz	usernr2
PDF	Die Welle	xXxUser
WORD	FMEA - Tutorial	User2
PDF	Diplomarbeit - EasyDoc	User4

Abbildung 49 Mobile Dokumentseite  
IST

Auch bei der mobilen Umsetzung wurde die Vorlage übernommen.

DateiTyp	Name	Autor
WORD	Deutsch-Aufsatz	Thomas Kerber
PDF	Laborbericht	Verena Gurtner
PDF	Laborbericht	Sara Hindelang
WORD	Textanalyse	Max Muster 10

DateiTyp	Name	Uploader
WORD	Deutsch-Aufsatz	usernr2
PDF	Die Welle	xXxUser
WORD	FMEA - Tutorial	User2
PDF	Diplomarbeit - EasyDoc	User4

Abbildung 48 Mobile Dokumentseite Plan

## Technische Umsetzung

### Implementierung der Dropzone

Vor dem eigentlichen Upload einer Datei müssen eine Reihe von Überprüfungen durchgeführt werden. Die folgende Grafik veranschaulicht die Abfolge der Aktionen.

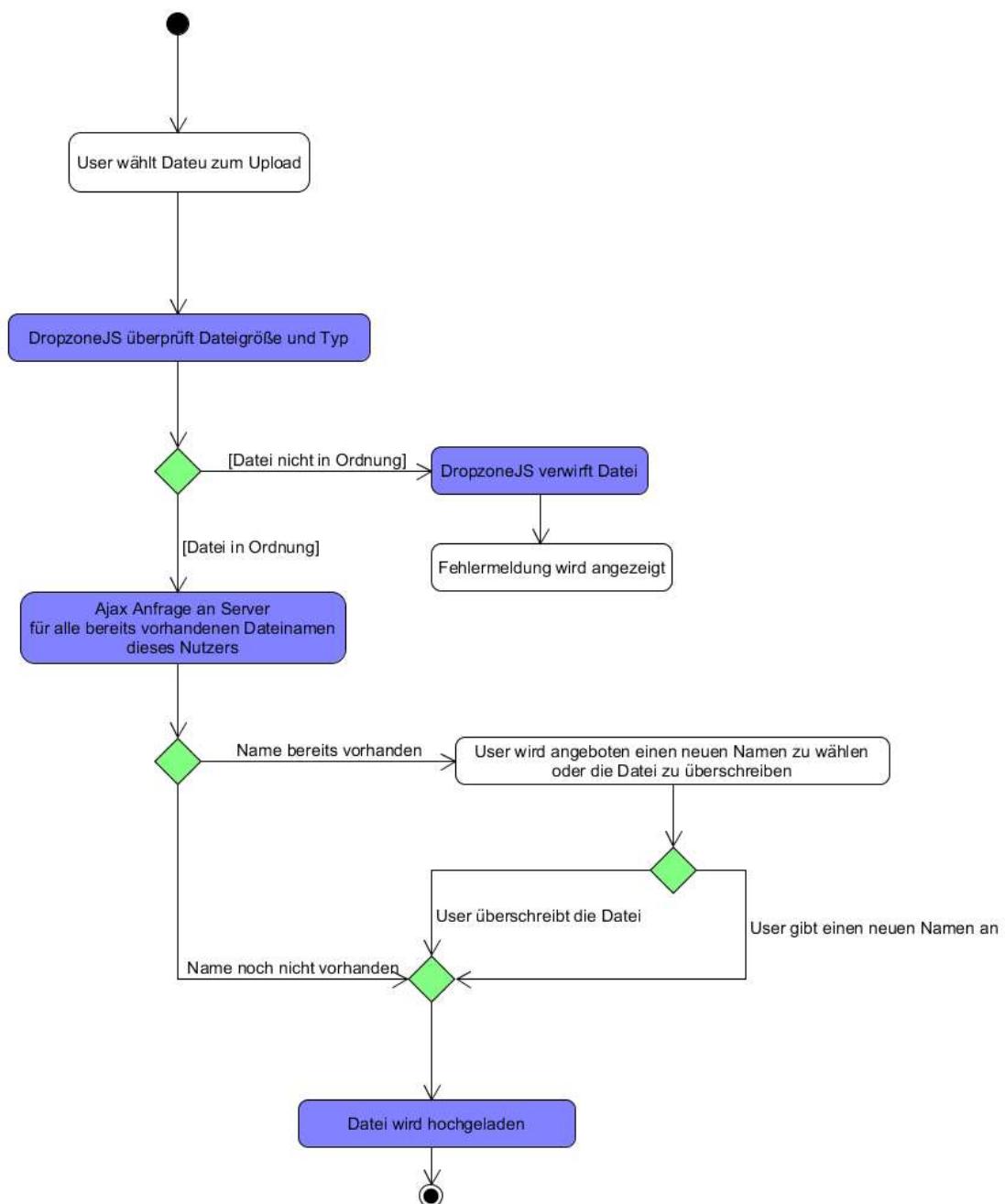


Abbildung 50 Ablaufdiagramm Upload



Vergleicht den Namen des Dokumentes mit den Namen des Arrays

```
function getDokumentNamen(namedatei) {  
  
    $.getJSON("DateienListServlet",function(responseText) {  
  
        var DokumentNamen = responseText;  
  
        var vorhanden = $.inArray(namedatei, DokumentNamen);  
  
        if (vorhanden === -1) {  
            sendfile();  
        } else {  
            // Name ist bereits vorhanden →  
            //addChecker überprüft Eingabe im neuen Fenster  
            addChecker(DokumentNamen);  
  
            // Fenster zum Überschreiben wird aufgerufen  
            showModal(namedatei);  
        }  
    });  
}
```

Konfiguration der akzeptierten Dateitypen, der Dateigröße, des verwendeten Servlets und der Übersetzung

```
maxFilesize : size,  
  
    paramName : "pdffile",  
  
    addRemoveLinks: true,  
  
    url : "UploadServlet",  
  
    acceptedFiles :  
"application/pdf,application/msword,application/vnd.openxmlformats-  
officedocument.wordprocessingml.document,text/plain",
```



```
    parallelUploads : 1,  
  
    autoQueue : false,  
  
    autoProcessQueue : false,  
  
    dictDefaultMessage : "Ziehe Dateien hierhin zum Hochladen",  
  
    dictFallbackMessage : "Dieser Browser wird leider nicht  
unterstützt",  
  
    dictFileTooBig : "Die Datei ist leider zu groß. Erlaubtes Maximum  
sind "+ size + " MB",  
  
    dictInvalidFileType : "Dies ist leider der falsche Dateityp. Es  
werden nur .pdf,.doc,.docx und .txt Dateien unterstützt"
```

## Implementierung des jQuery DataTables

### *Konfiguration des DataTables*

Zuerst muss die Datenerzeugung serverseitig konfiguriert werden.

```
var table2 = $("#datatable2").DataTable({  
"processing" : true, //während die Daten vom Server geholt werden erscheint ein  
//Ladezeichen  
"serverSide" : true, // gibt an, dass das Sortieren der Daten am Server erfolgt
```

### **Konfiguration des Ajax**

```
"ajax" : {  
        "url" : 'DataTableServlet', //URL des Servlets  
        "type" : 'POST', //Methode zum Senden  
        "data" : function(act){ //fügt zusätzliche  
//Daten zum Request.  
            act.user = '${user}'; // Name des Users  
            act.table = "table2"; // Table, welcher req  
//macht  
        },  
        "dataSrc": "data" //Verweist auf Struktur  
    },
```

Dann wird unter dem Punkt "language" die Sprache konfiguriert:  
Beispiel:

```
"language":(  
    "sEmptyTable": "Keine Daten in der Tabelle vorhanden",  
    .....  
)
```



## Verknüpfen der Daten mit der Tabelle

Hier wird definiert welche Daten von der JSON – Antwort, in welcher Spalte angezeigt werden. Reihenfolge der Spalten erfolgt nach Reihenfolge der Auflistung. Bei den Zugangsdaten wird eine Funktion angegeben, weil abhängig von der Einstellung entweder Öffentlich oder Privat eingestellt ist.

```
"columns" : [  
    {"data" : "ID"},  
    {"data" : "DateiTyp"},  
    {"data" : "Name"},  
    {"data" : "Uploader"},  
    {"data" : "Autor"},  
    {"data" : "UploadDatum"},  
    {"data" : "DokumentDatum"},  
    {"data" : function (Daten){  
        var state = Daten["ZUGANG"];  
        var text="";  
        switch(state){  
            case "public":  
                text = "<select class=\"privacy form-control fa\"><option value=\"public\" selected> &#xf0ac; Public</option><option value=\"private\">&#xf023 Private</option></select>";  
                break;  
  
            case "private":  
                text = "<select class=\"privacy form-control fa\"><option value=\"public\">&#xf0ac; Public</option><option value=\"private\" selected>&#xf023; Private</option></select>";  
                break;  
            default:  
                text="ERROR";  
                break;  
        }  
        return text;  
    }},  
    {"data" : "Download"},  
    {"data" : "Delete"}  
],
```

## Darstellen der Download und Delete Buttons

Code zum Erzeugen der Buttons

```
"columnDefs": [  
{  
    "targets": -2,  
    "searchable": false  
  

```



## Hinzufügen der Suchfunktion

Nachdem der DataTable sich fertig initialisiert hat, wird die standartmäßige Suchfunktion überschrieben und es werden 2 Button – Elemente hinzugefügt. Eines um die Suche zu starten und das andere um den Suchtext zu löschen. Dies ist notwendig, da standardmäßig bei jedem eingegebenen Buchstaben eine Suchanfrage geschickt wird und dies eine wesentlich höhere Auslastung der Datenbank zur Folge hat.

```
initComplete: function() { // wird aufgerufen, wenn der DataTable fertig geladen ist

var input = $('#datatable2_wrapper .dataTables_filter input').off();

//Löscht alle existierenden Listener von der Inputbox

self = this.api();

// referenziert den DataTable in eine Variable, damit er innerhalb der Suchen - Funktion
//aufgerufen werden kann

$searchbutton = $('<button class=\"btn-success dttopbtn\" data-toggle=\"tooltip\" title
=\\\"Suchen\\\">') //erstellt ein Buttonobjekt

.html('<span class="glyphicon glyphicon-search"/>') // Button - Text: Suchen

.click(function() { //Funktion welche bei drücken des Buttons aufgerufen wird

self.search(input.val()).draw();

//ruft die Search - Funktion des DataTables auf und aktualisiert

});

$deletebutton = $('<button class=\"btn-danger dttopbtn\" data-toggle=\"tooltip\" title
=\\\"Löschen\\\">') //erstellt ein Buttonobjekt

.html('<span class="glyphicon glyphicon-remove"></span>') //Button - Text: Löschen

.click(function() { //Funktion welche bei drücken des Buttons aufgerufen wird

input.val(''); //Setzt den Input wieder leer

$searchbutton.click();

//Betätigt die searchbutton - funktion, jetzt jedoch mit leerem Inhalt, zum Aktualisieren

});

$('#datatable2_wrapper .dataTables_filter').append($searchbutton, $deletebutton);

//Fügt beide Buttons zum DataTable hinzu

},
```



## Löschen, Download und Verfügbarkeit ändern - Funktionen

Diese Funktionen hören auf das Event, welches aufgerufen wird, wenn ein Downloadbutton bzw. Deletebutton gedrückt wird.

### Delete – Funktion

```
$(".table tbody").on("click",".deletebutton",function(){
    var sourcetable = getTableRow($(this));

    $.ajax({
        method:"POST",
        url:"DeleteServlet",
        data: {todelete: sourcetable}
    })
    .done(function(){
        refreshTables();
    })
});
```

Die Funktion „getTableRow“ liefert die aktuelle Reihe der Tabelle auf welcher sich der Button befindet. Ein Ajax Request mit der Information, welche Daten gelöscht werden sollen, geht an das DeleteServlet. Nach Abschluss des Ajax – Requests wird die Tabelle neu geladen.

### Download – Funktion

Auf Grund der Same Origin Policy ist es nicht möglich ein File Objekt mittels JavaScript auf dem PC zu speichern. Diese Policy sagt aus, dass JavaScript lediglich Objekte in seiner eigenen Sandbox verändern kann. Darunter fallen in diesem Fall der DOM der Website und natürlich das Script selbst. Deswegen ruft die Funktion mittels „window.location“ ein neues Fenster auf, welches dann einen Download initialisiert. Des Weiteren muss darauf geachtet werden, dass in der URL des geöffneten Fensters keine Leerzeichen oder sonstige illegale Zeichen vorkommen, da diese ansonsten missinterpretiert werden können und vom Browser abgewiesen werden könne. (Quelle: RFC 1738). Dieses Problem wird verhindert indem die URL vorher encoded wird. Dadurch werden die Zeichen dann in eine für den Browser verständliche Form verwandelt. Bsp.: Aus dem Leerzeichen wird %20.

```
$('.table tbody').on( 'click', '.downloadbutton', function () {
    var sourcetable = getTableRow($(this));
    var newsourcetable = encodeURIComponent(sourcetable);
    console.log(newsourcetable);
    window.location="DownloadServlet?download="+newsourcetable;
});
```

### Verfügbarkeit – Funktion

Die Verfügbarkeitsfunktion ist gleich aufgebaut wie die Delete – Funktion.



## Aufbau der JSON – Nachrichten

Dies ist die JSON Nachricht, welche vom Server erhalten wird.

```
{  
    "draw": "1",  
    "recordsTotal": 11,  
    "recordsFiltered": 11,  
    "data": [  
        {  
            "ID": "184",  
            "DateiTyp": "PDF",  
            "Name": "AA05-NAT-Kerber.pdf",  
            "Uploader": "thomas",  
            "Autor": "Thomas Kerber",  
            "UploadDatum": "15.03.2018",  
            "DokumentDatum": "01.12.2017",  
            "ZUGANG": "public"  
        },  
        {  
            "ID": "186",  
            "DateiTyp": "DOCX",  
            "Name": "AA15-wifi.docx",  
            "Uploader": "neu",  
            "Autor": "c102wm",  
            "UploadDatum": "18.03.2018",  
            "DokumentDatum": "22.02.2018",  
            "ZUGANG": "public"  
        },  
        {  
            "ID": "181",  
            "DateiTyp": "PDF",  
            "Name": "Abgabe.pdf",  
            "Uploader": "verena",  
            "Autor": "Verena Gurtner",  
            "UploadDatum": "01.03.2018",  
            "DokumentDatum": "11.12.2017",  
            "ZUGANG": "public"  
        }  
    ]  
}
```

Siehe JSON



## Lizenzen

MIT – Lizenz: Die MIT Lizenz erlaubt die Wiederverwendung der Software sowohl zum Kommerziellen als auch nicht kommerziellen Zweck. Die Software darf verändert, kopiert und sogar vertrieben werden. Jedoch wird keinerlei Haftung auf die Software gewährleistet.

Lizenztext:

Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## Ausblick in die Zukunft Thomas:

Die Frontend – Schnittstelle entspricht dem derzeit aktuellsten Stand der Technik. Dieser ist jedoch gerade in der Webentwicklung sehr kurzlebig. Deswegen wäre ein möglicher Verbesserungsschritt in der Zukunft die Anpassung des JavaScript – Codes auf ECMAScript 6. hierdurch könnte dann auch sehr einfach auf jQuery verzichtet werden. Ein weiterer Verbesserungsschritt wäre die Ersetzung von Bootstrap mit dem nativen CSS Grid, wenn dieses von allen Browsern unterstützt wird. Auch der Download könnte in Zukunft mittels JavaScript ablaufen, wenn eine entsprechende API von allen Browsern unterstützt wird. Aktuell gibt es eine solche Unterstützung nur auf Chrome, wobei diese noch nicht vollständig funktioniert. Die Register und Login Funktionen könnten mit Ajax anstatt mit einem Submit umgesetzt werden.



## Aufgabenbeschreibung Sara - Controller-Teil

Dieser Teilbereich der Diplomarbeit umfasste die Analyse beziehungsweise die Extraktion der Dokumente, um sie nach dem Hochladen durch den Benutzer auf der Website in die Datenbank zu speichern. Zu Beginn waren nur PDFs angedacht, im Laufe der Diplomarbeit stellte sich jedoch heraus, dass es sinnvoller wäre, den Anwendungsbereich auf die neuen, sowohl als auch alten Word- Formate auszuweiten. Des Weiteren wurde als kleine Ergänzung auch noch das .txt - Format miteinbezogen.

Umgesetzt wurde die Verarbeitung der Dokumente mittels Libraries. Die Libraries wurden jeweils aus dem Internet bezogen und weiterverwendet, da diese effektive Tools zur Datenverarbeitung darstellen und auch die neuesten Technologien verwenden und sich auch über die Zeit in den verschiedensten Aspekten bewährt haben.

Des Weiteren wurde Serverseitig ein benutzerbezogenes System erstellt, welches es ermöglicht einen Nutzer anzulegen und danach auch auf diesen bezogen die Daten zu verwalten mit weiteren Features die diesen Zyklus voll funktionsfähig machen, sowie die generelle clientseitige Verarbeitung der Dokumente allgemein wozu unter anderem der Up- und Download zählen.



## Extraktion der Dokumente Durch Libraries PDFs mittels PDFBox

Dokumente des PDF- Formats wurden durch die PDFBox- Library Analysiert und extrahiert.



Abbildung 51 Logo PDFBox

Apache PDFBox ist eine *open source*<sup>1</sup> Java- Library, welche das Arbeiten mit PDF Dokumenten erleichtern sollte. Mittels ihr kann man PDF Dokumente erstellen, bereits bestehende verändern und auch, was in dieser Diplomarbeit einen wichtigen Teilbereich darstellt, den Inhalt von vorgegebenen PDF Dokumenten herauslesen.

Die Apache PDFBox besteht aus folgenden Komponenten:

- PDFBox: Kernfunktion
- FontBox: Umgang mit Schriftarten
- XmpBox: Handling von standardisierten Metadaten
- Preflight (optional): überprüft PDF Dokumente auf PDF/A-1b Konformität<sup>2</sup>

Als Lizenz verwendet PDFBox die Apache Licence. Sie ist eine Freie Software-Lizenz, was auch von der ‚Free Software Foundation‘ anerkannt wurde, und dessen aktuellste Fassung, Version 2.0, wurde 2004 veröffentlicht. Sie beinhaltet die Erlaubnis die Software frei in jedem Umfeld zu verwenden, modifizieren und auch zu verteilen. Jedoch ist zu beachten, dass jedem verwendeten Paket eine Kopie der Lizenz beiliegen muss.

Abgesehen von PDFBox hätte es auch Libraries wie zum Beispiel jPDFText, iText oder PDFTextStream gegeben. Die genannten Libraries unterscheiden sich im Wesentlichen in den Punkten Lizenz, Anwendbarkeit und auch Bekanntheit.

Im direkten Vergleich achten PDFBox und iText im Gegensatz zu PDFTextStream nicht auf bestimmte Layouts beim Extrahieren der Dokumente, PDFTextStream ist aber wiederum ab der Verwendung von Multithreads kostenpflichtig und des Weiteren werden bei Letzterem gewisse Zeichen nicht, beziehungsweise nicht korrekt, angezeigt.

---

<sup>1</sup> Open Source, kommt aus dem Englischen und bezeichnet Software, welche einen öffentlichen Quelltext besitzt, sprich von Dritten eingesehen, geändert und genutzt werden darf. Sie kann in den meisten Fällen kostenlos genutzt werden.

<sup>2</sup> Standardisierte Version von PDF (= Portable Document Format) für die richtige Aufbewahrung von Dokumenten auf lange Sicht um diese auch zu späteren Zeitpunkten wieder verwenden zu können.



Die Wahl PDFBox zu verwenden fiel aufgrund der folgenden Vorteile welche PDFBox mit sich bringt:

- Open Source Software, welche kostenlos zur Verfügung steht
- hat einen hohen Bekanntheitsgrad und ist auch dementsprechend gut dokumentiert-
- - beziehungsweise auch getestet

Eventuelle Nachteile von PDFBox wären der richtige Umgang mit den ganzen unterschiedlichen Streams, da sie im Unterschied zu den anderen PDF Analyse Libraries mehrere unterschiedliche Streams zur Verarbeitung benötigt, was jedoch bei einer gewissenhaften Verwendung eher ein kleineres Problem darstellen sollte.

PDFBox ist somit ein geeignetes Tool um PDF Dokumente zu verwalten, die Verwendung beschränkte sich aber im Fall von EasyDocs auf die Extraktion des Inhalttextes, und die der Metadaten. Unter Metadaten sind in diesem Anwendungsbereich Infos wie der Name des Autors und das Erstelldatum gemeint.

Zu Beginn der Diplomarbeit war auch noch eine genauere Analyse des Inhaltstextes zur Beschlagwortung angedacht. Dies wurde jedoch nicht umgesetzt, da das datenbankinterne Ranking inklusive Wertung der Worte diese Aufgabe zu dem angedachten Zweck auf unseren Anwendungsbereich zu unserer vollsten Zufriedenheit bereits erfüllte.

Hier ist ein kleines Anwendungsbeispiel der PDFBox Library um Texte aus einer PDF- Datei zu extrahieren ausgeführt:

```
import java.io.File;
import java.io.IOException;
import org.apache.pdfbox.pdmodel.PDDocument;
import org.apache.pdfbox.text.PDFTextStripper;
public class LesenText {
    public static void main(String args[]) throws IOException {
        //Erstellung einer neuen Datei
        File file = new File("C:/PdfBox_Examples/new.pdf");

        //Erstellt aus der Datei eine PDF und analysiert diese
        PDDocument document = PDDocument.load(file);
        //Initialisieren eines PDFTextStrippers
        PDFTextStripper pdfStripper = new PDFTextStripper();
        //Rueckgabe von Text
        String text = pdfStripper.getText(document);
        //Ausgabe von Text
        System.out.println(text);
        //Schliessen des Objekts
        document.close();
    }
}
```



Bei dem Beispielprogramm ist zu sehen, dass zuerst ein Fileobjekt erstellt werden muss, welches das gewünschte Dokument auf das gewünschte Dokument verweist, anschließend wird dieses Objekt in ein PDF- Dokument (PDDocument) eingelesen.

Dies ist ein Objekt der PDFBox Library und dient dazu eine PDF- Dokument in Java darzustellen. Danach wird diese mittels der Methode `load()` analysiert.

Nach dieser Analyse wird nur noch der Text durch einen `pdfTextStripper`, was ebenfalls ein Objekt der Library ist, extrahiert und durch ein `println()` ausgegeben. Ganz wichtig ist nachdem alles abgeschlossen ist, dass erstellte und verwendeten Dokumentobjekte, beziehungsweise alle verwendeten Streams, wieder vollständig geschlossen und zurückgesetzt werden.

Letzteres hat bei der Durchführung zu einigen Problemen geführt. Da, um die hochgeladenen Dokumente zu analysieren, diese kurz auf den Server zwischengespeichert werden. Nach dem Abarbeiten der PDFs sollten diese gelöscht werden, was jedoch nicht möglich war, da sie laut der Fehlermeldung noch in Verwendung waren und man somit nicht auf diese zugreifen konnte.

Dies ist darauf zurückzuführen, dass bei dem tatsächlichen Code in unserer Diplomarbeit einige unterschiedliche Streams verwendet wurden, da dort ja mehrere Funktionen gegeben sind, was dazu führte, dass das ‚Nicht-Schließen‘ eines dieser vielen Streams, welcher übersehen wurde, zu intensiven Fehlersuchen führte.

Nachdem genauestens darauf geachtet wurde, dass alle verwendeten Ressourcen auch geschlossen wurden, konnte dieser Fehler dann behoben werden.

(PDFBox, Autor: Dirk Natschke, <https://pdfbox.apache.org/>, Aufruf 17.3.2018)



## PoiLibrary für Word Dokumente

Die PoiLibrary wurde verwendet um Dokumente mit der Endung .doc & .docx zu analysieren.



Abbildung 52 POI Logo

Sie ist ebenfalls wie PDFBox eine Apache Library. Apache POI ermöglicht allgemein durch Java Files in Microsoft Office Formaten wie Word, PowerPoint und Excel sowohl lesen als auch erstellen zu können.

Die Entscheidung diese Library zu verwenden wurde aufgrund der Tatsache gefällt, dass ApachePoi im Gegensatz zu seinen Alternativen gratis, weit verbreitet und für die Zwecke in der Anwendung von EasyDocs voll ausreichend war.

Eine Alternative wären zum Beispiel Aspose gewesen, diese ist allerdings kostenpflichtig und fiel somit von Anfang an aus der Auswahl. Weitere Alternativen haben sich aufgrund der Komplexität, die mit der *Handhabung von Microsoft- Produkten*<sup>3</sup> verbunden ist, nicht unbedingt durchgesetzt. Dies hat dann auch Auswirkungen auf die Problembehebung beziehungsweise die Testung im Betrieb wegen der fehlenden Erfahrung in diversen Anwendungsbereichen und fiel somit ebenfalls aus der Auswahl.

Die PoiLibrary umfasst das XML Word Processor Format, kurz XWPF und das Horrible (siehe Index Nr. 5) Word Processor Format, HWPF welche im Bezug auf EasyDocs verwendet wurden.

(Oracle: OracleGroup, <https://poi.apache.org/>, Aufruf 18.3.2018)

---

<sup>3</sup> Auf diese Komplexität ist auch die Namensgebung der POI Library Inhalte zurückzuführen, da die Entwickler zu Beginn des Projektes skeptisch waren, ob beziehungsweise mit welchem Aufwand sie es schaffen würden die eigene Funktionsweise von Microsoft mittels Reverse-Engineering zu entschlüsseln um richtig mit den Dokumenten arbeiten zu können.



### *Extraktion DOCX / DOC Files:*

In EasyDocs wurde wie bereits erwähnt das XWPF verwendet, welches das exakte Lesen und Schreiben von allen Worddokumenten von den Versionen nach Word 97, welche die Endung .docx besitzen, ermöglicht, wobei hier nur das Lesen verwendet wurde. Für ältere Versionen unter Word 97, mit der Endung .doc, wurde das HWPF verwendet.

Bei EasyDocs werden aus den Dokumenten durch die PoiLibrary auch Metadaten wie der Autor des Dokuments und das Erstelldatum ausgelesen.

In folgendem Beispiel wird eine simple Anwendung der POI-Library zur Veranschaulichung gezeigt, bei der Inhalte eines .docx Dokuments extrahiert werden:

```
import org.apache.poi.xwpf.extractor.XWPFWordExtractor;
import org.apache.poi.xwpf.usermodel.XWPFDocument;

public class leseText {

    public static void main(String args[]) throws IOException {

        //Erstellen der benötigten Ressourcen
        FileInputStream fis;
        XWPFWordExtractor oleTextExtractor;

        //Füllen der Ressourcen
        fis = new FileInputStream("C:/POI_Examples/new.docx ");
        oleTextExtractor = new XWPFWordExtractor(new XWPFDocument(fis));

        //Extraktion des Textes
        String text = oleTextExtractor.getText();
        System.out.println(text);

        //Schließen der Ressourcen
        oleTextExtractor.close();
        fis.close();

    }
}
```

Hier kann gesehen werden, dass zuerst ein Inputstream, welcher das Dokument beinhaltet erstellt wird, und dann ein Objekt vom Typ XWPFWordExtractor, welcher dann durch die Methode `getText()` den Text aus dem Dokument holt, erstellt wird.

Bei Dokumenten mit der .doc Endung müsste man ein HWPFDocument- und für das Extrahieren selbst ein WordExtractor- Objekt verwenden.



Wie bei dem zuvor beschriebenen Extrahieren der PDF Files ist auch hier sehr wichtig die verwendeten Streams wieder zu schließen um eine reibungslose Weiterverarbeitung zu garantieren.

#### TXT Format: Simple Extraktion ohne Library

Um mit Textdaten, sprich in diesem Fall Dokumente mit der Endung .txt, zu arbeiten, wurde im Gegensatz zu den anderen Dokumenttypen, welche neben dem Text Metainformationen zur Beschreibung des Textlayouts, der Struktur und der verwendeten Schriften oder ähnlichem enthalten, keine spezielle Library verwendet. (*IT-Handbuch für Fachinformatiker, Galileo Computing*)

Bei der Extraktion der Textdateien wurde somit eine Kombination von Readern verwendet, der BufferedReader da er eine hohe Lesegeschwindigkeit besitzt und über die praktische readLine() Methode verfügt zusätzlich noch der InputStreamReader mit dem FileReader um das richtige Encoding<sup>4</sup> zu erhalten. Wie auch schon bei den Arbeiten an den anderen Dokumenttypen ist auch hier wichtig die verwendeten Ressourcen anschließend wieder schließen.

Da Textfiles dieser Art wie zuvor erwähnt keine Metadaten an sich enthalten wurden hier statt dem Autor des Dokuments der Name des PCs beziehungsweise dessen Nutzer verwendet.

```
public class leseText {  
    public static void main(String args[]) throws IOException {  
  
        File fileDir = new File("C:/Examples/new.txt");  
        BufferedReader in = new BufferedReader(isr);  
        InputStreamReader isr = new InputStreamReader(fis, "UTF8");  
        FileInputStream fis = new FileInputStream(fileDir);  
        StringBuilder sb;  
        String text;  
  
        while ((text = in.readLine()) != null) {  
            sb = new StringBuilder();  
            sb.append(text);  
            System.out.println(sb);  
        }  
        fis.close();  
        isr.close();  
        in.close();  
  
        return sb.toString();  
    }  
}
```

<sup>4</sup> Auch Zeichenkodierung genannt, sie erlaubt die eindeutige Zuordnung von Schriftzeichen und Symbolen innerhalb eines Zeichensatzes. Ein Beispiel wäre hier, dass das Ü auch als dieses angezeigt wird.

Strategy- Pattern:

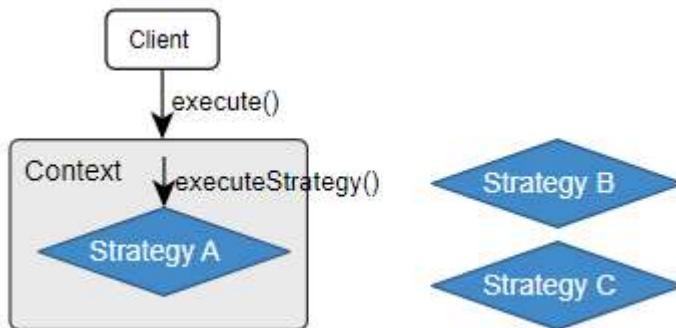


Abbildung 53 Strategy Pattern

(Quelle Bild und Information: Autor: Philipp Hauer, URL:  
<https://www.philippauer.de/study/se/design-pattern.php>, Aufruf: 18.3.2018)

Design Pattern, auch Entwurfsmuster genannt, umfassen allgemein einheitliche, bewährte Lösungswege für immer wiederkehrende Problemstellungen in der Softwareentwicklung. Das Strategy- Pattern wird dann verwendet, wenn für eine und dieselbe Aufgabe zur Laufzeit unterschiedliche Algorithmen verwendet werden sollen, sprich das flexible Wechseln von alternativen Verhalten möglich sein sollte. Dabei ist wichtig, dass die eigentliche Aufgabe selber nicht beeinflusst wird, sondern nur die jeweiligen, zur Aufgabe ausgeführten Teilalgorithmen die von außen zugewiesen werden.

Kurz an dem konkreten Beispiel von EasyDocs würde dies bedeuten, dass das Extrahieren von den unterschiedlichen Dokumenttypen zwar an sich immer erfolgen muss, je nach Typ von diesem aber eine andere Art der Extraktion ausgeführt wird.

Die Vorteile des Strategy- Patterns belaufen sich allgemein darauf, dass die Verwendung von Patterns an sich in Bereichen der Flexibilität, der Wiederverwendbarkeit, und auch der Erweiterbarkeit sowohl als, dass sie einfacher zu verwenden sind als auch dass sie änderungsstabil sind. Des Weiteren, konkret auf das Strategy-Pattern bezogen, können die einzelnen Strategien in dem Kontext erweitert werden, ohne dass sich das Grundkonzept ändert.

Nachteil bei der Verwendung von diesem Pattern ist, dass sich die Anzahl der Klassen beziehungsweise der Objekte erhöht und dadurch auch zusätzliche Schritte durch die Erzeugung und Zuweisung der Strategie entstehen.



Hier ist eine grobe Skizzierung der Anwendung des StrategyPatterns durch ein Klassendiagramm in EasyDocs zu sehen:

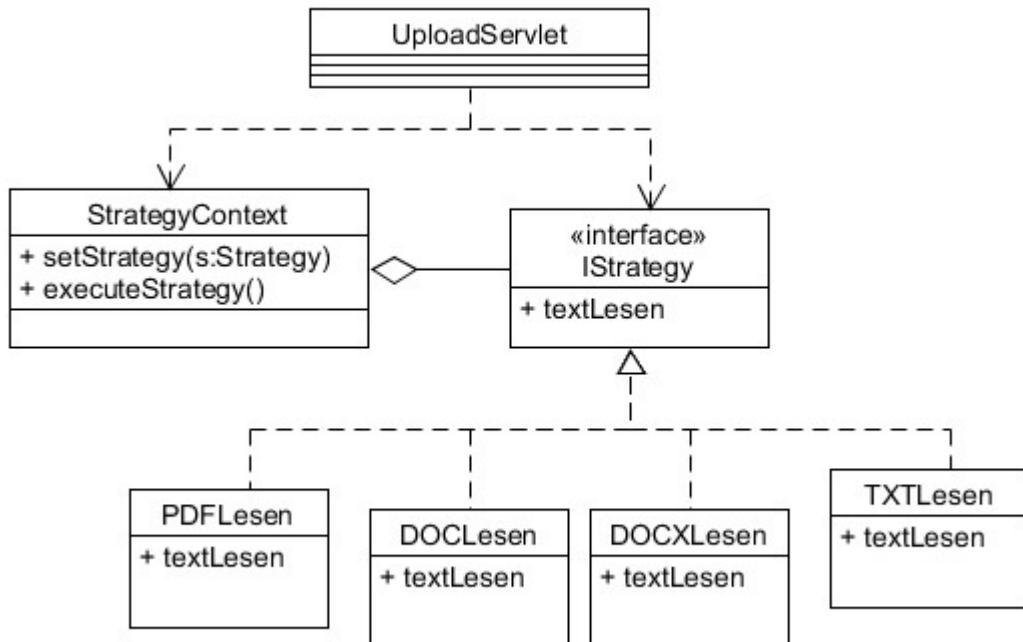


Abbildung 54: Strategy Pattern EasyDocs

Das Interface `IStrategy` beinhaltet wie für Interfaces üblich nur den Methodenkopf der Methode `textLesen()`, alle Klassen die das Interface, also die Strategie implementieren, müssen somit auch diese Methode mitimplementieren.

Die unterschiedlichen Klassen, die jeweils zu einem Dokumenttyp gehören, extrahieren alle, wie zuvor in der Diplomschrift schon beschrieben, mittels für den Typ passenden Libraries, den Inhaltstext. Dies geschieht also in je einer anderen Form, aber immer in der implementierten Methode `testLesen()`.

Zur richtigen Verwendung des StrategyPatterns gehört auch die Implementation eines `StrategyContexts`, dieser dient dazu, die jeweilige Strategie zu setzen, durch die Methode `setStrategy()`, und anschließend auch auszuführen, was mit der `executeStrategy()` Methode ausgeführt wird.

Die tatsächliche Anwendung erfolgte dann im `UploadServlet`, dort wurde durch den `StrategyContexts` nach der Unterscheidung des Dokumenttyps die Strategie gesetzt und ausgeführt.



## Benutzer Registrierung und Login

Um dem Dokumentverwaltungssystem EasyDocs mehr Struktur zu geben und für den jeweiligen Benutzer eine bessere Übersicht zu verschaffen, wurde die ganze Webanwendung benutzerbezogen realisiert.

Dies ermöglicht die exakte Zuweisung der Dokumente zu dem jeweiligen User und damit auch den Zugriff von überall auf diese, auch ohne diese für jeden öffentlich zugängig zu machen, es sei denn dies ist explizit erwünscht.

Dass dieses System auch vollständig von Anfang bis Ende funktioniert, wurde somit eine Registratur-, Login- und ‚Passwort- Vergessen‘-Funktion eingeführt.

### Das Registrieren

Das Serverseitige Registrieren wird wie in Abbildung 34 (*siehe Clientseitiger Registrierablauf*) bereits zu sehen ist, nach der ersten Clientseitigen Kontrolle ausgeführt.

Dass das Registrieren sowohl Client, als auch Serverseitig erfolgt ist äußerst wichtig, da der Erstcheck durch den Client nur grob dafür sorgen kann das gewisse Vorschriften eingehalten werden und keinesfalls die Validierung durch den Server ersetzen können. Der clientseitige Check versucht sozusagen nur, das Absenden von ungeeigneten Daten im Vorfeld zu verhindern. Der Server hat, dadurch dass durch ihn die Daten in die Datenbank eingetragen werden einen direkteren Einfluss darauf, welche Daten dann endgültig eingetragen werden und welche nicht.

Beim Registrieren wird ein neuer Nutzer für EasyDocs angelegt. Um die Korrektheit der Benutzerdaten sicherzustellen, werden die clientseitig eingetragenen Daten serverseitig validiert. (*siehe Abbildung 53*)

Diese Kontrolle umfasst zuerst die Validierung hinsichtlich der Vorgaben, sprich der Username darf zum Beispiel keine Sonderzeichen enthalten und darf eine gewisse Länge weder unter- noch überschreiten, oder auch dass das Passwort mindestens eine Ziffer beinhalten muss und auch länger als 8 Zeichen insgesamt sein muss.

Nachdem die Eingaben auf ihre Richtigkeit geprüft wurden, wird noch kontrolliert, ob die Daten bereits in der Datenbank vorhanden sind, jede Email und auch jeder Benutzername darf nur einmal vorhanden sein.

Nachdem all diese Kontrollen durchgeführt wurden und auch jeweils ein positives Ergebnis zurückgeliefert haben, wird der neue Benutzer in der Datenbank angelegt, abschließend erhält der Benutzer noch eine Bestätigungsemail, dass er erfolgreich registriert wurde.



Dieser Vorgang wurde in folgendem Sequenzdiagramm veranschaulicht:

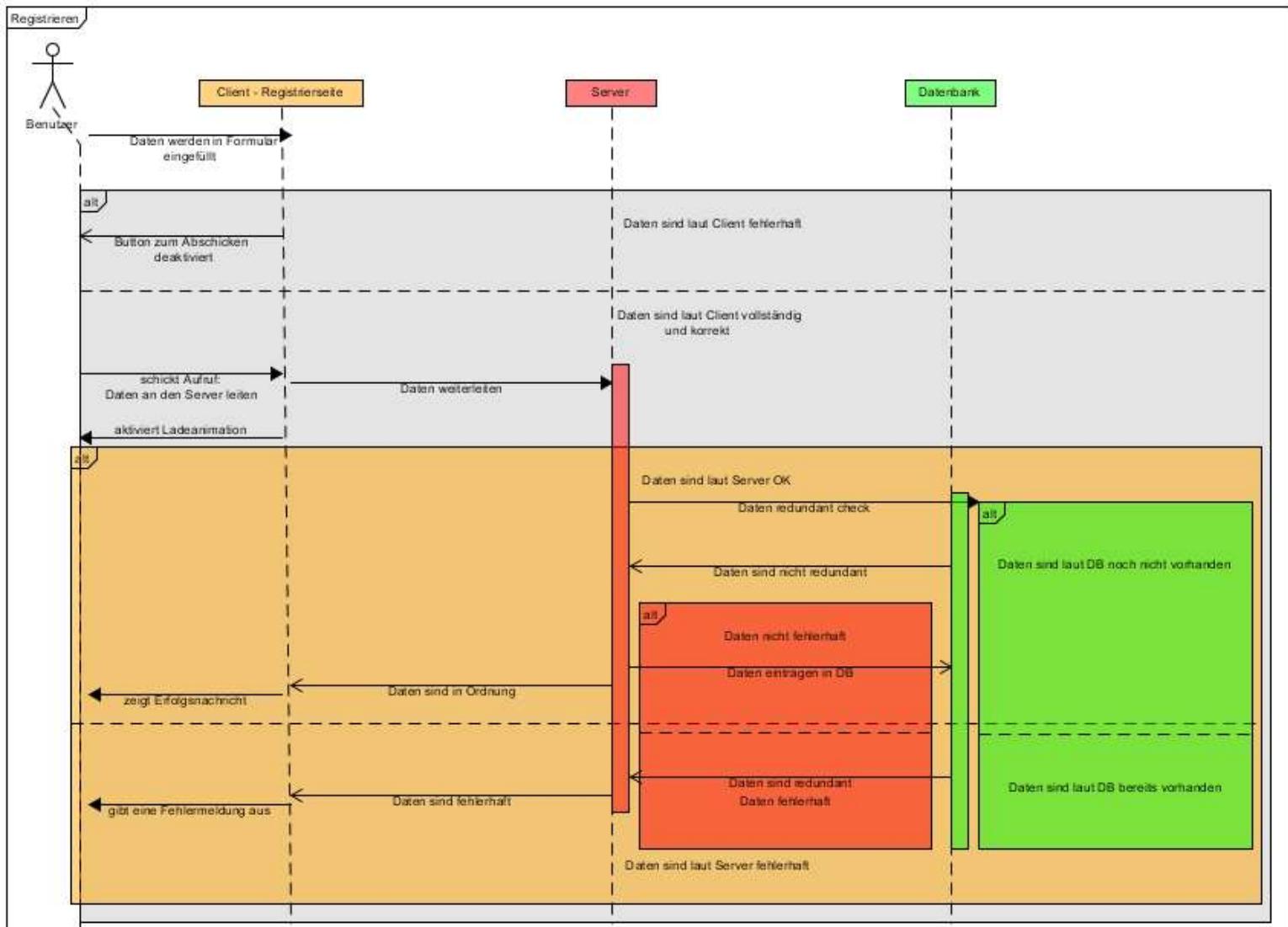


Abbildung 55 Sequenzdiagramm Registrieren

Oben genanntes wurde durch das RegistriserServlet realisiert, es umfasst:

- Kontrolle ob die Eingaben die Voraussetzungen erfüllen
  - Passwörter müssen länger als 8 Zeichen sein und müssen eine Zahl enthalten
  - Benutzernamen dürfen keine Sonderzeichen enthalten und mindestens 3 Zeichen lang sein aber auch nur maximal 20 Zeichen lang sein



- Kontrolle ob die Daten bereits in Datenbank vorhanden sind
  - Keine Email und auch kein Benutzername dürfen doppelt verwendet werden
- Abfangen möglicher Fehler und Rückmeldung an den, Client
  - z.B: Falls Passwort nicht den Anforderungen entspricht oder der Name oder die E-Mail bereits vorhanden sind
- Falls Login erfolgreich war:
  - Eintragen des neuen Nutzers in DB
  - Versenden einer Mitteilung an Nutzer durch Email als Bestätigung zu Registrieren

### *Password Hashen:*

Das vom Benutzer eingegebene Password wird zur Erhöhung der Sicherheit mittels einem Programm gehasht<sup>5</sup>. Dies kann man sich grundsätzlich so vorstellen, dass das Passwort durch einen speziellen Verschlüsselungsalgorithmus, in unserem Fall MD5, verschlüsselt wird.

Dabei wird aus dem als Klartext eingegebenen Wort eine kryptische<sup>6</sup> Prüfsumme erstellt. Hier in der nachstehenden Tabelle sind Beispiele zu sehen welche mittels dem MD5 Algorithmus verschlüsselt wurden.

Tabelle 3 MD5 Verschlüsselung

Klartext	MD5 Hash
Katze	ce6779ff36bd319077abbdb6a81cf7e6
KatzeKatze	c9e1760de0ee09aee956f2d70b270a26
123	202cb962ac59075b964b07152d234b70
eztak	5ccb7149faf17ff6655f2700010ed080
Fischers Fritze fischt frische Fische, frische Fische fischt Fischers Fritze.	755bebb2a99791cdd0c145aec1edabcc

<sup>5</sup> Stammt vom englischen Verb *to hash*, das sich als „zerhacken“ übersetzen lässt, umfasst das Umwandeln einer Eingabemenge auf eine veränderte Zielmenge. → dient dazu, die Daten zu „verstreuen“ beziehungsweise zu „zerhacken“, um sie unerkenntlich zu machen  
(Quelle: Handbook of Applied Cryptography, Autor: A. Menezes, P.)

<sup>6</sup> „unklar in seiner Ausdrucksweise oder Darstellung und daher schwer zu deuten, dem Verständnis Schwierigkeiten bereitend“ – (Duden zur Definitionsbeschreibung, <https://www.duden.de/rechtschreibung/kryptisch> , Aufruf 18.3.2018 )



Das Wort ‚Katze‘ wird unabhängig von seiner Länge im Fall des MD5 Algorithmus auf 32 Zeichen gehasht. Wenn man zwei gleiche Wörter aneinanderhängt, ändert sich die Zeichenfolge von Grund auf, sprich man kann auch mittels dem zuvor gehashten, einzelnen Wort nicht darauf zurückschließen, dass es sich zweimal um dasselbe Wort handelt. Auch wenn die Eingabe weit länger oder kürzer als das gehashte Ergebnis ist, werden es immer 32 Zeichen bleiben.

Dieser Prozess verhindert, dass Passwörter in Datenbanken als Klartext gespeichert und somit direkt eingesehen beziehungsweise kopiert werden können um im Falle eines Angriffes aber auch allgemein die Sicherheit zu erhöhen.

Alleine durch den Fakt, dass Benutzer oft für unterschiedlichste Anwendungen immer die gleichen Passwörter benutzen, macht es dieses Thema noch wichtiger.

### Der Login

Nachdem ein Nutzer registriert wurde, kann er sich mit dem zuvor angegebenen Benutzernamen und dem Passwort anmelden, die Anmeldung erfolgt nach einer serverseitigen Kontrolle. Dort wird überprüft, ob die eingegebenen Daten richtig sind beziehungsweise ob es zu dem Nutzer Einträge in unserer Datenbank gibt.

Falls die Überprüfung positiv ausfällt, hat der Nutzer Zugriff auf die Kernseiten von EasyDocs, auf welchen er dann auch seine Dokumente ablegen, beziehungsweise die bereits abgelegten einsehen, downloaden und löschen kann. Des Weiteren hat er noch Zugriff auf die Seite auf der die gelöschten Dokumente wiederhergestellt werden können.

Beim Anmelden des Benutzers wird serverseitig eine Session erstellt, durch welche es möglich ist einen Benutzerinformationen zur Sitzung zu speichern. Im weiteren Verlauf der Dokumentation folgt dazu später mehr (*siehe Sessions*).

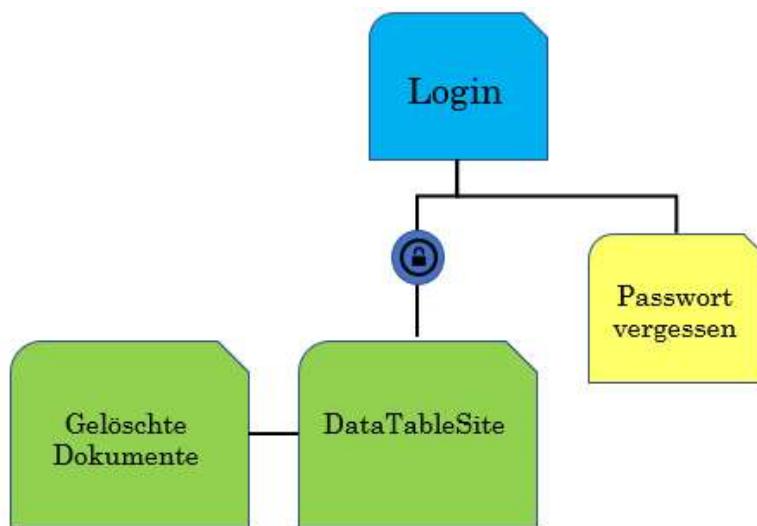


Abbildung 56 Zugriff Login



Das LoginServlet war für die Umsetzung des Logins zuständig, es umfasste folgendes:

- Kontrolle ob Benutzer in DB eingetragen ist
- Kontrolle ob Benutzername und das Passwort übereinstimmen
- Erstellen der Session für jeweiligen Nutzer mit Weiterleitung auf DataTableSeite

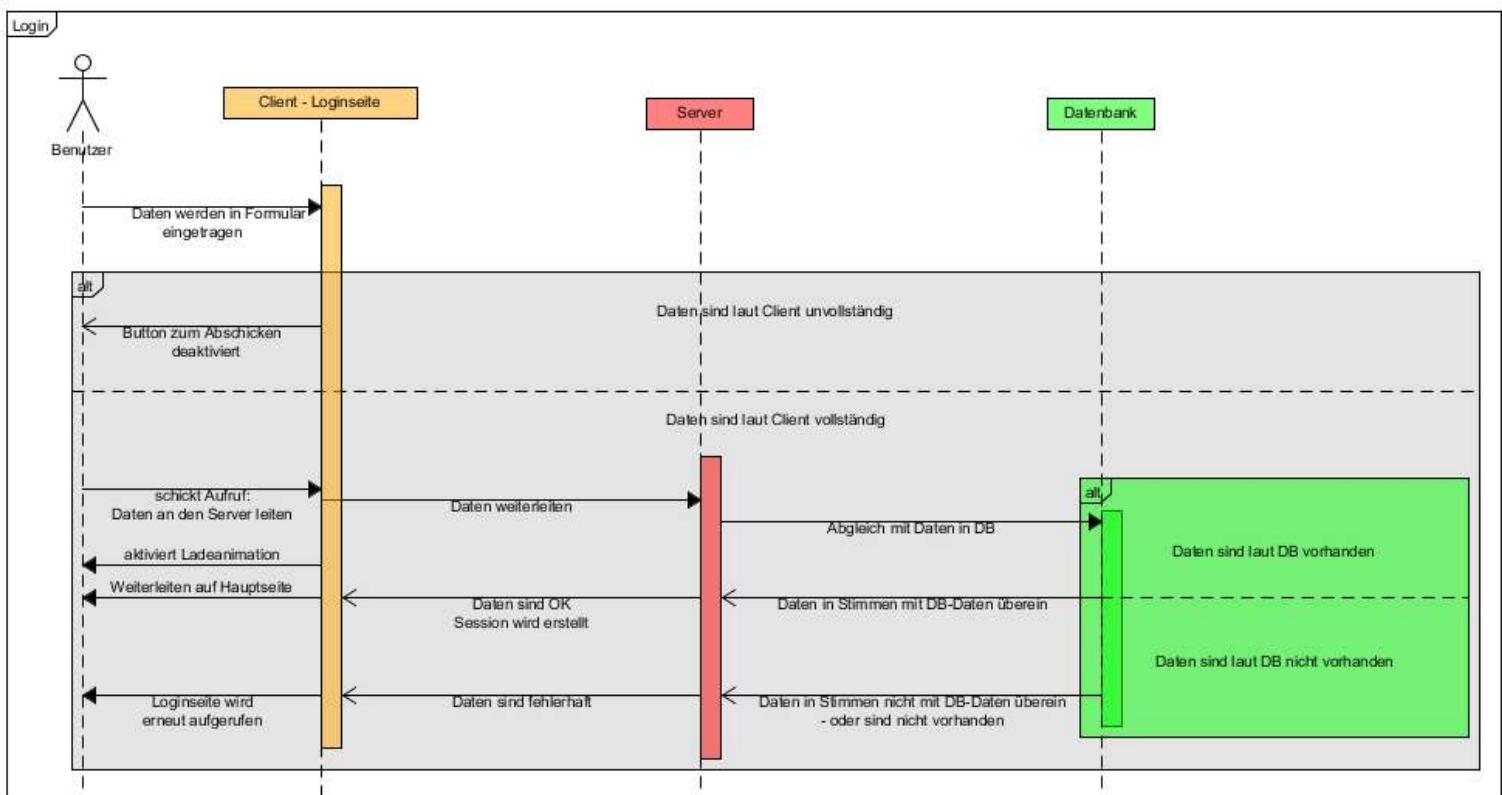


Abbildung 57 Sequenzdiagramm Login

Um den Nutzer auch wieder abzumelden gibt es auch ein LogoutServlet in welchem einfach die zuvor erstellte Session wieder gelöscht und der Nutzer auf die Startseite weitergeleitet wird.

#### Passwort-Vergessen Feature

Um wie oben schon erwähnt das benutzergerechte System vollständig zu machen, wurde das Passwort-Vergessen Feature eingebaut. Dies ermöglicht es dem Benutzer falls er sein Passwort vergessen hat, trotzdem wieder auf seinen Account und somit auf seine Daten zugreifen zu können.

Wenn der Benutzer sich anmelden möchte sein Passwort nichtmehr weiß, gibt es den Link *Passwort vergessen?* unterhalb des Loginbuttons.



Siehe Bild:



Abbildung 58: Passwort vergessen? Feature

Nach dem Klicken dieses Links wird man auf eine Seite weitergeleitet, die ein Eingabefeld, in der man seine zum Account gehörige Emailadresse, angeben kann. Dies wird dann im Hintergrund durch den Server gecheckt und falls die E-Mail zu einem Benutzer gehört, wird auf diese Adresse ein einmalig generierter Link, welcher durch das MD5 Verschlüsselungsverfahren, weiter oben bereits erläutert, generiert wurde, gesendet.

Durch diesen, für den jeweiligen Benutzer eigens generierten Link, wird der Benutzer dann auf eine Seite weitergeleitet, bei der er sein Passwort neu setzen kann. Um die Sicherheit zu garantieren, wird dann beim Aufrufen des Links zusätzlich noch geprüft, ob denn der generierte Hash, der die Seite aufruft auch zu dem Nutzer, welcher eine Passwort- Änderung beantragt hat gehört, da dieser in der Datenbank hinterlegt wurde.

Beim Zurücksetzen werden auch, wie beim Registrieren, Client als auch Serverseitig die Vorgaben, dass es länger als 8 Zeichen und eine Zahl enthalten muss, kontrolliert.

Falls dies alles stimmt erhält man eine Information, dass das Ändern erfolgreich war, anschließend kann man sich wie gewohnt durch die normale Loginseite mit dem neu gesetzten Passwort, welches ebenfalls gehasht wurde, anmelden.

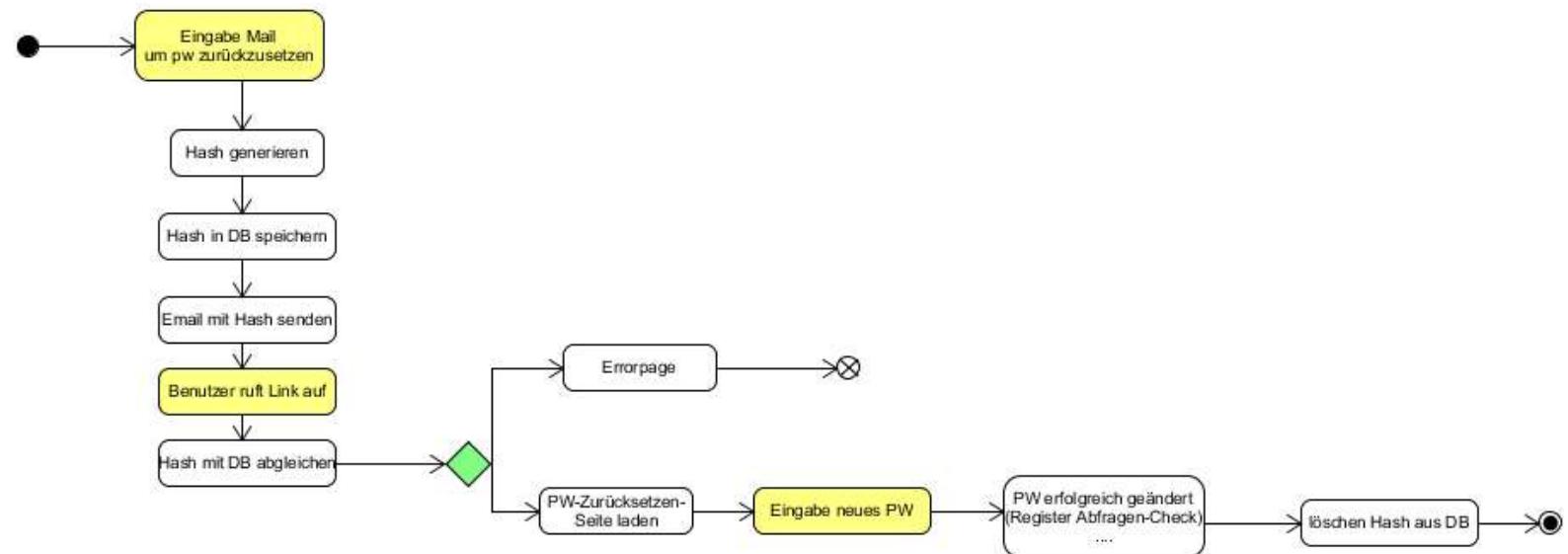


Abbildung 59 Registerablauf

### Grafische Veranschaulichung des Zurücksetzens des Passworts:

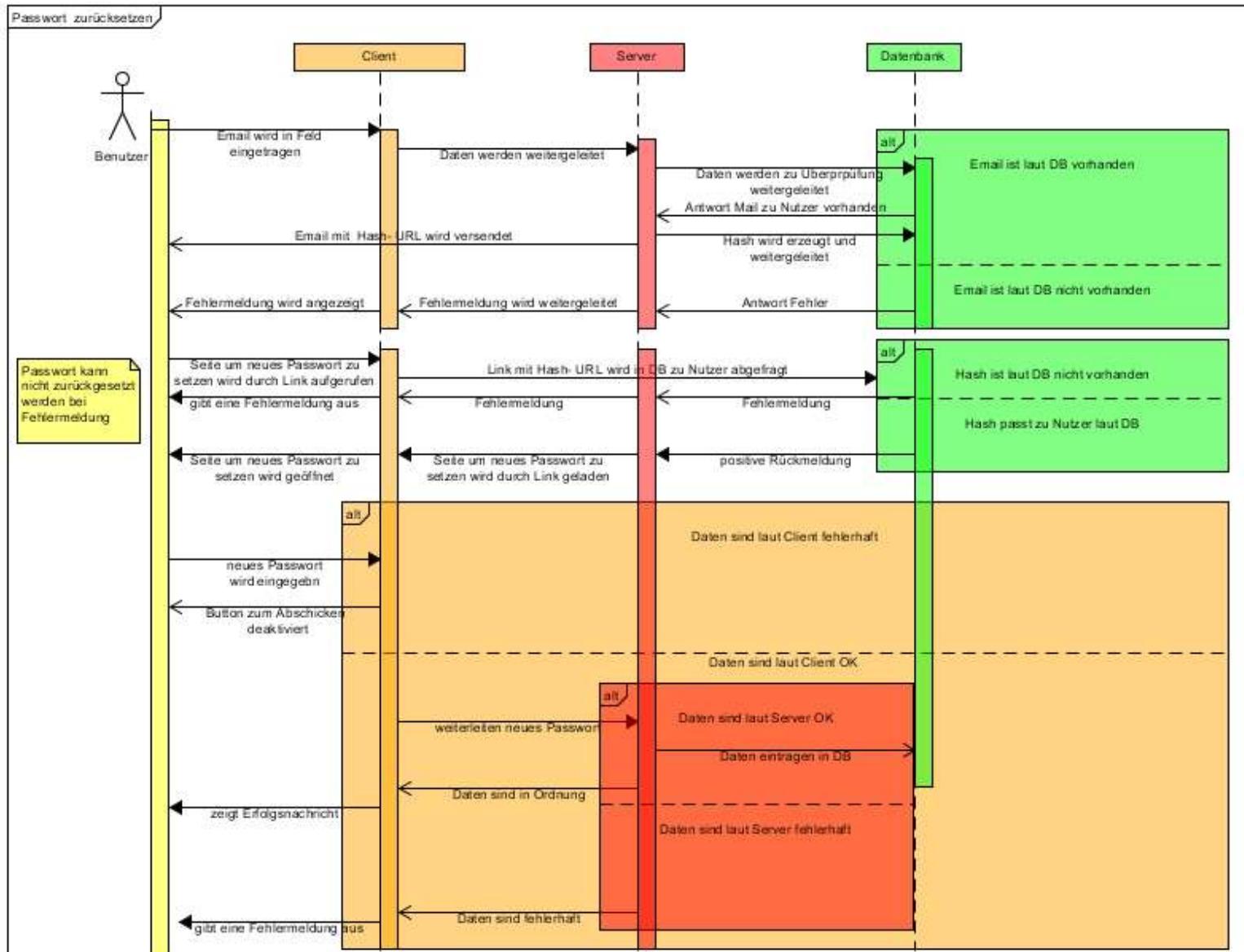


Abbildung 60 Sequenzdiagramm Passwort zurücksetzen

### Emailversenden durch Programm

Um einen persönlicheren Bezug zu den Nutzern von EasyDocs herzustellen, beziehungsweise eine gewisse Kundenbindung zu erstellen wird beim Anmelden, sowohl als auch beim Ändern des Passworts dem Benutzer eine E-Mail gesendet, dies läuft im Hintergrund ab und ist durch die plattformunabhängige JavaMail API<sup>7</sup> von Oracle möglich.

<sup>7</sup> Programmierschnittstelle, genauer auch Schnittstelle zur Anwendungsprogrammierung genannt; API ausgesprochen als Application Programming Interface' (Joshua Bloch: How to Design a Good API and Why it Matters)



Das `javax.mail` Paket beinhaltet Klassen und Methoden zur Verwendung von den gängigsten Emailsystemen und arbeitet daher auch mit den gängigsten Internetstandards wie zum Beispiel MIME, SMTP, welches in dem Fall von EasyDocs angewendet wurde, und einigen weiteren.

SMTP steht für Simple Mail Transfer Protokoll und ist wie schon erwähnt Teil der Internetprotokollfamilie und dient zum Austausch von Emails in Computernetzen, es wird vorrangig zum Einspeisen und Weiterleiten von E-Mails verwendet. Die Kommunikation erfolgt hier zwischen einem E-Mail-Client und einem SMTP-Server.

Hier ist ein allgemeines Programm welches eine simple Email versendet:

```
import java.util.*;
import javax.mail.*;
import javax.mail.internet.*;
import javax.activation.*;

public class SendEmail {

    public static void main(String [] args) {
        String to = "abcd@gmail.com";           // Emailadresse des Empfängers
        String from = "web@gmail.com";          // Ausgangs- Emailadresse
        String host = "localhost";              //Host von dem die Mail gesendet wird

        //setzen der System & Maileigenschaften
        Properties properties = System.getProperties();
        properties.setProperty("mail.smtp.host", host);

        Session session = Session.getDefaultInstance(properties);
        try {
            // Erstellen eines MimeMessage object.
            MimeMessage message = new MimeMessage(session);
            // Setzen des Absenders
            message.setFrom(new InternetAddress(from));
            // Setzen des Empfängers
            message.addRecipient(Message.RecipientType.TO, new InternetAddress(to));
            // Setzen des Betreffs der Mail
            message.setSubject("This is the Subject Line!");
            // Inhaltstext der Mail
            message.setText("Hallo, das ist eine Mail");

            // Senden der Nachricht
            Transport.send(message);
            System.out.println("Email wurde erfolgreich versendet...");
        } catch (MessagingException e) {
            e.printStackTrace();
        }
    }
}
```



## Sessions:

Da HTTP ein ‚stateless‘ Protokoll ist, was so viel bedeutet wie, dass der Client jedes Mal, wenn er eine Webseite aufruft, eine separate Verbindung zum Webserver öffnet, und der Server automatisch keine früheren Clientanforderungen speichert, existieren unterschiedliche Möglichkeiten, um Daten einem Client über mehrere Anfragen zuzuordnen. (Sessions Tutorialspoint: <https://www.tutorialspoint.com/servlets/servlets-session-tracking.html> Aufruf: 16.3.18)

Diese wären:

### Cookies:

Der Webserver ordnet jedem Client bei der Nutzung einer eindeutige Session ID zu und speichert sich diese zu dem jeweiligen Nutzer mit anderen Informationen die für die Session relevant sind, Cookies werden jedoch nicht von jedem Browser unterstützt.

### Hidden Form Fields:

Der Webserver sendet ein verstecktes HTML- Formfeld mit einer eindeutigen Session ID bei jeder Anfrage vom Client an den Server mit. Dies stellt aber keine sehr effiziente Sitzungsverfolgung dar, da immer eine Formularübermittlung stattfinden muss.

### URL- Rewriting:

Hier werden am Ende von jeder URL zusätzliche Daten angehängt um eine Sitzung zu identifizieren. Der Nachteil beim URL- Rewriting besteht darin, dass jede URL dynamisch generiert werden muss, um eine Sitzungs-ID zu übermitteln. Dies verhindert mitunter das Speichern von statischen HTML-Seite im Client Cache.

HTTP Caching ist eine Technik im Hypertext Transfer Protocol (HTTP), um Ressourcen, wie Dokumente, Bilder und Dateien allgemein, anhand bestimmter Kriterien in einem Speicherbereich am Client abzulegen. Dadurch werden unnötige Datenübertragungen und Serveranfragen vermieden um somit die Zugriffszeiten zu verringern.

### Das HttpSession Objekt:

Eine weitere Möglichkeit einen Benutzer über mehrere Seiten zu identifizieren ist das HttpSession Objekt, welches bei EasyDocs zur Identifikation und damit einhergehend Sammlung von Informationen, auf den Benutzer bezogen, verwendet wurde. HttpSession ist eine Servlet- API welche entweder auf Cookies oder dem URL-Rewriting basiert. Das

Handling der ganzen Vorgänge passiert dabei völlig automatisch im Hintergrund.

Dabei instanziert der Servlet Container für jede Sitzung eine HttpSession mit einem einzigartigen Session- Identifier. Um Daten für eine Sitzung speichern zu können, werden ihr mit der Methode `setAttribute()` Informationen zugewiesen, welche als Parameter den Schlüssel und den zugehörigen Wert besitzt, die dann miteinander verbunden werden. Diese Informationen können durch `getAttribute()` wieder gelesen werden. Diese Sitzung ist für einen bestimmten Zeitraum gültig und bleibt für mehr als eine Verbindung oder eine Seitenanforderung des Benutzers bestehen.

Folgendermaßen werden der Session Daten zugewiesen:

```
Session.setAttribute("username", "Name");
```

Um diese dann auszulesen kann der Wert zu einem String gecastet werden:

```
String username = (String) session.getAttribute("username");
System.out.println("Name: "+ username);
```

Hier ist der ganze Prozess noch einmal grafisch veranschaulicht:

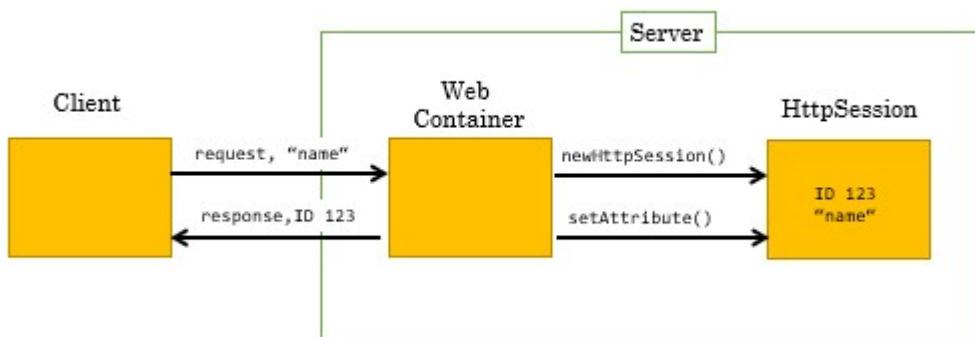


Abbildung 61 HttpSession

Als erstes sendet hier der Client eine Anfrage an den Server, in welchem dann der WebContainer eine neue Session zu dem zugehörigen Attribut, welches vom Client mitgesendet wurde, erstellt. Zusätzlich erstellt er eine eindeutige Kennung, unter der diese Session vorzufinden ist, und schickt mittels dieser dann die dazugehörige Antwort. Bei jeder folgenden Anfrage durch den Client wird diese eindeutige ID mitgesendet, dies erleichtert dem WebContainer dann die Zuordnung zu der Session woher die Anfrage kommt, um eine passende Antwort zu senden.

## Servlets

Servlets sind Java-Klassen, die innerhalb eines Webservers Anfragen von Clients entgegennehmen, was ein *Request* darstellt und auch beantworten, welches die *Response* versinnbildlicht.

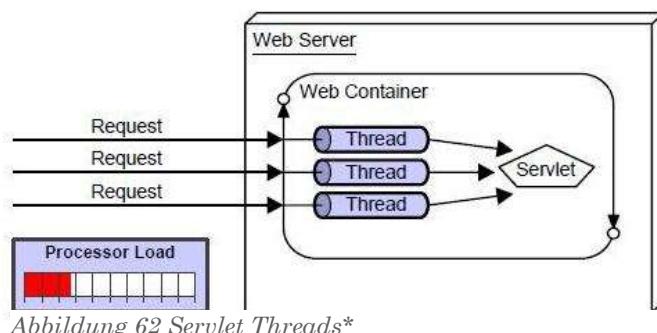
Das Konzept von Servlets wurde um 1994 herum von einem der Begründer von Java entwickelt, als er den ersten vollständig mit Java funktionierenden Webserver erstellte. Kurz danach entwickelte sich daraus die Technik der JSP, da die Verbindung von Visualisierung und Logik Probleme bereitete, auf JSPs wird im Verlauf des Dokuments noch mehr eingegangen (*Siehe JSP*).

Der Begriff Servlet setzt sich aus den im clientseitig arbeitenden Webbrowser-Applets, und den serverseitigen Programmen für Server zusammen.

Dass die Technologie der Servlets funktioniert, muss ein servletfähiger Web- oder Applikationsserver eingesetzt werden. Unter diese fallen alle Java- EE (Enterprise Edition) Server, zu welchen zum Beispiel Apache Tomcat oder Glassfish zählen. EasyDocs verwendet dazu Apache Tomcat, welches einen Servlet Container und einen eigenen HTTP- Server besitzt.

Ein Servlet ist eine spezielle Art von Java- Klasse, welche innerhalb eines Webcontainers, der eine Laufzeitumgebung für diese darstellt, instanziert wird. Der Webcontainer ist des Weiteren auch für die Zustandsverwaltung von den Servlets zuständig, wozu die Kommunikation nach außen, sowie die Verwaltung von dessen Lebenszyklus zählen.

Im Gegensatz zu dem vor Servlets weit verbreitetem CGI, Common Gateway Interface, verwenden Servlets nicht für jede eigene Anfrage einen neuen Prozess, sondern erstellen nur einen neuen Thread. Zusätzlich sind sie auch noch Plattformunabhängig und wesentlich sicherer, da sie in Java programmiert werden und des Weiteren sind sie durch die JVM, Java Virtuel Machine noch robuster.





Servlet Container werden verwendet um serverseitig dynamische Webseiten zu generieren. Sie stellen daher einen Teil eines Webservers dar, der mit dem Servlet interagiert, um die dynamischen Webseiten vom Client zu verarbeiten. Sie sind für unter anderem für das Lebenszyklusmanagement und die Sicherheit zuständig.

Wenn also eine Servletklasse erstellt wird, muss das Servlet Interface implementiert werden, dadurch müssen die `doGet()` und die `doPost()` Methode, welche die zwei wichtigsten HTTP Methoden darstellen, ebenfalls implementiert und im späteren Verlauf überschrieben werden. Um dem Web- Server von dem Servlet mitzuteilen, müssen Metainformationen von dem Servlet in die web.xml Datei, auch Deployment Descriptor genannt, eingetragen werden.

Die erwähnten Get- und Post Methoden behandeln die Requests der jeweiligen Servlets, sie unterscheiden sich dabei in folgenden Punkten:

Tabelle 4 Get / Post Servlet

GET	POST
Die Get- Anfrage hat nur eine sehr begrenzte Kapazität von Daten die mittels ihr gesendet werden können	Post- Anfragen können große Mengen an Daten umfassen
Get- Anfragen sind nicht sicher, da die Daten in der URL enthalten sind → sollte daher nie für sensible Daten verwendet werden	Requests von dem Typ Post sind sicherer da sie nicht in der URL enthalten sind
Gets können als Lesezeichen gespeichert werden	Posts können nicht in Lesezeichen gespeichert werden und deren Inhalte sind nicht in dem Verlauf zu sehen
Gets sind <i>idempotent</i> , was so viel heißt, wie dass nach der ersten Anfrage jede folgende solange ignoriert wird, bis die erste abgearbeitet ist	Posts sind nicht <i>idempotent</i>
Gets sind effizienter und werden daher öfters, falls in Anwendung möglich, verwendet	Posts sind nicht ganz so effizient weshalb sie auch weniger oft verwendet werden



Eine klassische GET- Anfrage würde folgendermaßen aussehen:

.../RegisterPage.jsp?name1=value1&name2=value2

& würde folgende Information beinhalten:



Abbildung 63 GET Methode

Eine klassische Post- Anfrage würde folgendermaßen aussehen, da sie wie bereits erwähnt die Daten im Body der Nachricht versendet:

POST/RegisterPage.jsp HTTP/1.1

Host: www.website.com

name1=value1&name2=value2

und diese Informationen beinhalten:

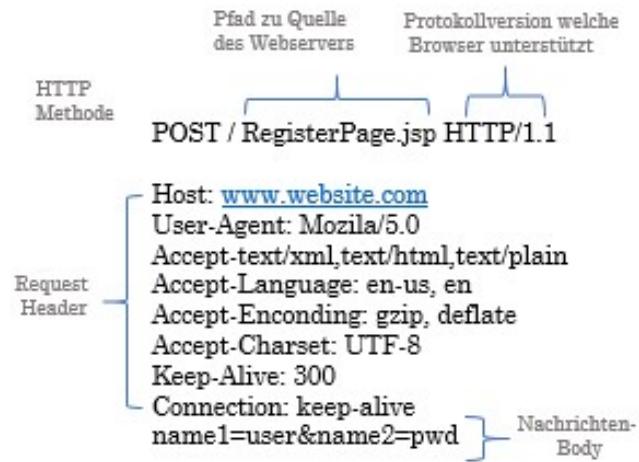


Abbildung 64 Post Methode

Die unter dem Punkt ,Accept‘ erwähnten ContentTypes stellen eine Beschreibung dar, was an den Browser gesendet wird, sodass dieser den Inhalt richtig interpretieren kann.

Die im Kontext zu Servlets, genauer HttpServlets, verwendeten Begriffe Response und Request beziehungsweise auch genauer gesagt die Objekte HttpServletResponse und HttpServletRequest spielen bei der Funktion von Servlets eine Kernfunktion, da diese die Kommunikation zwischen Client und Serverseite ermöglichen. Das Request stellt dabei die Anfrage, als was es auch übersetzt werden kann dar, und die Response, zu Deutsch Antwort, die Rückgabe vom Server an den Client.

Hier ist das Wichtigste von Servlets noch einmal veranschaulicht:

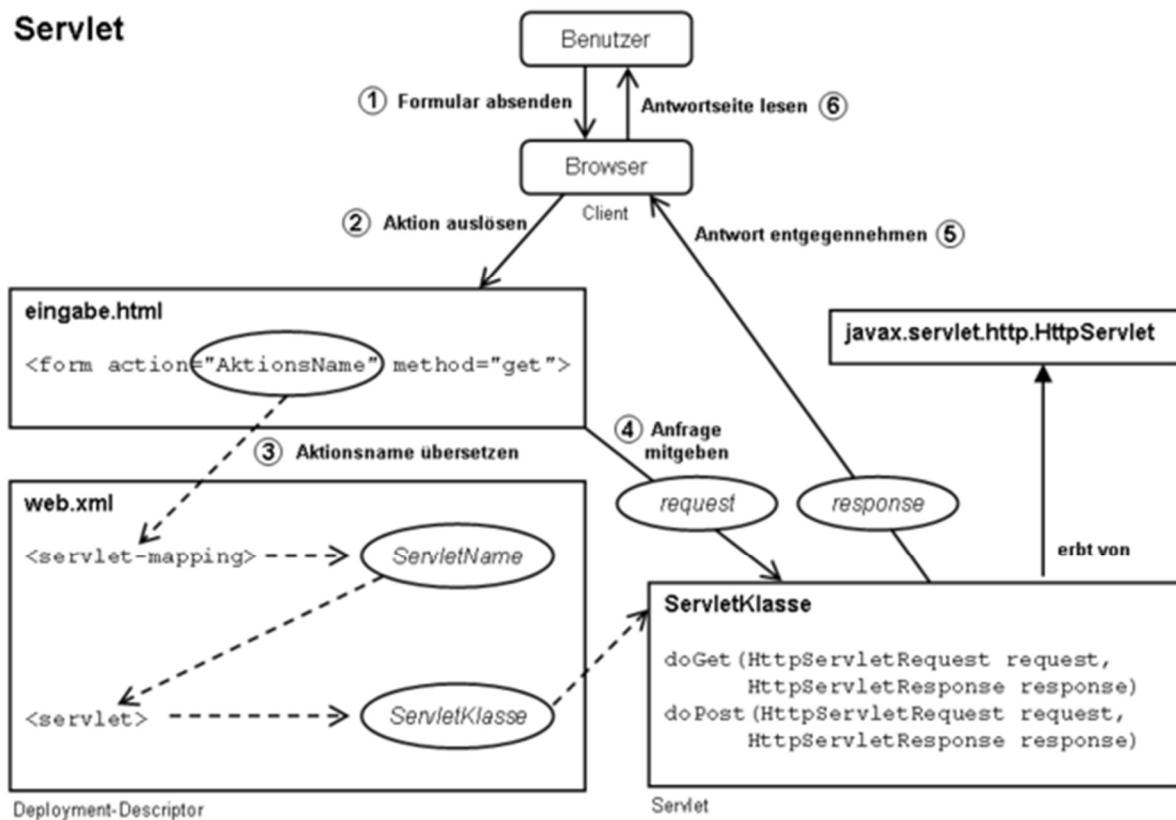


Abbildung 65 Servlet Quelle Wikipedia

(Quelle: Buch *JSP, Servlets, and MySQL* von David Harms)



## UploadServlet

Das UploadServlet stellt eine der Kernfunktionen von EasyDocs dar, durch dieses können die Daten, welche Clientseitig im Browser vom Benutzer zum Hinaufladen ausgewählt wurden, auch wirklich gespeichert und im Bezug auf die Suche und das Sortieren verwendet werden.

Es besteht aus einer Post und einer Get- Methode, wobei die Get nur die Post aufruft, welche die ganze Funktion beinhaltet.

In der Post- Methode werden die von der JSP durch *Requests* angefragten Parameter zu Weiterverarbeitung verwendet, so dass zum Beispiel das hochzuladende Dokument vom Browser in einen InputStream umgewandelt werden kann, ein weiterer Parameter, der dazu dient zu checken, ob denn das Dokument schon vorhanden ist, oder auch um den derzeit angemeldeten Benutzer und daher den Besitzer des Dokuments festzuhalten, sowie den Name der Datei.

Letzteres wäre zum Beispiel folgendermaßen umzusetzen:

```
String dateiname = request.getParameter("dateiname");
System.out.println("Name der Datei: "+dateiname);
```

Hier wird durch das Request das auf der JSP gleichnamige Feld ‚dateiname‘ in einen String umgewandelt und anschließend ausgegeben.

Nachdem also die Kontrolle erfolgt ist, ob die Datei bereits vorhanden ist, wird je nach Dateityp der Inhaltstext, das Erstell- sowie auch das Uploaddatum und der Autor beziehungsweise auch der Benutzer extrahiert und durch im DatenbankManager definierte Methoden in die Datenbank weitergegeben.

Da die unterschiedlichen Textsorten, beziehungsweise dessen Extraktion vom Inhalttext, bereits durch ein StrategyPattern (*Siehe Strategy*) realisiert wurden, mussten diese nur noch ausgeführt werden.

Des Weiteren befindet sich im Servlet noch eine Uploader Methode, die die zuvor erwähnte Datei in Form eines InputStreams in den im web.xml fix angegebenen Pfad, der auf einen eigenen Ordner im Server liegt, speichert. Diese Speicherung dient dazu, dass die unterschiedlichen Methoden zur Verarbeitung des Dokuments dadurch, dass sie direkten Zugriff auf die Datei haben ausgeführt werden können.

Falls die Datei bereits vorhanden ist und der Nutzer diese zu überschreiben wünscht, wird die alte aus der Datenbank gelöscht und die neue eingetragen. Wenn er ihr nur einen neuen Namen geben will, wird die ganze Datei ebenfalls wie eine Neue hinzugefügt und verarbeitet, nur wie erwähnt unter einem anderen Namen.



Nachdem all diese Vorgänge erfolgt sind, ist es ganz wichtig, darauf zu achten, dass die verwendeten Streams, beziehungsweise alle Ressourcen, wie zum Beispiel die Files, auch wieder geschlossen werden, um Probleme bei weiteren Vorgehen zu vermeiden. Auch die zuvor erwähnte, zwischengespeicherte Datei im Ordner des Servers wird ebenfalls wieder gelöscht.

### DownloadServlet

Das DownloadServlet ermöglicht es dem Benutzer, die Daten welche er zuvor hinaufgeladen hat, beziehungsweise ein anderer Benutzer öffentlich zur Verfügung gestellt hat, herunterzuladen.

Es besteht wie das UploadServlet aus einer Get- und einer Post-Methode, wobei die Get die Anfrage an die Post delegiert.

Die Post-Methode bekommt von der JSP den Parameter ‚download‘ durch welchen er ein Gson und ein JsonObject erstellt (*Siehe Verena*) um unter anderem dann den ContentType und den Header der Antwort, der Response, zu setzen.

Um die in der Datenbank abgelegte Datei auszulesen, wird eine in dem DatenbankManager erstellte Methode BLOBAuslesen() (*siehe BLOB*), verwendet, diese gibt die Datei, wie ihr Name schon sagt, in Form einer BLOBDatei weiter. Diese Methode verwendet dabei zwei Parameter, einmal die Connection, um Zugriff auf die Datenbank zu erhalten, und einmal die ID des JsonObject auf welches es zugreifen möchte. Die somit wieder neu erstellte Datei in Form eines einfachen byte[] wird durch den eben schon erwähnten Header (*siehe Thomas Header*), welcher zuvor gesetzt wurde, als das angefragte Dokument mittels der flush() Methode als Servletsoutputstream zurückgegeben.

### JSP in EasyDocs

JSP steht für Java Server Pages, es ist eine von Sun Microsystems entwickelte Webprogrammiersprache, welche dazu dient einfache dynamische HTML und XML Ausgaben durch einen Webserver zu ermöglichen. JSP haben Zugriff auf alle Java- APIs, wozu auch die JDBC API um auf Datenbanken zuzugreifen zu können, zählen. Sie können in folgende Elemente unterteilt werden:

- statischer Inhalt wie HTML
- JSP-Direktiven, sie dienen der Übermittelung von Informationen
- JSP-Skriptelemente
- JSP-Aktionen: Funktionalitäten von Webservern
- JSP-Tag-Bibliotheken, wie Tag- Libraries



Allgemein funktioniert JSP so, dass zuerst eine HTTP-Anfrage durch den Browser an den Webserver gesendet wird. Der Webserver erkennt, dass die Anfrage auf eine JSP bezogen ist und leitet diese an eine JSP-Engine weiter. Wird die JSP das erste Mal aufgerufen, erzeugt die Engine aus der JSP Definition ein Servlet und kompiliert dieses. Das erzeugte Servlet verarbeitet den Request. Dieser Prozess ist in folgender Grafik veranschaulicht (Quelle Bild: [https://www.tutorialspoint.com/jsp/jsp\\_architecture.html](https://www.tutorialspoint.com/jsp/jsp_architecture.html) Stand: 26.3.18):

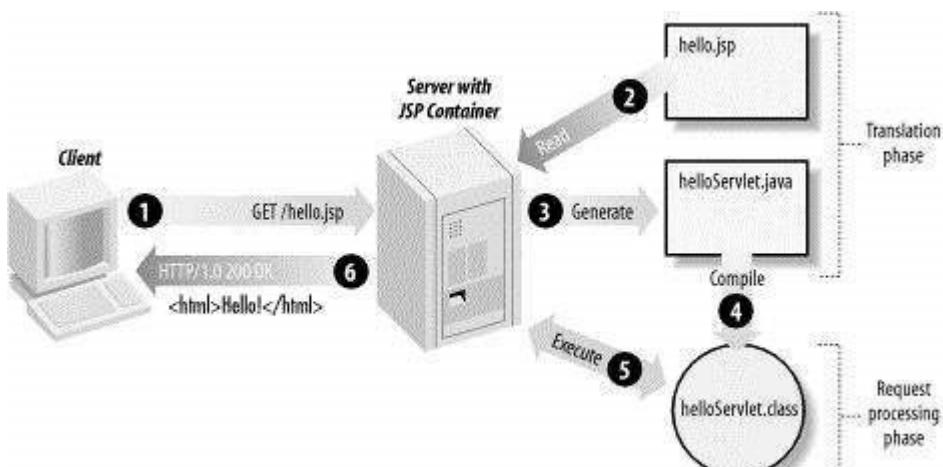


Abbildung 66 Funktionsweise JSP

Im Gegensatz zu den Common Gateway Interface, kurz CGI, welche oft zum gleichen Zweck verwendet werden, bringen sie viele Vorteile mit sich:

- Die Leistung der JSP ist wesentlich besser
- JSP werden vor ihrer Verarbeitung durch den Server kompiliert statt, dass sie wie CGI von neuem durch einen Interpreter und dem Zielskript geladen werden müssen
- JSP sind auf einer Java- Servlets- API aufgebaut, so dass sie wie Servlets auch Zugriff auf alle Java- APIs haben
- JSP-Seiten können in Kombination mit Servlets verwendet werden

JSP und ihre Tags können für eine Vielzahl von Zwecken verwendet werden, wie zum Beispiel, um nur ein paar zu erwähnen, Informationen aus einer Datenbank abzurufen beziehungsweise Benutzereingaben in diese einzufügen, oder um die Steuerung zwischen mehreren Seiten zu ermöglichen und Informationen zwischen Seiten und deren Anforderungen freizugeben.



Bei EasyDocs war der Grund zur Verwendung von JSPs dieser, dass wie schon zuvor erwähnt, im Gegensatz zu einfachen HTML- Seite in einer JSP Java Code ausgeführt werden kann. Um diesen zu verwenden, muss er nur in zwischen den Tags `<% Javacode %>` eingebettet werden.

Konkret wurde durch den Einsatz der JSP bei den jeweiligen Webseiten die Abfrage ob der Benutzer angemeldet, sprich Zugriff auf die angeforderte Seite hat, oder nicht durchgeführt. Falls dies nicht der Fall war, wurde direkt durch eine response auf die LoginSeite verwiesen.

Die Verwendung von JSP wäre durchaus ausbaufähig gewesen, da diese enorme Möglichkeiten in unterschiedlichen Bereichen mit sich bringen, aber auch für den kleinen Teilbereich war es ein schon eine enorme Erleichterung.

### Tomcat

Tomcat ist ein Open Source Webserver und Webcontainer von Apache, der in Java geschriebene Web Anwendungen auf Servlet beziehungsweise JSP- Basis ausführen kann. Er besteht aus drei unterschiedlichen Komponenten, einmal aus dem Servlet-Container Catalina, dann der JSP- Engine Jasper und dem Connector- Framework Coyote.

Die Apache Tomcat Software wurde von einigen der besten Programmierern von der ganzen Welt in einem offenen und in einem für theoretisch Jeden zugänglichem Umfeld entwickelt und wurde unter der Apache License Version 2. veröffentlicht.

Die aktuellste Version von Tomcat ist die 8.5.29, die in der Diplomarbeit verwendete ist 7.0.85, welche unter folgendem Link zu finden ist:

<https://tomcat.apache.org/download-70.cgi> .

Tomcat kann entweder als eigenständiger Testserver verwendet werden, um Servlets und JSPs auszuführen, oder auch als interner Webserver, welcher mit anderen Webservern zusammenarbeitet. Dass Tomcat funktioniert benötigt er ein Java Runtime Enterprise Environment welches zu den JRE 1.1 oder späteren Versionen kompatibel ist.

Des Weiteren ist noch zu erwähnen, dass der Classpath eine wichtige Rolle spielt, da er der Java Virtuellen Maschine, JVM, sagt, wo die Klassen und Packages sich befinden, um das Programm laufen zu lassen, dieser wird automatisch, separat von außen gesetzt und aktualisiert um die Systemunabhängigkeit von Java zu gewährleisten.

Um Tomcat in Eclipse zu starten, muss dieser zuerst als Standardserver bei den Properties eingestellt werden und kann danach wie auf dem untenstehenden





Bild zu sehen ist, ganz einfach gestartet werden. Nachdem er erfolgreich gestartet wurde, je nach Anwendung entweder durch die Konsole oder wie in unserem Fall durch Eclipse direkt, ist er und dessen Inhalt unter <http://localhost:8080/> erreichbar.

Nicht zu vernachlässigen ist, dass gewisse Webanwendungen möglicherweise externe Bibliotheken benötigen um vollständig zu funktionieren. Wenn also eigene Bibliotheken verwendet werden sind diese in den Ordner *lib* im WebContent in dessen Unterordner dem WEB-INF zu hinterlegen. Siehe Abbildung Library WebContainer.

### Ausblick in die Zukunft Sara

Erweiterungsmöglichkeiten im Teilbereich der Steuerung wären zum Beispiel die Implementierung anderer Textsorten, wie zum Beispiel die Open Office Formate, PowerPoint und Excel oder auch die Ablage von Bildern, welche mittels einer weiteren Erweiterungsmöglichkeit und zwar der Beschlagwortung von diesen realisierbar wäre.



Abbildung 68 Library Webcontainer

Die Beschlagwortung würde beim Hochladen der Daten zusätzliche Tags in Form einzelner Worte, die den Inhalt des Uploads beschreiben, umgesetzt werden, diese würden dann auch separat zu dem Dokument beziehungsweise dem File allgemein in der Datenbank abgelegt werden, was das Durchsuchen ein wenig besser gestalten würde, wenn diese Tags mit einer höheren Wertung an die vorhandenen Volltextsuche angeschlossen werden würde.

### Aufgabenbeschreibung Verena - Datenbankprogrammierung

Mein Aufgabenbereich umfasst:

- das Erstellen eines Datenbankdesigns und Umsetzung dieses Designs sowie dessen Veränderungen und Anpassungen an das Projekt
- die Datenbankanbindung mittels Java
- Speicherung der eigentlichen Daten in BLOB-Feldern
- Implementierung der Volltextsuche
- Anzeige der Daten auf Website
- Weitere Features



## Verwendete Technologien und Prinzipien

Postgres

### Allgemeines über PostgreSQL:

PostgreSQL, oder auch kurz Postgres, ist eine objekt-relationale<sup>8</sup> Open Source Datenbank mit einem Relationalen Datenbankmanagementsystem, kurz RDMBS, genauere Erklärung folgt. Postgres wurde am 29.01.1997 zum ersten Mal veröffentlicht, seitdem arbeitet die PostgreSQL Global Development Group an weiteren Entwicklungen und ständiger Verbesserung der Datenbank. Die Lizenzierung von Postgres ist vergleichbar mit der BSD-Lizenz oder der MIT-Lizenz. (Postgres Global Development Group, <https://www.postgresql.org/about/>, 24.03.2018)

Um auf die Datenbank zugreifen zu können und die verschiedenen Operationen wie SELECT, INSERT, UPDATE und DELETE ausführen zu können, verwendet Postgres die Structured Query Language, *siehe SQL*.

Postgres unterstützt verschiedenste Betriebssysteme, dazu zählen Linux, Unix, macOS. Weiters werden verschiedenste Schnittstellen zu Programmiersprachen und Skriptsprachen wie JAVA, C/C++, Perl, PHP, Phyton, Ruby, Lisp, Scheme und in der eigenen Programmiersprache namens PL/pgSQL von Postgres zur Verfügung gestellt.

### Postgres hat folgende Features:

#### **Daten Integrität**

Datenintegrität bedeutet, dass eine Datensicherheit gegeben ist, das bedeutet, dass eine Korrektheit der Daten vorliegen muss.

Um diese Sicherheit gewährleisten zu können, gibt es verschiedenste Constraints und Trigger in Postgres. Ein Constraint, zu Deutsch Zwangseinschränkung, bedeutet, dass Daten nur dann in der Datenbank gespeichert werden dürfen, wenn die vordefinierten Constraints eingehalten werden.

Beispiele von Constraints:

- Primary Key Constraint:  
Pro Tabelle müssen eine oder mehrere Spalten als Primär-Schlüssel festgelegt werden, diese Spalten müssen immer eindeutig sein und diese Werte dürfen nicht NULL sein. Weiteres wird auf diese Felder ein Index

---

<sup>8</sup>

keine direkte objektorientierte Datenbank, sondern übernimmt nur Teilfunktionen der objektorientierten Programmierung



angelegt, dies muss mittels dem Primary Key Constraints überprüft werden.

- **Foreign Key Constraint:**

Wenn eine Verbindung zwischen zwei Tabellen realisiert werden sollte, muss man einen Fremdschlüssel festlegen. Dieser Fremdschlüssel muss der Primary Key der zu verbindenden Tabelle sein.

- **Check Constraint:**

Bei einem CHECK Constraint werden Bedingungen festgelegt, welche erfüllt sein müssen, bevor Daten in die Datenbank gespeichert werden können.

- **NOT NULL Constraint:**

Der Wert in jener Spalte mit dem Constraint NOT NULL darf nie NULL sein.

- **UNIQUE Constraint:**

Der Wert in dieser Spalte muss eindeutig sein, das bedeutet, kein anderer Wert aus dieser Spalte darf denselben Lexem besitzen.

- **DEFAULT Constraint:**

Bei der Definition eines Default Constraint, wird für eine Spalte ein Default-Wert festgelegt. Beim Speichern einer neuen Datenzeile wird, wenn kein Wert für eine Spalte mit Default-Wert angegeben wird, der im Vorhinein festgelegte Wert hineingespeichert.

## Zusätzliche Features

Zur Erzeugung von eindeutigen Spaltenwerten bietet Postgres Sequences an.

Eine Sequence ist ein Nummerngenerator, dabei wird bei jedem Eintrag eine fortlaufende Nummer als ID gespeichert. Werte mit einer Sequence werden automatisch als NOT NULL definiert da eine Sequenz nie keinen Wert besitzt. Bei einer Sequenz kann man folgende Parameter einstellen:

- name: ...
- increment by; bedeutet ob ein Wert ins Positive oder Negative steigt
- minvalue
- maxvalue
- start
- restart
- cache
- cycle
- no cycle
- owned by table.column/NONE
- new\_owner
- new\_schema



- new\_name

### LIMIT/OFFSET:

Ein LIMIT und OFFSET ist zur Begrenzung der Daten beim Auslesen von diesen Daten.

Syntax:

```
SELECT select_list
FROM table_information
[ORDER BY...]
LIMIT { number | ALL } OFFSET number
```

Hierbei kann anstatt des Platzhalters *number*, eine Zahl angegeben werden um die Anzahl an anzuzeigenden Daten bestimmen zu können.

Indexes:

Ein Index ist eine Struktur zur Optimierung der Abfragen in einer Datenbank. Es gibt zwei verschiedene Arten von Indizes: feldbasierende und funktionale Indizes.

### Such-und Sortieralgorithmen

Postgres verwendet GIST (Generalized Search Tree), welches ein öffentliches Netz an verschiedenen Suchalgorithmen zur Verfügung stellt, dazu zählen OpenFTS und PostGIS. Wir verwenden OpenFTS, auf welches im Laufe des Dokumentes noch genauer eingegangen wird.

### Table Inheritance

Die Table Inheritance, zu Deutsch Tabellenvererbung, ist die Weitergabe von vordefinierten Datenstrukturen an unterliegenden Tabellen. Postgres verwendet Single Inheritance, zu Deutsch einfache Vererbung, dabei wird nur aus einer Tabellen geerbt. Sowie auch Multiple Inheritance, zu Deutsch mehrfache Vererbung, dabei wird aus mehreren Tabellen geerbt.



## Relationales Datenbankmanagementsystem

Um eine Datenbank als Relational bezeichnen zu können, müssen bestimmte Voraussetzungen erfüllt sein. (E. F. Edgar, *The Relational Model for Database Management - Version 2*)

Darunter gibt es drei Hauptkriterien für eine relationale Datenbank:

1. Flexibilität:

Um seine Datenbank als flexibel bezeichnen zu können müssen alle Daten in der Datenbank vorhanden sein.

2. Möglichkeit der Bearbeitung der Daten:

Die abgelegten Daten müssen mindestens mit folgenden drei Operationen bearbeitet werden können:

- Selektion:

Bei einer Selektion werden mittels Abfragen bestimmte Datensätze vorübergehend in eine temporäre Tabelle speichert.  
(Ergebnistabelle)

- JOIN:

Durch den Befehl JOIN können zwei oder mehreren Tabellen durch ein namensgleiches oder funktionalgleiches Attribut verbunden werden.

- Projektion:

Bei einer Projektion werden einzelne Spalten, welche eine bestimmte Bedingung erfüllen, vorübergehend in eine temporäre Tabelle speichert. (Ergebnistabelle)

3. Die Beschreibung der Daten in einer Tabelle muss durch die eigenen Daten gegeben sein.



## Die 12 goldenen Regeln der Relationalität

Weitere wichtige Faktoren, um eine Datenbank als relational bezeichnen zu können, werden folgend erläutert, diese nennt man auch die 12 goldenen Regeln der Relationlität von Edgar F. Codd.

### Regel 0: Verwaltung der Datenbank

Die Verwaltung der Datenbank läuft vollständig über das Datenbank-Management- System (DBMS) ab, die Daten werden über die Relation jedes einzelnen Elementes behandelt.

### Regel 1: Darstellung von Information

Alle Informationen einer Datenbank müssen einheitlich und strukturiert abgelegt werden, dazu zählen die eigentlichen Daten, Definitionen von Tabellen und Attributen, Sicherheitsinformationen und vieles mehr.

### Regel 2: Auffindbarkeit von Daten

Alle Daten, welche in einer Datenbank abgespeichert, sind müssen jederzeit auffindbar und identifizierbar sein, durch eine Kombination aus Primär-Schlüssel, Tabellenname und Spaltennamen.

### Regel 3: Behandlung von NULL- Werten

Unter der systematischen Behandlung von Null-Werten versteht man das datentypunabhängige Gleichstellen von nicht eingetragenen Spaltenwerten in einer Datenbank durch den Spaltenwert Null.

### Regel 4: Katalogisierte Datenstrukturen

Unter katalogisierter Datenstruktur versteht man den logischen Aufbau einer Datenbank, welcher ebenfalls abgespeichert werden muss. Somit gibt es einerseits Benutzertabellen, in welchen Daten abgespeichert werden, und Systemtabellen, in welchen Metadaten<sup>9</sup> der Datenbank abspeichert werden.

### Regel 5: Umfassende Abfragesprache

Jede Datenbank muss mindestens eine Abfragesprache zur Verwaltung der Datenbank besitzen. Hierbei sind folgende Funktionen ein Grundbedürfnis:

- Datendefinition
- Datenmanipulation
- Regeln setzen

---

<sup>9</sup> Metadaten sind in diesem Fall Daten, welche Informationen zur Struktur einer Datenbank speichern.



- Rechte vergeben

#### Regel 6: Aktualisierung der Sichten

Eine Aktualisierung aller Sichten und Daten muss ohne Zeit- und Materialaufwand machbar sein.

#### Regel 7: Editierung und Abfragen einer ganzen Tabelle

Die Editierung und das Abfragen von Tabellen sollten ebenfalls vom Datenbankmanagementsystem möglich sein.

#### Regel 8: physikalische Unabhängigkeit der Daten

Das Auslesen der Daten darf nicht von der physikalischen Struktur der Datenbank abhängig sein.

#### Regel 9: logische Unabhängigkeit der Daten

Wenn sich die Definierung von Tabellen ändert, Bsp. Aufspaltung einer Tabelle in zwei Tabellen, dürfen sich die logischen Anwendungen und Zugriffe nicht verändern.

#### Regel 10: Integritätsunabhängigkeit

Datenintegrität besagt die Richtigkeit der Daten. Dabei gibt es zwei Unterteilungen.

Vollständigkeitsintegrität:

Bei der Vollständigkeitsintegrität muss überprüft werden, ob ein Primärschlüssel gegeben ist und ob dieser nicht den Wert Null besitzt.

Beziehungsintegrität:

Bei der Beziehungsintegrität muss überprüft werden ob zu jedem Fremdschlüssel auch ein passender Primär- Schlüssel vorhanden ist.

#### Regel 11: Verteilungsunabhängigkeit

Bei Applikationen sollte nicht darauf geachtet werden müssen, ob eine Datenbank verteilt oder nicht- verteilt ist.

#### Regel 12: Nichtgefährdungsregel

Durch die Nichtgefährdungsregel soll keine Unterscheidung zwischen LOW- LEVEL-Abfragesprache oder HIGH-LEVEL-Abfragesprache vorliegen.



## Warum Postgres

- Postgres ist eine gut entwickelte Datenbank.
- Postgres unterstützt JAVA, Tomcat sowie auch Servlets, welche essentiell für unser Projekt sind.
- Postgres hat eine integrierte Volltextsuche-> ein Kernpunkt
- Postgres unterstützt das Speichern von BLOB-Objekten -> Abspeicherung der Dokumente.

## pgAdmin

PgAdmin ist eine grafische Oberfläche zur Entwicklung und Administration einer Postgres Datenbank PgAdmin ist eine Open-Source-Software, welche Linux, Unix Mac OS X und Windows unterstützt. (pgAdmin Group, <https://www.pgadmin.org/>, 01.01.2018)

Es kann jede beliebige grafische Software zur Datenbankprogrammierung, welche Postgres unterstützt, verwendet werden. Bei EasyDocs wurde zur Testung und Implementierung der Datenbank die Version 1.6 von pgAdmin 4 verwendet.

## Normalisierung einer Datenbank

Die Normalformen oder auch die Normalisierung der Datenbank müssen gegeben sein, um eine relationale Datenbank zu erstellen. Durch Normalisierung und deren Normalformen wird überprüft, ob die Tabellen so aufgeteilt wurden, dass keine redundanten Informationen gespeichert werden, das Vermeiden von Anomalien und die Erstellung einer strukturierten Datenbank. (Ramez Elmasri, Shamkant B. Navathe, Grundlagen von Datenbanksystemen)

### Nullte Normalform:

Definition:

Eine Datenbank kann sich als in der nullten Normalform bezeichnen, wenn alle Daten ohne Strukturierung in einer Tabelle abgespeichert sind. Diese Form der tabellarischen Darstellung von Informationen nennt man auch eine unnormalisierte Tabelle.

Ein gutes Beispiel dafür wäre Excel.



## Erste Normalform:

Definition:

Eine Datenbank kann sich als in der ersten Normalform bezeichnen, wenn alle Attribute einen atomaren<sup>10</sup>, oder auch atomischen, Wertebereich besitzen.

Darunter ist zu verstehen, dass in jeder Spalte einer Tabelle in einer Datenbank nur eine einzige Information gespeichert wird.

Beispiel:

In der Tabelle Benutzer mit den Spalten Name und Adresse müssen diese Attribute, um die erste Normalform zu erfüllen, aufgegliedert werden. Nach dieser Überarbeitung würden statt dem Feld Name die Felder Vorname und Nachname in der Tabelle existieren. Weiters müsste das Feld Adresse ebenso in einzelne Spalten namens Straße, Hausnummer, PLZ und Ort aufgegliedert werden.

## Zweite Normalform:

Definition:

Eine Datenbank kann sich als in der zweiten Normalform bezeichnen, wenn alle Tabellen in der ersten Normalform sind und alle Nicht-Schlüssel Attribute<sup>11</sup> vollständig vom Primär-Schlüssel abhängig sind.

Beispiel:

Bei einer Tabelle namens Rechnung, mit den Informationen zu einem Artikel, einem Kunden und einer Rechnung, müssten diese in drei neue Tabellen, Artikel, Kunde und Rechnung, aufgegliedert werden.

## Dritte Normalform:

Definition:

Eine Datenbank kann sich als in der dritten Normalform bezeichnen, wenn alle Tabellen in der zweiten Normalform sind und alle Nicht-Schlüssel Attribute, welche transitiv abhängig vom Primärschlüssel sind, in einer extra Tabelle ausgelagert werden.

---

<sup>10</sup> Atomar bedeutet, dass eine Einheit nicht noch mehr unterteilbar ist.

<sup>11</sup> Nicht-Schlüssel Attribute sind jene Spalten welche nicht als Teil des Primärschlüssel deklariert sind.



Bespiel:

Man betrachte wieder das erste Beispiel zu der ersten Normalform. Die Tabelle Benutzer mit seiner Adresse, mit den Informationen Straße, Hausnummer, Ort und PLZ. Diese Attribute sind jedoch nicht direkt vom Benutzer abhängig, diese Attribute sind von der PLZ abhängig. Somit wird die Adresse zu einer neuen Tabelle mit den Spalten Hausnummer, Straße, Ort und PLZ.

Insgesamt gibt es fünf Normalformen jedoch spricht man nach der Erfüllung der dritten Normalform schon von einer relationalen Datenbank mit der perfekten Mitte zwischen Performance, Flexibilität und Redundanz.

Aus diesem Grund wurde die Datenbankstruktur von EasyDocs in der dritten Normalform designed.

## SQL

SQL, was für Structured Query Language steht, ist eine Datenbanksprache für relationale Datenbanken. (W3 Group, <https://www.w3schools.com/sql/>, 30.03.2018) Unter anderem unterstützt SQL folgende Operationen:

SELECT Abfragen:

<u>SELECT</u>	Auswahl von Spalten die Benutzer braucht
<u>FROM</u>	Tabellen, die für die Abfrage benötigt werden
<u>WHERE</u>	Filtert Zeilen aufgrund von Bedingungen
<u>GROUP BY</u>	Beeinflusst Funktionen der Aggregatsfunktionen
<u>HAVING</u>	filtert Aggregate der Gruppen
<u>ORDER BY</u>	Ausgabe nach Kriterien sortieren

} optional

Hierbei werden verschiedenste Operationen verwendet:

Tabelle 5 SQL Befehle

Begriff	Erklärung
Selection	Bei dieser Abfrage werden Zeilen mit bestimmten Kriterien ausgelesen.
Projection	Bei dieser Abfrage werden die angegebenen Spalten aus einer Tabelle ausgelesen.
Union	Bei der Union, zu Deutsch Vereinigung, werden zwei oder mehrere Tabellen zu einer Tabelle zusammengefasst. Vorausgesetzt, die Datentypen und Wertebereiche sind ident.



Intersection	Die Intersection-Abfrage, zu Deutsch Schnittstellen, liefert als Ergebnis aller Spalten, welche in den beiden angegebenen Tabellen vorhanden sind.
Minus	Die Minus-Abfrage, oder auch Differenz, liefert als Ergebnis alle Werte, welche nicht in beiden angegebenen Tabellen vorhanden sind.
JOIN	<p>Die JOIN-Abfrage, zu Deutsch Verbindung, verbindet zwei Tabellen unter der Voraussetzung, dass eine Verbindung zwischen den beiden Tabellen gegeben ist, somit ein Fremdschlüssel.</p> <p>Es gibt viele verschiedene Arten von JOINS:</p> <ul style="list-style-type: none"><li>• Cross JOIN</li><li>• Inner JOIN</li><li>• Natural JOIN</li><li>• LEFT OUTER JOIN</li><li>• RIGTH OUTER JOIN</li><li>• FULL OUTER JOIN</li><li>• UNION JOIN</li><li>• SEMI-JOIN</li><li>• SELF JOIN</li></ul>

INSERT Statement:

INSERT INTO (Spaltenname, ...)

Befehls- und Tabellenangabe welche für die Abfrage benötigt wird und Aufzählung der Spalten, in welche ein Wert hineingespeichert werden sollte.

VALUES (Wert, ...)

Angabe der einzuspeichernden Werte

UPDATE Statement:

UPDATE      Befehls- und Tabellenangaben, welche für die Abfrage benötigt wird  
SET              Festlegung der upzudatenden Spalten + neuen Wert  
WHERE          Filtert Zeilen aufgrund von Bedingungen



DELETE Statement:

DELETE FROM      Befehls- und Tabellenangaben, welche für die Abfrage benötigt werden  
WHERE                Angabe einer Bedingung

## JDBC

JDBC, ausgeschrieben auch Java Database Connectivity API, ist der Standard für Datenbankverbindung in der Programmiersprache Java. JDBC wurde grundsätzlich für relationale Datenbanken entwickelt und ist für alle Standarddatenbanken wie MySql, MariaDB, Postgres, etc. verwendbar. ORACLE Group, <http://www.oracle.com/technetwork/java/overview-141217.html>, 01.02.2018)

Vier Hauptaufgaben eines JDBC-Treibers:

- Verbindung zur Datenbank aufbauen
- SQL-Anweisungen senden
- Aufarbeitung der Resultate
- Verbindung zu Datenbank wieder unterbinden

Aufbau eines JDBC-Treibers:

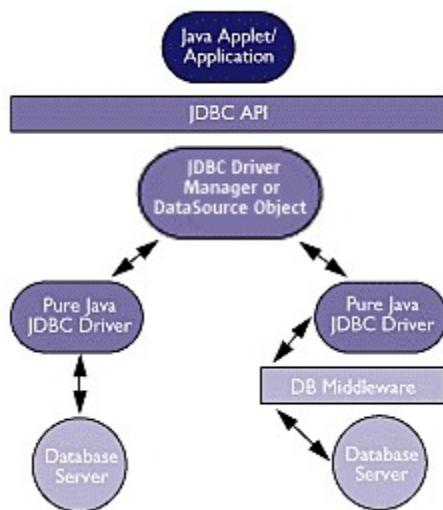


Abbildung 69 Weg der Daten von JAVA Schnittstelle zur Datenbank

Hierbei gibt es zwei verschiedene Wege von der JAVA Schnittstelle zur relationalen Datenbank.



Der direkte Weg:

Die linke Abzweigung der Abbildung 68 stellt den direkten Weg von der JAVA Schnittstelle zur Datenbank dar, hierbei wird die Abfrage mittels dem Netzwerkprotokoll direkt zum DBMS gesendet.

Der indirekte Weg

Die rechte Abzweigung der Abbildung 68 stellt den indirekten Weg dar. Hierbei muss zusätzlich zur JAVA Schnittstelle auf eine weitere Instanz zugegriffen werden, der DB Middleware<sup>12</sup>. Dabei wird die Abfrage mittels dem Netzwerkprotokoll zu der Middleware gesendet und dann mittels dem Vendor's Protokoll zum DBMS gesendet.

Vorteile von JDBC:

- Einfache Nutzung von vorhanden Daten in verschiedenen Datenbankmanagementsystemen
- Vereinfachte Unternehmerentwicklung, es werden viele schwierige Datenzugriffsaktionen im Hintergrund gemacht
- Keine Konfiguration auf Client-PC's notwendig
- Datenbankverbindung wird durch die URL aufgebaut
- Einfache Verfügbarkeit der Library

### PreparedStatement

Das PreparedStatement ist ein Interface vom Typ Statement, welches durch Speicherung der SQL-Abfrage in einem PreparedStatement-Objekt mehrfach ausführbar ist. (ORACLE Group,

<https://docs.oracle.com/javase/7/docs/api/java/sql/PreparedStatement.html>,  
15.03.2018)

Syntax:

Codeteile wurden aus unserem Projekt entnommen.

Als Erstes wird ein Objekt vom Typ PreparedStatement mit einer SQL-Operation als Parameter erzeugt. Die SQL-Operation kann entweder mittels einer Variable oder direkt als String angegeben werden.

---

<sup>12</sup> Die Middleware ist ein Verbindungstück zwischen Backend und Frontend einer Software, welche über ein Netzwerkprotokoll durch die Middleware kommunizieren.



```
String SQL = " update uploaddaten set zustand = 'false', deletedatum = ?,  
status='private' where uploadid = ?";
```

```
PreparedStatement pstmt = new PreparedStatement(SQL)
```

Durch die Methoden `setString`, `setBinaryStream` und `setBoolean` können die einzufügenden Werte in der Insert-Operation richtig gesetzt werden. Als Parameter müssen die Positionen und der Wert angegeben werden.

Mittels dem Befehl `executeUpdate()` wird das PreparedStatement-Objekt an die Datenbank geschrieben.

```
pstmt = conn.prepareStatement(sql);  
pstmt.setString(1, getAktuellesDatum());  
pstmt.setInt(2, uploadid);  
pstmt.executeUpdate();
```

### Warum ist ein PreparedStatement besser als ein Statement?

Ein PreparedStatement kann eine SQL-Injection verhindern.

SQL Injections:

SQL-Injections sind eines der meist- verwendeten Hackerverfahren im Internet. Dabei werden Daten, welche über den Benutzer eingetragen werden, durch eine SQL-Operation erweitert. Die Ziele von SQL-Injections sind eine Datenbank unbrauchbar zu machen oder eine Datenbank komplett zu löschen.

Schutz vor SQL-Injections:

Vergleich von Syntax eines Statements und eines PreparedStatement:

Syntax von einem Statement:

```
PreparedStatement stmt = conn.createStatement("INSERT INTO students VALUES(" + user + "));  
stmt.execute();
```

Syntax von einem PreparedStatement:

```
PreparedStatement stmt = conn.prepareStatement("INSERT INTO student VALUES(?);  
stmt.setString(1, user);  
stmt.execute();
```

Bei einem PreparedStatement werden die Operation und der einzufügende Code getrennt durch verschiedene Protokolle an die Datenbank gesendet. Somit wird der einzusetzende Codeteil nochmals überprüft. Und falls dieser fehlerhaft oder unlogisch erweitert wurde, wird das Statement nicht mehr ausgeführt.

Beispiel einer SQL-Injection:



Benutzer Input wäre:

```
benutzer1
```

SQL-Injection Input wäre:

```
benutzer1 '); DROP TABLE benutzer;--
```

Somit könnte man durch die Eingabe von zum Beispiel Benutzernname und Passwort ganze Tabellen oder die ganze Datenbank löschen.

## ResultSet

Das ResultSet ist ein Interface und wird dazu verwendet, die Datensätze, welche durch ein PreparedStatement herausgelesen werden, anzuzeigen. (ORACLE Group, <https://docs.oracle.com/javase/7/docs/api/java/sql/ResultSet.html>, 15.03.2018)

Syntax:

Zuerst muss ein neues Objekt vom Typ PreparedStatement erzeugt werden, um Daten aus der Datenbank auszulesen und anzeigen zu können. Bei einer Operation mit Ausgabe muss der Befehl executeQuery verwendet werden.

```
String READ_DATEN_PRIVATE="select uploadid,dateityp, dateiname, autor,  
uploaddatum, dokumentdatum, status from uploaddaten where uploader=? and  
zustand='true' order by ? "+reihung+";";  
PreparedStatement pstmt = new PreparedStatement(READ_DATEN_PRIVATE)  
  
pstmt.setString(1, sortierparameter)  
pstmt.setInt(2, spalte);  
  
rs = pstmt.executeQuery();  
  
        while(rs.next())  
        {  
            int uploadid = rs.getInt(1);  
            String dateityp = rs.getString(2);  
            String dateiname = rs.getString(3);  
            String autor = rs.getString(4);  
            String uploaddatum = rs.getString(5);  
            String dokumentdatum = rs.getString(6);  
            String status = rs.getString(7);  
            Daten zeile = new  
Daten(uploadid,dateityp,dateiname,autor,uploaddatum,dokumentdatum, status)  
                DatenSortiertPrivate.add(zeile);  
        }
```



Als Nächstes werden die Parameter ausgelesen. Dafür wird je nach Datentyp des Parameters eine Variable deklariert und durch den Befehl `getDatatype(columnIndex)` aus der Datenbank ausgelesen. Statt dem `Datatype` muss der richtige Datentyp, zum Beispiel bei String Variablen `getString(columnIndex)` angegeben werden.

Weiters zeigt ein ResultSet immer auf genau eine Zeile in der Datenbank, um nun die nächste Zeile auszulesen zu können, muss der Befehl `rs.next()` ausgeführt werden. Wenn jedoch keine Zeile mehr ausgelesen werden muss, schließt sich das ResultSet automatisch, daher kann in dieser Anwendung eine while-Schleife verwendet werden.

Wichtigste Befehle in JAVA:

Daten in Datenbank speichern:

- `PreparedStatement stmt = new PreparedStatement()`
- `pstmt = conn.PreparedStatement(SQL)`
- `pstmt.setDateType(parameterIndex, Wert)`
- `pstmt.executeUpdate()`

Daten aus Datenbank schreiben:

- `ResultSet rs = new Resultset()`
- `PreparedStatement stmt = new PreparedStatement()`
- `pstmt = conn.PreparedStatement(SQL)`
- `pstmt.setDateType(parameterIndex, Wert)`
- `pstmt.executeQuery()`
- `rs.next()`
- `rs.getDateType(columnIndex);`

### Volltextsuche von Postgres

Die Volltextsuche ist eine komplexe und umfangreiche Aufgabe. Es müssen viele Parameter analysiert und betrachtet werden. Für EasyDocs wurde die Volltextsuche von Postgres verwendet, da hierbei die wichtigsten Faktoren zu einer leistungsfähigen und schnellen Volltextsuche implementiert sind. (Rachid Belaid, <http://rachbelaid.com/postgres-full-text-search-is-good-enough/>, 25.02.2018)

Die Volltextsuche besteht aus drei Teilen, das Vereinfachen der Texte, das Durchsuchen der Texte und das Ranking der Texte in Bezug auf das gesuchte Wort.



## Vereinfachung der Texte

Hierbei wird die Methode to\_tsvector verwendet, welche Texte und einzelne Wörter nachfolgenden Aspekten vereinfacht:

- Groß- und Kleinschreibung ist nicht relevant, alles wird kleingeschrieben
- Manche Vor- und Nachsilben werden weggelassen
- Wörter wie:
  - Artikel
  - Häufig verwendete Verben wie ist, sein, haben, werden,...
  - Personalpronomen
  - Füllwörter wie auch, dazu, zwar, ...werden weggelassen
- Mehrzahl wird zu Einzahl
- Oft werden Wortenden abgeschnitten, um die Wörter zu vereinfachen, z.B.: interpretieren-> interpreti

Zusätzlich werden die Wörter alphabetisch angeordnet und mit der Position im Text, welche als Zahl angegeben wird, aufgelistet, siehe Abbildung 69.

Beispiel:

```
SELECT to_tsvector('Eine meiner Leidenschaften ist die Gartenarbeit.');
```

Ergebnis:

	to_tsvector	tsvector
1	'gartenarbeit':6 'leidenschaft':3	

Abbildung 70 Ergebnis zum Veranschaulichen der Vereinfachung der Texte

Wie man an diesem Beispiel gut erkennen kann, werden alle Artikel, Personalpronomen und häufig verwendete Verben entfernt und die zwei wichtigen Nomina, welche Gartenarbeit und Leidenschaften lauten, nochmals vereinfacht durch Weglassung der Groß- und Kleinschreibung und des Plurals.

Simple Vereinfachung:

Bei dem Befehl to\_tsvector kann ein weiterer Parameter angegeben werden, welcher simple lautet. Dabei werden lediglich alle eingegebenen Wörter klein geschrieben und alphabetisch geordnet und mit der ursprünglichen Position im Satz bzw. Text angezeigt.

Beispiel:

```
SELECT to_tsvector('simple','Eine meiner Leidenschaften ist die Gartenarbeit.');
```



Ergebnis:

	to_tsvector
	tsvector
1	'die':5 'eine':1 'gartenarbeit':6 'ist':4 'leidenschaften':3 'meiner':2

Abbildung 71 Ergebnis zum Veranschaulichen der simplen Vereinfachung der Texte

Diese Art der Vereinfachung wird oft zum Vereinfachen von Autoren oder Dateinamen.

#### *Durchsuchen der Texte*

Mit den Operatoren @@ kann man Texte durchsuchen.

Beispiel:

```
select ('Meine Leidenschaft ist die Gartenarbeit.') @@ 'Leidenschaft';
```

Ergebnis:

	?column?
	boolean
1	true

Abbildung 72 Ergebnis der Durchsuchung von Texten

Da dies jedoch oft zu einem verfälschten Ergebnis führen kann, wenn man ein Suchwort nicht genau angibt vereinfacht man sowohl den Text als auch das angegebene Suchwort. Mit tsquery wird ein Wort zum gleichen Lexem umgewandelt wie bei to\_tsvector.

Beispiel:

```
SELECT to_tsquery('Leidenschaften');
```

Ergebnis:

	to_tsquery
	tsquery
1	'leidenschaft'

Abbildung 73 Ergebnis der Vereinfachung von einzelnen Wörtern mittels to\_tsquery



Vergleich to\_tsvector:

```
SELECT to_tsvector('Leidenschaften');
```

Ergebnis:

	to_tsvector
	tsvector
1	'leidenschaft':1

Abbildung 74 Ergebnis der Vereinfachung von einzelnen Wörtern mit to\_tsvector

Wie man an diesem Beispiel erkennen kann, wird es zu demselben Wort vereinfacht, aber eine Information fällt weg, nämlich wo die Ursprungsposition im Satz bzw. Text ist.

#### Weitere Information:

Postgres Fulltextsearch unterstützt verschiedene Sprachen, wie Dänisch, Niederländisch, Englisch, Finnisch, Französisch, Deutsch, Ungarisch, Italienisch, Norwegisch, Portugiesisch, Rumänisch, Russisch, Spanisch, Schwedisch und Türkisch.

Um die Sprache zu setzen, werden folgende Befehle verwendet:

```
ALTER TABLE Uploaddaten ADD LANGUAGE text NOT NULL DEFAULT('german');
```

Dabei wird eine neue Spalte definiert und wenn keine Sprache vordefiniert wird beim Hinzufügen einer neuen Zeile, wird automatisch durch den Befehlteil DEFAULT('german') die Sprache Deutsch eingetragen. Ansonsten wird die Sprache, welche bei der Spalt language angegeben wird, hineingespeichert.

Ebenfalls werden auch Akzente vereinfacht und unterstützt, dies ist besonders wichtig, wenn man mit der Sprache französisch arbeitet.

Ein Beispiel wäre:

```
CREATE EXTENSION unaccent;
SELECT unaccent('éèêàùûîôœç');
```

Abbildung 75 Eingabe von Methode zum Vereinfachen von Akzenten

	unaccent
	text
1	eeeauuiooec

Abbildung 76 Ergebnis der Vereinfachung von Akzenten



## Ranking von Postgres

Um Dokumente nach ihrer Relevanz anzurufen, müssen Wertungen gesetzt werden.

Postgres stellt hierbei zwei Arten des Rankings zur Verfügung:

- `ts_rank()`
- `setweight()`

### Die Methode `ts_rank`:

`ts_rank` ist eine Methode welche von Postgres bereitgestellt wird, dabei wird die Relevanz des Suchwortes in Bezug auf das Gesamtdokument berechnet und mittels einer Dezimalzahl dargestellt.

Beispiel:

```
select ts_rank(to_tsvector('Heute ist ein sehr schöner Tag.'),  
to_tsquery('Tag'))as relevancy;
```

Ergebnis:

	relevancy
1	0.0607927

Abbildung 77 Ergebnis der Wertung der Relevanz eines Wortes in Bezug zum Text

```
select ts_rank(to_tsvector('Heute ist ein sehr schöner Tag, an diesem  
wunderschönen Tag mache ich einen Spaziergang.'), to_tsquery('Tag'))as relevancy;
```

Ergebnis:

	relevancy
1	0.0759909

Abbildung 78 Ergebnis der Wertung der Relevanz eines Wortes in Bezug zum Text

An diesen beiden Beispielen kann man gut erkennen, dass die Relevanz sowohl von der Länge als auch von der Anzahl in einem Text liegt.

### Die Methode `setweight`:

Mit der Methode `setweight`, welche ebenfalls von Postgres bereitgestellt wird, können verschiedene Gewichtungen auf verschiedene Parameter gesetzt werden, mittels der Markierung von A bis D. Attribute, welche mit A markiert wurden, zählen am meisten. Attribute, welche mit dem Wert D markiert wurden, werden am wenigsten gewichtet.



Die Methode setweight kann nicht direkt ausgeführt werden, dies ist nur in Zusammenarbeit mit einer Abfrage oder Sortierung oder Durchsuchung von Daten möglich.

Syntax:

```
Setweight((Parameter/Attribute/Vereinfachter Text), 'A')
```

## JSON

JSON ist eine Open-Source Google Library, um Daten in JSON zu serialisieren und zu deserialisieren, damit diese Daten an weitere Programme weitergeleitet werden können. (Sandeep Kumar Patel, Instant JSON)

Die Serialisierung ist jener Vorgang, bei welchem ein Objekt in eine bestimmte Form konvertiert wird, um die Informationen dann zu erhalten und weiterleiten zu können.

Das Gegenteil ist die Deserialisierung des Objektes, dabei wird das Objekt wieder konvertiert, um die Informationen wieder lesen und weiterverwenden zu können.

Ablauf:

- Erstellen eines JSON-Objektes
- Erstellung eines JSON-Stringes, dieser JSON-String muss eine genaue Form besitzen.
- über JSON-Objekt wird JSON-String hinausgeschrieben

NORMAL:

```
TestBean bean = new TestBean();
json0.put("bean", bean);

System.out.println("OUTPUT - perHand:");
System.out.print(json0.toString());
```

JSON:

```
TestBean bean = new TestBean();
String out = gson.toJson(bean);

System.out.println("OUTPUT - per JSON Object:");
System.out.println(out);
```



## JSON

JSON steht für Java Script Object Notation und ist ein Format, welches Java Script Objekte textuell darstellt.

JSON-Struktur:

Ein JSON String besteht aus vielen Wertepaaren, ein Wertepaar besitzt einen Namen und den dazugehörigen Wert.

Beispielcode aus der Diplomarbeit:

Man kann diese Struktur händisch anfertigen:

```
antwort = "{\"draw\":\"+draw+\", \"recordsTotal\":"+anzahl+", \"recordsFiltered\":"+anzahl+", \"data\":[{\"ID\":\""+daten.get(i)[0]+\"", \"DateiTyp\":\""+daten.get(i)[1]+"\", \"Name\":\""+daten.get(i)[2]+\"", \"Uploader\":\""+daten.get(i)[3]+\"", \"Autor\":\""+daten.get(i)[4]+\"", \"UploadDatum\":\""+daten.get(i)[5]+\"", \"DokumentDatum\":\""+daten.get(i)[6]+\"", \"ZUGANG\":\""+daten.get(i)[7]+"\"]}]};
```

Der unterstrichene Teil zeigt die jeweiligen Werte-Paare.

JSON String mittels JSONObject:

```
JSONObject test = new JSONObject();
test.addProperty("ID", daten.get(i).getUploadid());
test.addProperty("DateiTyp", daten.get(i).getDateityp());
test.addProperty("Name", daten.get(i).getDateiname());
test.addProperty("Autor", daten.get(i).getAutor());
test.addProperty("UploadDatum", daten.get(i).getUploaddatum());
test.addProperty("DokumentDatum", daten.get(i).getDokumentdatum());
test.addProperty("ZUGANG", daten.get(i).getStatus());

data.add(test);
```

Hierbei werden die Paare mittels einer ArrayList zusammengesetzt und als JSONObject hinausgeschrieben.

Vorteile:

- Für Mensch einfach zu schreiben
- Für Maschine einfach zu parsen
- Einfache Implementierung und Handhabung



## Speicherung der Dokumente durch BLOBs

Erklärung:

BLOB, Basic Large Object oder auch Binary Large Object ist ein großes binäres Datenobjekt. Ein BLOB Feld kann nicht gesucht, gefiltert oder sortiert werden. (Narayan Prusty, <http://qnimate.com/an-introduction-to-javascript-blobs-and-file-interface/>, 02.02.2018)

Verwendung:

BLOB-Dateien können in Postgres auf zwei verschiedene Arten abgespeichert werden. Entweder mittels dem Datentyp bytea, welcher das BLOB-Objekt direkt speichert, oder mittels dem Datentyp oid, Object Identifier, welcher auf das BLOB-Objekt referenziert.

Die wichtigsten Befehle dazu sind:

- getBytes()
- setBytes()
- getBinaryStream()
- setBinaryStream()

Gründe für die Abspeicherung der Dokumente als BLOB:

Die Speicherung der Dokumente als BLOB wurde verwendet, da das Dokument dann leicht erhältlich ist und es eine weitverwendete Methode ist, welche gut getestet ist.

## Github

Github ist ein Open Source Onlineversionsverwaltungssystem, dessen grafische Ausführung auf einen webbasierten Online-Dienst zurückzuführen ist.

Zusätzlich ist Github ein Filehosting-System, in welchem die verschiedenen Versionen von einem Programmcode abgespeichert werden. Entwickelt wurde es von Chris Wanstrath, PJ Hyett und Tom Preston-Werner. Ursprünglich wurde Github in Ruby und Rails programmiert, 2009 stieg man auf den Webhoster Rackspace um. (John D. Blischak, Emily R. Davenport, Greg Wilson, A quick Introduction to Version Control with Git and GitHub)

Das Motto von Github lautet: build software better, together



## Befehle GitHub:

Tabelle 6 Git- Befehle

Befehl	Wirkung
git init	erzeugt eines Repository im lokalen Verzeichnis
git status	Ausgabe über den aktuellen Status des Repository, zeigt alle Änderungen seit dem letzten commit
git add <dateien>	übernimmt folgende/angegebene Dateien in den zu commiteden Status
git commit-m“Nachricht”	erzeugt eine neue Revision mit den Änderungen
git diff	Ausgabe über aktuelle Änderungen
git log	Ausgabe der Revisions-Historie mit allen commit-Messages
git checkout	springt zu einer bestimmten Revision oder Verzweigung
git clone	klont das angegebene Repository in ein lokales Verzeichnis
git push	Schiebt die neuesten commits in das remote Repository
git pull	holt die Änderung des remote Repository unter der Voraussetzung das ein Repository geklont wurde
git ignore	Festlegung der Klassen welche nicht gepusht werden
git help	zeigt die 20 meist verwendete Befehle und wie man diese verwendet

## Repository:

Um in Github ein neues Projekt anlegen zu können, muss zuerst ein Repository erstellt werden. Das Repository ist jener Bereich, in welchem der eigentliche Programmcode steht bzw. alle dazu notwendigen Files.

Es gibt zwei Arten von Repositoryen:

- Private Repository → nur für zahlende Benutzer
- Public Repository → jeder kann sich dieses Repository ansehen



Github in Java:

Repositorien in Java verwalten:

In dem Ansichtsfenster ‚Git Repositories‘ von Eclipse können Befehle zum Erstellen eines neuen Repository, Klonen oder Hinzufügen eines bestehenden Repository ausgeführt werden.

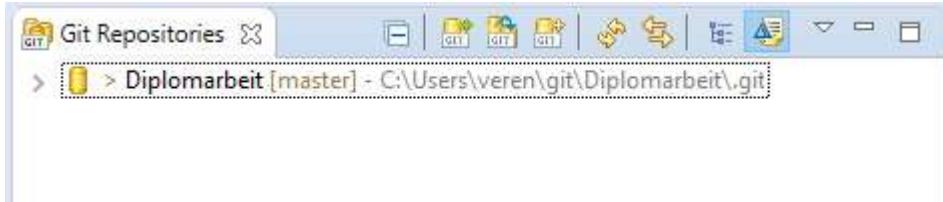


Abbildung 79 Anzeige und Verwaltung von Repositoryen in JAVA

In dem Ansichtsfenster Git Staging‘ von Eclipse können Befehle wie *git add* und *git commit* ausgeführt werden.



Abbildung 80 Verwaltung von einem Repository in JAVA

Weitere Befehle zur Kommunikation mit der Versionsverwaltungssoftware wie *git push*, *git commit*, etc. können mittels Rechtsklick auf das Projekt im Unterpunkt Team ausgeführt werden, siehe Abbildung 80.

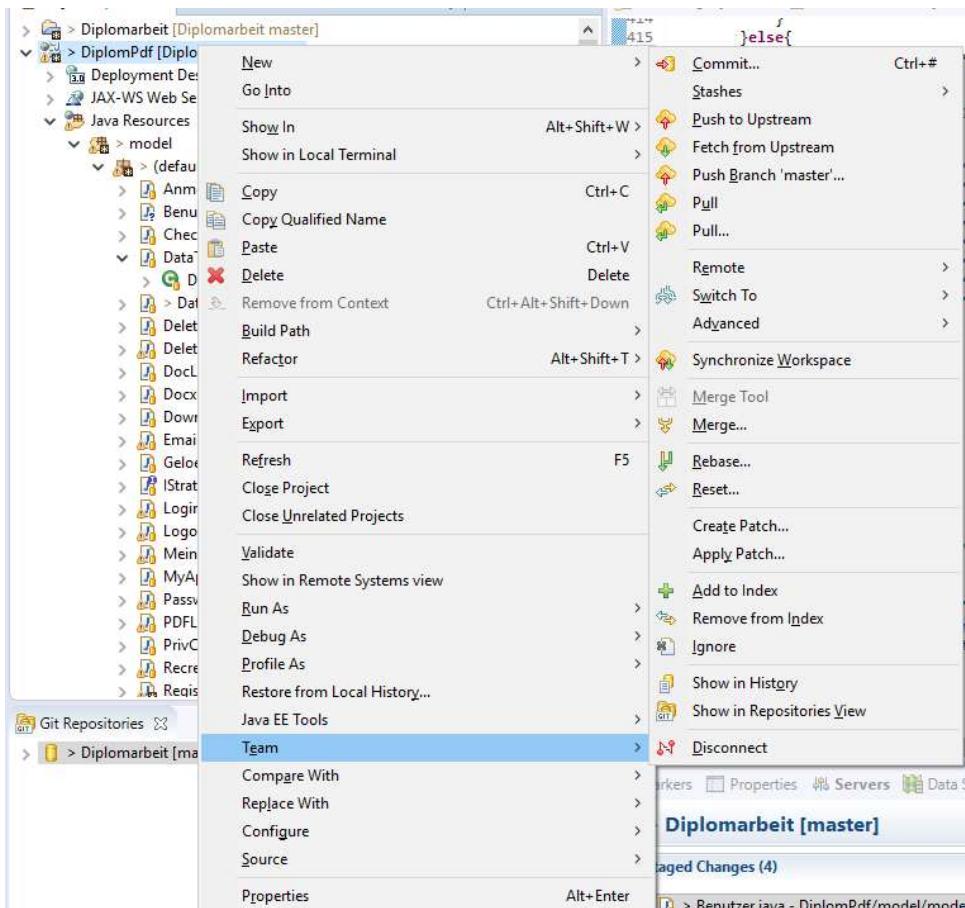


Abbildung 81 Ausführmöglichkeit weitere Befehle in JAVA

Github in Bezug auf unser Projekt:

Vorteile:

- Leichter Zugriff auf Projekt für alle Mitarbeiter und Beteiligte, zum Beispiel dem Betreuungslehrer
- Einfach zu verwalten, einfach Programmiererteile hinzuzufügen
- Gute grafische Aufbereitung und Veranschaulichung der älteren Versionen/ der History
- Stand von Mitarbeitenden zu sehen

Nachteile:

- Sehr oft merge Konflikte, Abhilfe durch bessere Absprache wer wann etwas in welchen Klassen programmiert
- Ohne Internetverbindung kein pullen oder pushen



## Umsetzung

Datenbankdesign und Realität

Design:

Wir als Gruppe haben dieses Design festgelegt.

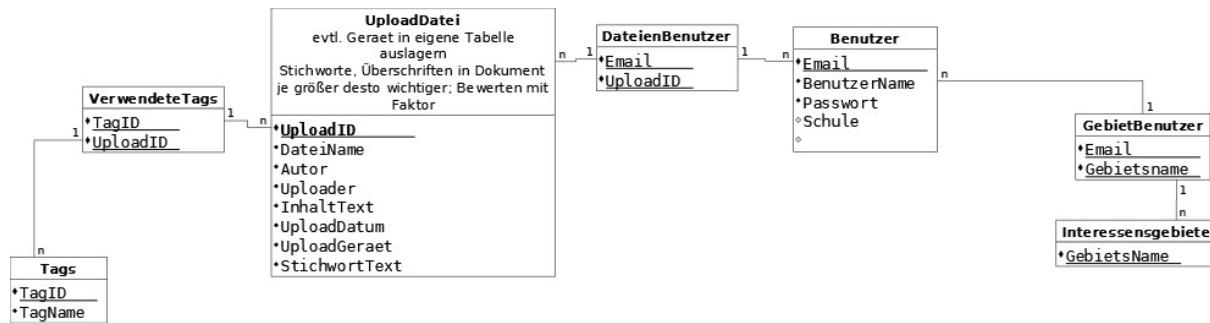


Abbildung 82 erstes Datenbankdesign

Nach Absprache mit unserem Lehrer haben wir es auf diesem Design reduziert:

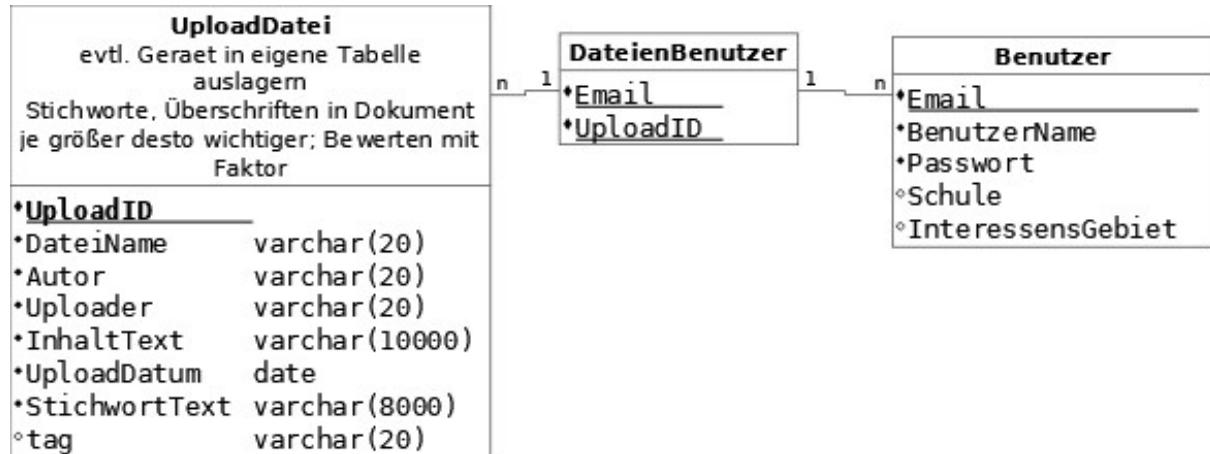


Abbildung 83 zweites Datenbankdesign



Umsetzung:

Erstellung der Tabelle Uploadaddaten:

```
create sequence UploadaddatenIdGenerator
    minvalue 1;

create table Uploadaddaten (
    uploadID int primary key default nextval('UploadaddatenIdGenerator'),
    dateiname character varying not null,
    autor character varying not null,
    uploader character varying not null,
    inhaltText character varying not null,
    stichwortText character varying not null,
    dateityp character varying(4) not null,
    language character varying,
    blobDatei bytea not null,
    status character varying(7),
    uploaddatum character varying(10),
    dokumentdatum character varying(10),
    zustand boolean not null,
    deletedatum character varying not null)
WITHOUT OIDS;

Alter table Uploadaddaten ADD language text NOT NULL DEFAULT('german');
```

Die Tabelle *Uploadaddaten* speichert alle Dokumentdaten zu den gelöschten und den aktiven Dokumenten. Als Information zu einem Dokument werden eine Uploadid, der Dateiname, der Autor, der Uploader (angemeldete Person), der Inhalttext, der Stichworttext, der Dateityp (PDF, DOCX, DOC, TXT), die Sprache, das BLOB-Objekt, den Status (private oder public), das Uploaddatum, das Dokumentdatum, der Zustand (gelöscht oder aktiv) und das Deletedatum eines Dokumentes abgespeichert.

Die Spalte Uploadid wurde als Primärschlüssel deklariert und als Default-Wert wird ein Sequence verwendet.

Mittels dem Uploader entsteht eine Verbindung zu der Tabelle Benutzer, da dieser als Fremdschlüssel festgelegt wird.

Mittels dem Befehl in der letzten Zeile wird bei der Spalte langauge als Default-Wert german angegeben.

Erstellung der Tabelle Benutzer:

```
create table Benutzer(
    benutzername varchar(20) primary key,
    email character varying(100) NOT NULL unique,
    passwort character varying(32) NOT NULL,
    authcode character varying(40));
```



Die Tabelle *Benutzer* ist zur Speicherung der Informationen aller Benutzer. Dabei werden die E-Mail-Adresse, der Benutzername, das Passwort und authcode gespeichert. Der authcode wird für die Funktion Passwort vergessen benötigt.

Als Primärschlüssel ist der Benutzername definiert.

Erstellung der Tabelle Suchwoerter:

```
create table Suchwoerter(
    benutzername varchar(20) primary key,
    email character varying NOT NULL,
    vorname text NOT NULL,
    nachname text NOT NULL,
    passwort text NOT NULL);
```

In der Tabelle Suchwoerter werden alle eingegebenen Suchwörter und der angemeldete User gespeichert.

Enddesign der Datenbank:

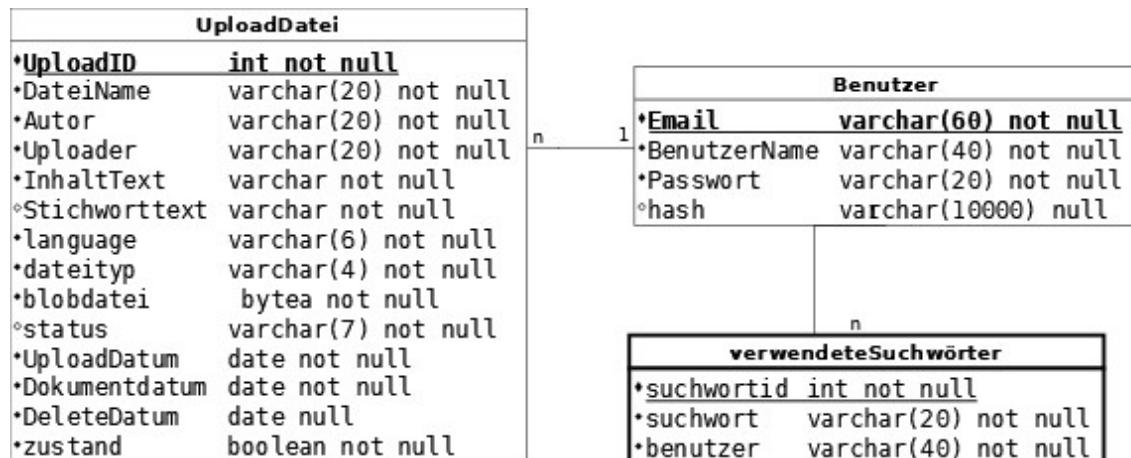


Abbildung 84 endgültiges Datenbankdesign

### Klassendesign

- DBManager
  - Konstruktor
  - getConnection
  - releaseConnection
  - Insert-Methoden
  - Select-Methoden
  - Update-Methoden
  - Delete-Methoden



- Strukturklassen für Arraylisten
  - Daten
  - Benutzer
  - Suchwoerter
  - Aufbau:
    - Variablen/Parameter
    - Getter und Setter
    - Konstruktor
    - `toString` Methode
- Test  
In der Klasse Test wurden die neu hinzugefügten Methoden ausprobiert und getestet, ob diese die gewünschten Ergebnisse liefern.

BLOB Umsetzung:

#### *BLOB-Datei abspeichern*

Nachfolgender Codeausschnitt stellt exemplarisch das Speichern eines BLOB-Feldes dar. Die relevanten Teile sind gelb hinterlegt.

```
pstmt = conn.prepareStatement(INSERT_DATA_SQL);
fis = filePart.getInputStream();

pstmt.setString(1, uploaddaten.getInhalttext());
pstmt.setString(2, uploaddaten.getUploader());
pstmt.setString(3, uploaddaten.getAutor());
pstmt.setString(4, uploaddaten.getDateiname());
pstmt.setString(5, uploaddaten.getStichworttext());
pstmt.setString(6, uploaddaten.getDateityp());
pstmt.setString(7, "private");
pstmt.setString(8, uploaddaten.getDokumentdatum());
pstmt.setString(9, getAktuellesDatum());
pstmt.setBinaryStream(10, fis, (int)filePart.getSize());
pstmt.setBoolean(11, true);
pstmt.executeUpdate();
```

Als Argument an die Funktion wurde das Objekt „filePart“ übergeben. Dieses stellt eine Methode „`getInputStream()`“ zur Verfügung. Mittels des Streams können die Daten der mitverbundenen Datei gelesen werden.

Das JDBC-Statement hat eine Methode „`setBinaryStream()`“. In der zweiten gelb markierten Zeile wird der binär Stream zum Schreiben mit dem übergehenden Lesestream verknüpft.



## BLOB-Datei auslesen

```
public byte[] BLOBAuslesen(Connection conn,int id)
{
    byte[] buf=null;
    try {
        String query = "SELECT blobdatei, LENGTH(blobdatei) FROM
uploaddaten WHERE uploadid = ?";
        pstmt = conn.prepareStatement(query);
        pstmt.setInt(1, id);
        ResultSet result = pstmt.executeQuery();
        result.next();

        int len = result.getInt(2);
        buf = result.getBytes(1);
        System.out.println("buf"+buf);

        pstmt.close(); pstmt=null;
    } catch (Exception ex) {
        ex.printStackTrace();
    }
    return buf;
}
```

Zum Auslesen des BLOB-Objektes werden das eigentliche BLOB-Objekt sowie die Länge des BLOB-Objektes ausgelesen.

Die Länge des BLOB-Objektes wird mittels der Methode *LENGTH(blobdatei)* ermittelt und dann als int ausgelesen.

Das eigentliche BLOB-Objekt wird mittels der Methode *getBytes()* aus der Datenbank ausgelesen und als byte in JAVA gespeichert.



## Volltextsuche Umsetzung

Die bereits im Detail einzeln beschriebenen Teile werden im folgenden Beispiel zusammengeführt:

```
public ArrayList<Daten> durchsuchenPrivate(Connection conn, String wort, String
username) {
ArrayList<Daten> daten = new ArrayList<Daten>();

String SEARCH_FOR_DATA_SQL_DATEN =
"SELECT uploadid, dateityp, dateiname, autor, uploaddatum, dokumentdatum, status
FROM (SELECT uploaddaten.uploadid as uploadid, uploaddaten.dateityp as dateityp,
uploaddaten.dateiname as dateiname, uploaddaten.dokumentdatum as dokumentdatum,
uploaddaten.uploaddatum as uploaddatum, uploaddaten.status as status,
uploaddaten.autor as autor, uploaddaten.uploader as uploader, uploaddaten.zustand
as zustand, setweight(to_tsvector(uploaddaten.language::regconfig,
uploaddaten.dateiname), 'A') ||
setweight(to_tsvector(uploaddaten.language::regconfig, uploaddaten.inhalttext),
'B') || setweight(to_tsvector('simple', uploaddaten.autor), 'C') as document"
FROM uploaddaten) p_search
WHERE p_search.document @@ to_tsquery('german', ?) and uploader=? And
zustand='true'
ORDER BY ts_rank(p_search.document, to_tsquery('german', ?)) DESC";

System.out.println(SEARCH_FOR_DATA_SQL_DATEN);
try {
    pstmt = conn.prepareStatement(SEARCH_FOR_DATA_SQL_DATEN);
    pstmt.setString(1, wort);
    pstmt.setString(2, username);
    pstmt.setString(3, wort);
    rs = pstmt.executeQuery();
    rs.getFetchSize();
    System.out.println("rs.getFetchSize()"+rs.getFetchSize());
    while (rs.next()) {
        int uploadid = rs.getInt(1);
        String dateityp = rs.getString(2);
        String dateiname = rs.getString(3);
        String autor = rs.getString(4);
        String uploaddatum = rs.getString(5);
        String dokumentdatum = rs.getString(6);
        String status = rs.getString(7);

        Daten zeile = new Daten(uploadid,dateityp,dateiname, autor,
uploaddatum, dokumentdatum, status,i);
        daten.add(zeile);
    }
    pstmt.close(); pstmt=null;
    rs.close(); rs=null;
} catch (SQLException e) {
    e.printStackTrace();
}
return daten;
}
```



## Weitere Features:

### 1. Sotierung der Daten

Daten nach

- Autor
- Dateiname
- Uploaddatum
- Dokumentdatum

zu filtern und zu sortieren

### 2. Daten löschen und wiederherstellen

Gelöschte Dokumente werden in der Datenbank mit der Kennung gelöscht versehen und bei Suchoperationen ausgeschlossen.

Das Wiederherstellen gelöschter Dokumente ist somit rein durch die Änderung der Kennung möglich.

Das endgültige Löschen der Dokumente wird durch Bestätigung derselben in der Ansicht „Gelöschte Dokumente“ bewirkt. Mit dem Löschen werden alle Daten zum Dokument aus der Datenbank entfernt.

### 3. Daten nach öffentlichen und privaten Daten unterscheiden

In allen Abfragen wurde der Öffentlichkeitsstatus des Dokumentes berücksichtigt.

### 4. Passwort zurücksetzen

Zur Abwicklung zum Zurücksetzen des Passwortes wurden die notwendigen Strukturen definiert.

## Testen

Es wurden laufend die neu implementierten Methoden durch eine Testklasse getestet. Weiters wurden alle SQL-Operationen erstmals in dem Programm pgAdmin getestet, bevor diese in eine Methode in JAVA implementiert wurden.

## Ausblick in die Zukunft Verena

- Analyse bei Suchwörtern, welcher Benutzer, welche Suchwörter am häufigsten verwendet
- Ausbau der Benutzertabelle -> genauere Registrierung von Benutzer
  - um Interessensgebiet besser einteilen zu können
  - Sprache zu definieren
- Leichte Erweiterungsmöglichkeiten bei DB
- Nicht-gruppierten Index hinzufügen->Tag
- Häufig verwendete Fehler beim Eingeben der Suchwörter abspeichern und als Suchparameter hinzufügen
- Eigene Methoden zum Ranking der Texte schreiben



## Anhang

### Glossar/ Stichwortverzeichnis – Wortdefinition:

Website	gesamter Webauftritt
Homepage	Startseite einer Website
Webseite	eine spezifische Seite mit einer URL - Adresse
W3C	World Wide Web Consortium
CDN	Content Delivery Network (Netzwerk, welches Inhalte liefert / bereitstellt)
Plugin	Erweiterung
Kreativitätstechnik	
Front-End	In der Informatik wird hiermit allgemein die Seite des Clients, bei einer Client – Server Struktur bezeichnet.
Framework	Hiermit wird ein Programmiergerüst bezeichnet, welches noch keinen fertigen Code darstellt, jedoch Komponente zum Bau eines Codes.
Back – End	In der Informatik wird hiermit allgemein die Seite des Servers, bei einer Client – Server Struktur bezeichnet.
Dead – end	Bezeichnet in der Webentwicklung eine Seite von welcher aus man nicht mehr auf eine andere Seite durch einen Link oder Button weiterkommen kann. → die Seite ist eine Sackgasse.
Wireframe	Ein konzeptioneller Entwurf einer Website
HTTP	Kommunikationsprotokoll welches die Kommunikation zwischen Client und Server aufbaut



## Quellenverzeichnis

URL: <http://www.ecma-international.org/publications/files/ECMA-ST-ARCH/ECMA-262%205th%20edition%20December%202009.pdf>

URL: <https://www.w3.org/TR/2014/REC-html5-20141028/>

URL: [https://www.w3schools.com/js/js\\_versions.asp](https://www.w3schools.com/js/js_versions.asp)

URL: <http://adaptivepath.org/ideas/ajax-new-approach-web-applications/>

URL: <http://tools.ietf.org/html/7231#section-4>

URL: <https://getbootstrap.com/docs/3.3/>

URL: <https://github.com/1000hz/bootstrap-validator>

URL: <https://cookieconsent.insites.com>

URL: <https://www.ncbi.nlm.nih.gov/pubmed/22869334>

URL: [https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin\\_policy](https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy)

URL: <https://opensource.org/licenses/MIT>

URL: <https://pdfbox.apache.org/>

URL: <https://www.duden.de/rechtschreibung/kryptisch>

URL: <http://www.oracle.com/technetwork/java/javamail/index.html>

URL: <https://www.philippbauert.de/study/se/design-pattern.php>

URL: <http://tomcat.apache.org/>

URL: <https://www.javatpoint.com/servlet-tutorial>

URL: <https://www.postgresql.org/about/> Postgres

URL: <https://www.pgadmin.org/> pgAdmin

URL: <https://www.w3schools.com/sql SQL>

URL: <http://www.oracle.com/technetwork/java/overview-141217.html> JDBC,

URL: <https://docs.oracle.com/javase/7/docs/api/java/sql/PreparedStatement.html>  
PreparedStatement

URL: <https://docs.oracle.com/javase/7/docs/api/java/sql/ResultSet.html> ResultSet

URL: <http://rachbelaid.com/postgres-full-text-search-is-good-enough/>  
Volltextsuche

URL: <http://qnimate.com/an-introduction-to-javascript-blobs-and-file-interface/>,  
BLOB-Objekte



Servlet- Bild Quelle: <https://www.javatpoint.com/servlet-tutorial>

Strategy. Pattern Bild Quelle: <https://www.philippauer.de/study/se/design-pattern/strategy.php>, von Philipp Hauer

## Literatur:

Bezug	Quelle
Anspracherichtung	Buch Rationales Werben, Werbemedien und Werbeplanung, S. 21
API	Joshua Bloch: How to Design a Good API and Why it Matters
das große Du	Duden, Sprachwissen
Designkonzept	Basic Design of Everyday Things _ Feedback von Don Norman
Github	A quick Introduction to Version Control with Git and GitHub, John D. Blischak, Emily R. Davenport, Greg Wilson
GSON/JSON	Instant GSON, Sandeep Kumar Patel
Hash	Handbook of Applied Cryptography, Autor: A. Menezes, P
Hashfunktion	The Art of Computer Programming, geschrieben von Donald E. Knuth
Kreativitätstechnik 635	James M. Higgins, Gerold G. Wiese: Innovationsmanagement. Kreativitätstechniken für den unternehmerischen Erfolg. Springer, Berlin 1996, ISBN 3-540-60572-X.
Marketingkonzept	TÜV Marketing für Techniker von Rupert Erhart
Normalformen	Grundlagen von Datenbanksystemen, Ramez Elmasri, Shamkant B. Navathe
Nutzen für den Kunden	Grundlagen Marketing Werbung S. 8 von Bernhard Ulrich
Relationale Datenbanksysteme	The Relational Model for Database Management - Version 2, E. F. Edgar
Servlet	JSP, Servlets, and MySQL
Sessions	Introduction to HTTP von Launchschool <a href="https://launchschool.com/books/http/read/statefulnessn">https://launchschool.com/books/http/read/statefulnessn</a>
SMTP	Programmer's Guide to Internet Mail (SMTP, POP, IMAP, and LDAP - HP Technologies), John Rhoton



## Tabellenverzeichnis

Tabelle 1 Browser.....	11
Tabelle 2 Nutzen .....	31
Tabelle 3 MD5 Verschlüsselung .....	73
Tabelle 4 Get / Post Servlet .....	82
Tabelle 5 SQL Befehle .....	98
Tabelle 6 Git- Befehle .....	112

## Abbildungsverzeichnis

Abbildung 1 DataTableServlet Request (siehe DataTableServlet).....	15
Abbildung 2 Response des DataTableServlet (siehe DataTableServlet) .....	16
Abbildung 3 Inhalt Response DataTableServlet .....	16
Abbildung 4 Ajax.....	17
Abbildung 5 DataTable.....	19
Abbildung 6 Dropzone.....	20
Abbildung 7 Modal .....	22
Abbildung 8 Bootstrap Validator.....	23
Abbildung 9 Navigationsbar angemeldet.....	26
Abbildung 10 Navigationsbar nicht angemeldeter Benutzer.....	26
Abbildung 11 Registrierfeld.....	26
Abbildung 12 Anmeldefeld .....	26
Abbildung 13 Abmeldefeld.....	27
Abbildung 14 Private/ Publicfeld.....	27
Abbildung 15 Symbol für PDF - Dokumente .....	27
Abbildung 16 Symbol für Microsoft Word Dokument .....	27
Abbildung 17 Symbol für Textdokument .....	27
Abbildung 18 Mobile Ansicht 1 .....	27
Abbildung 19 Mobile Ansicht 2 .....	27
Abbildung 20 Logo .....	29
Abbildung 21 Sitemap.....	30
Abbildung 22 Zugriff von Überall.....	32
Abbildung 23 Work smart, not hart .....	33
Abbildung 24 Community .....	33
Abbildung 25 Navigationsbar.....	34
Abbildung 26 Heroimage Startseite.....	35
Abbildung 27 Parallaxeffekt.....	36
Abbildung 28 Wireframe IST.....	37
Abbildung 29 Wireframe Plan .....	37



Abbildung 30 Wireframe.....	37
Abbildung 32 Mobiles Design Startseite 1 .....	38
Abbildung 31 Mobiles Design Startseite 2 Menü .....	38
Abbildung 33 Meet The Team IST .....	40
Abbildung 34 Meet The Team Plan.....	40
Abbildung 35 Registrieren Clientseitig Sequenzdiagramm.....	42
Abbildung 36 Heroimage Login, Registrieren ect.....	43
Abbildung 37 Wireframe Register Plan.....	44
Abbildung 38 Wireframe Register IST.....	44
Abbildung 39 Mobiles Registrieren .....	44
Abbildung 40 Wireframe Login IST .....	48
Abbildung 41 Wireframe Login Plan.....	48
Abbildung 42 Login Mobil Plan.....	49
Abbildung 43 Login Mobil IST .....	49
Abbildung 44 Sequenzdiagramm Dokumentseite .....	50
Abbildung 45 Dokumentseite Plan.....	51
Abbildung 46 Dokumentseite IST .....	52
Abbildung 47 Mobile Dokumentseite Plan .....	53
Abbildung 48 Mobile Dokumentseite IST .....	53
Abbildung 49 Ablaufdiagramm Upload .....	54
Abbildung 50 Logo PDFBox.....	63
Abbildung 51 POI Logo.....	66
Abbildung 52 Strategy Pattern .....	69
Abbildung 53: Strategy Pattern EasyDocs .....	70
Abbildung 54 Sequenzdiagramm Registrieren.....	72
Abbildung 55 Zugriff Login.....	74
Abbildung 56 Sequenzdiagramm Login .....	75
Abbildung 57: Passwort vergessen? Feature .....	76
Abbildung 58 Registerablauf .....	76
Abbildung 59 Sequenzdiagramm Passwort zurücksetzen .....	77
Abbildung 60 HttpSession .....	80
Abbildung 61 Servlet Threads* .....	81
Abbildung 62 GET Methode .....	83
Abbildung 63 Post Methode.....	83
Abbildung 64 Servlet Quelle Wikipedia.....	84
Abbildung 65 Funktionsweise JSP.....	87
Abbildung 66 Starten von Tomcat- Server .....	88
Abbildung 67 Library Webcontainer.....	89
Abbildung 68 Weg der Daten von JAVA Schnittstelle zur Datenbank .....	100
Abbildung 69 Ergebnis zum Veranschaulichen der Vereinfachung der Texte ..	105
Abbildung 70 Ergebnis zum Veranschaulichen der simplen Vereinfachung der Texte ..	106
Abbildung 71 Ergebnis der Durchsuchung von Texten.....	106



Abbildung 72 Ergebnis der Vereinfachung von einzelnen Wörtern mittels to_tsquery .....	106
Abbildung 73 Ergebnis der Vereinfachung von einzelnen Wörtern mit to_tsvector .....	107
Abbildung 75 Eingabe von Methode zum Vereinfachen von Akzenten .....	107
Abbildung 74 Ergebnis der Vereinfachung von Akzenten .....	107
Abbildung 76 Ergebnis der Wertung der Relevanz eines Wortes in Bezug zum Text .....	108
Abbildung 77 Ergebnis der Wertung der Relevanz eines Wortes in Bezug zum Text .....	108
Abbildung 78 Anzeige und Verwaltung von Repositoryen in JAVA .....	113
Abbildung 79 Verwaltung von einem Repository in JAVA .....	113
Abbildung 80 Ausführmöglichkeit weitere Befehle in JAVA .....	114
Abbildung 81 erstes Datenbankdesign .....	115
Abbildung 82 zweites Datenbankdesign .....	115
Abbildung 83 endgültiges Datenbankdesign .....	117

## Arbeitszeiten Thomas Kerber

Datum	Stunden	Kurzbeschreibung
15.09.2017	4	Studium von Servlets
20.09.2017	12	Erstellen einer Teseite, welche Upload und Download via php ermöglicht
22.09.2017	1	Besprechung mit Betreuungslehrer
25.09.2017	2	Senden von Suchbegriffen an Server
26.09.2017	2	JSON Studium
26.09.2017	4	GIT Studium
27.09.2017	2	GSON Studium
01.10.2017	1	Servlet JSON - GSON Suchbegriffe
03.10.2017	3	UploadServlet Filter
05.10.2017	1	Upload von Client
06.10.2017	2	Upload von Client
07.10.2017	3	Upload von Client
09.10.2017	2	Upload von Client mit DropzoneJS
05.11.2017	2	Upload von Client mit DropzoneJS
06.11.2017	2	jQuery DataTables erste Implementierung
13.11.2017	2	jQuery DataTables Search und Delete Button
15.11.2017	2	Besprechung Gruppe ohne Betreuungslehrer
15.11.2017	2	jQuery DataTables Servlet 1. Antwort
17.11.2017	4	Teamarbeit, Fehlerbehebung
18.11.2017	2	jQuery DataTables
20.11.2017	2	jQuery DataTables Servlet Parameter
25.11.2017	2	Studium von Ajax



28.11.2017	2	Implementierung Downloadbutton mit Servlet
29.11.2017	2	Implementierung Deletebutton mit Servlet
29.11.2017	1	Ajax Bugfix
01.12.2017	2	Modal zum Namen ändern bei Upload
02.12.2017	2	JS Namen ändern bei Upload
04.12.2017	1	Schnittstellen kurz beschrieben, Upload.html Refactoring Teil 1
05.12.2017	2	fixed json answer, erweitert daten, welche geschickt werden
12.12.2017	2	Downloadbutton Animation
13.12.2017	1	Besprechung mit Betreuungslehrer
13.12.2017	2	Implementierung von Icons
14.12.2017	1	DataTables Design
15.12.2017	4	Teamarbeit, Fehlerbehebung
16.12.2017	4	Startseite
17.12.2017	1	Startseite
20.12.2017	2	Login, HTML
21.12.2017	2	Register, HTML, jQuery Datatables 2nd Table
22.12.2017	2	DataTable Site Styling mobile + desktop
27.12.2017	1	Navbar collapse implementation
28.12.2017	1	DataType in Table
04.01.2018	1	Upload mit DataTableSite mergen
05.01.2018	2	Meet the Team
06.01.2018	2	Meet the Team
07.01.2018	2	Startseite Textinhalt
08.01.2018	1	Meet the Team skalierbarkeit
10.01.2018	2	Privacy Settings bei DataTable
11.01.2018	1	JavaScripts zusammenfügen
11.01.2018	1	id in table hidden
13.01.2018	1	remove files(visible) in dropzone after leaving modal
16.01.2018	1	hashcode generator
17.01.2018	3	Client side verification
19.01.2018	4	Teamarbeit, Fehlerbehebung
25.01.2018	6	Teamarbeit, Fehlerbehebung, Koordinierung
26.01.2018	2	Error Page
26.01.2018	4	Download Servlet
27.01.2018	3	Email Servlet
30.01.2018	6	Teamarbeit, Fehlerbehebung, Koordinierung
02.02.2018	1	Passwort Reset Site
02.02.2018	2	Register Serverside Antwort in Client zeigen
04.02.2018	2	Register Serverside Antwort in Client zeigen
05.02.2018	2	Send Email Serverside Antwort in Client zeigen
07.02.2018	1	Bilderoptimierung
10.02.2018	1	jQuery DataTables erweiterte Parameter



11.02.2018	4	Überschreiben implementieren
13.02.2018	1	remove delete option from public table
14.02.2018	1	Ajax auf jQuery
16.02.2018	1	check delete und privacy only changeable by owner
17.02.2018	1	charakter encoding
19.02.2018	1	delete hash after pw reset
20.02.2018	4	Teamarbeit, Fehlerbehebung
21.02.2018	2	fix mobile view
23.02.2018	2	hinzufügen des Logos
25.02.2018	2	Fehlerbehebung URI
28.02.2018	4	Studium von Amazon AWS
01.03.2018	2	Hochladen der Website über Amazon AWS
03.03.2018	1	Client side verification Erweiterung

## Arbeitszeiten Verena Gurtner

Datum	Stunden	Kurzbeschreibung
06.09.2017		Beginn der Diplomarbeit
15.09.2017-17.09.17	15	Recherche über Volltextsuche, Postgres, Servlets
20.09.2017	5	Recherche über Github, Mergeconflicte
22.09.2017	2	Erstellung einer Datenbank mit 1 Tabelle
18.09.17-24.09.17	10	Testung der von Postgres zur Verfügung stehenden Methoden, to_tsvector, ts_rank, tsquery
22.09.2017	1	Besprechung mit Professor Holzmann
29.09.2017	2	Update der ersten Tabelle, durch Sequence, Constrains
30.09.2017	2	Datenbank befüllt mit Dummydaten
07.10.17-08.10.17	4	Rechere BLOB-Dateien
14.10.17-15.10.17	7	Datenbank mit Eclipse verbinden
21.10.2017	2	Erstellung der Klasse Datenbank, Konstrukter, getConnection, releaseConnection
04.11.2017-05.11.2017	2	Erstellung von Methoden zum Auslesen aller Daten, alle Autoren, alle Dateinamen, ...
15.11.2017	2	Besprechung Gruppe ohne Betreuungslehrer
17.11.2017	4	Teamarbeit
18.11.2017	1	Implementierung der Methoden zum Hineinschreiben von Daten
19.11.2017	3	Erstellung von Testdokumenten zum befüllen und testen(PDF)+Fehlerbehebung
25.11.2017	3	Im Uploadservlet im vorgegebenen Strategy Pattern Methode zum Daten in DB speichern ausgeführt
08.12.2017	3	Ausbesserung bzw neu schreiben von Methoden zum auslesen von Daten
13.12.2017	1	Besprechung mit Betreuungslehrer
23.12.17-24.12.17	4	Methoden zum Vereinfachen von Texte schreiben
29.12.17-30.12.17	3	Methode zum Durchsuchen von Texten
06.01.18-07.01.18	5	Methoden zum Ranking der Texte
10.01.2018	2	Aufspaltung der Klasse Datenbank, in 3 Klassen



20.01.2018	2	Hinzufügen von Datenklasse, Benutzerklasse als ArrayList Dateityp, Update bestehender Klassen
22.01.2018	2	Methoden zum Registrieren, Login, Passwort vergessen
25.01.2018	6	Teamarbeit
27.01.2018	4	Daten auf DatatableSite ausgeben, durch JSON Antwort
28.01.2018	4	Fehlersuche bei JSON Antwort
29.01.2018	2	Verbesserung der JSON Antwort
30.01.2018	6	Teamarbeit
01.02.2018	3	Ausgabe der Daten in sortierter Form, nach Autor, Dateiname, Uploaddatum, Dokumentdatum
03.02.2018	2	Recherche über SQL-Injections
04.02.2018	3	Verbesserung der Methoden zum Schutz vor SQL-Injections
10.02.18-11.02.18	10	BLOB-Objekte speichern, Übung durch Bilder
11.02.2018	1	Update der INSERT Methode, hinzufügen der BLOB-Objekte
11.02.2018	2	Methode zum Auslesen der Dokumente hinzufügen
18.02.2018	1	Methoden zum Ändern von Zugriff, private/public
24.02.18-25.02.18	8	Hinzufügen der Papierkorbfunction, Erstellen neuer Tabelle+Methoden
03.03.18-04.03.18	5	Papierkorbfunction mittels Erweiterung der Uploaddaten realisiert + Methoden (Update einzelner Parameter)
10.03.2018	5	Löschen der überflüssigen Tabelle und Einfügen von Methode zur Papierkorbfunction
11.03.2018	4	Fehlerbehebung bei anderen Features durch löschen der überflüssigen Tabelle
15.03.2018	3	Aufbereitung der bestehenden Methoden
17.03.2018		Beginn der Dokumentation



## Arbeitszeiten Sara Hindelang

Datum	Stunden	Kurzbeschreibung
11.09.2017	2	Einführung funktionen GIT
22.09.2017	1	Besprechung Prof. Holzmann
24.09.2017	7	PDF auslesen
26.09.2017	2	PDF Fehlerbehebung
03.10.2017	3	lesen Textfiles
15.10.2017	7	lesen von Word Dokumenten
16.10.2018	5	StrategyPattern Files
19.10.2017	2	Umstrukturierung Extraktionsmethoden
22.10.2017	3	Einführung JSP und Anwendung
01.11.2017	5	Text über Uploadservlet auslesen
15.11.2017	2	Gruppenbesprechung Team ohne Prof.
15.11.2017	4	Fehlersuche Libraries
17.11.2017	4	Team Nachmittag
23.11.2017	6	Login bzw RegistrierServlet
26.11.2017	3	Extraktion der Metadaten PDFs
01.12.2018	4	Accountgebunden Ausbau Strukturierung
03.12.2017	3	Verbessern Registrieren
07.12.2017	7	PDF Änderung
13.12.2017	1	Besprechung Prof. Holzmann
14.12.2017	4	Registrieren & Login Update
16.12.2018	5	Fehlersuche Verknüpfung
17.12.2017	3	Metadaten WordDocs
27.12.2017	5	Suche Metadaten Textfiles
02.01.2018	7	DownloadServlet
05.01.2018	4	Fehlerbehebung allg.
13.01.2017	4	Ausbessern Download
25.01.2018	6	Emailsenden Passwort zurücksetzen/ Team
26.01.2018	3	Zurücksetzen ausbauen
30.01.2018	6	Team Nachmittag
01.02.2018	5	Fehlerbehebung Upload
02.02.2018	3	Ausbessern Email mit DB
07.02.2018	4	Exceptionhandling Fehlerhafte Dokumente
15.02.2018	2	Emailsenden Registrieren
24.02.2018	3	Auslagern Email universal
27.02.2018	7	Loginzyklus fix
02.03.2018	2	Datumsformat Metadatenfix
05.03.2018	5	Fehlersuche Fileopen



03.03.2018	4	Fehlerbehebung Download
06.03.2018	8	allgemeines Ausbessern überall
07.03.2018	4	Passwort Hashen