TYPO3 CMS:

Analyse und Optimierung von Usability-Prinzipien für Endanwender unter Einsatz von Backend Apps

M a s t e r a r b e i t

an der Hochschule für Angewandte Wissenschaften Hof
Fakultät Informatik
Studiengang Internet-Web Science

Vorgelegt von:	B.Eng. Andreas Steiger
Vorgelegt bei:	Prof. Dr. Jürgen Heym
Betreuer (TRITUM GmbH):	Dipl.-Betriebsw. Björn Jacob

Hof, 10.09.2015

TYPO3 CMS:

Analysis and Optimization of Usability Principles for End Users Employing Backend Apps

M a s t e r T h e s i s

supervised by the University of Applied Sciences Hof
faculty of Computer Science
degree course Internet-Web Science

submitted by:	B.Eng. Andreas Steiger
submitted to:	Prof. Dr. Jürgen Heym
supervisor(TRITUM GmbH):	Dipl.-Betriebsw. Björn Jacob

Hof, 10.09.2015

Abstract

Das TYPO3 CMS konnte in den vergangenen Jahren bis zur Version 6.2 LTS (Long Term Support) hinsichtlich Sicherheit, Funktion und Features stetig verbessert sowie erweitert werden. Jedoch ist der Stand der Benutzerfreundlichkeit im System veraltet. Die sich vollziehenden Veränderungen in der Benutzeroberfläche und der User Experience (Englisch für: Benutzererfahrung) des CMS der künftigen Version 7 sind vielversprechend. Die Ergebnisse der vorliegenden Ausarbeitung können ergänzend dazu beitragen, die Produktivität und Bedienbarkeit für TYPO3-Redakteure und -Integratoren beim Umgang mit dem TYPO3 CMS zu verbessern. Es soll ein Usability- und Software-Konzept für aufgabenfokussiertes Arbeiten mittels Backend Apps entwickelt werden. Die Details der Datenvisualisierung und einer Filterkomponente werden am Beispiel einer Listenansicht analysiert. Durch eine Konkurrenz-, Zustands- und Ideenanalyse werden die Grundlagen für die Entwicklung von TYPO3 Backend Apps geschaffen. Mit Hilfe von Anwendungsbeispielen werden die Systemintegration, Konfigurationen und Wireframes der Datenvisualisierung sowie des Filters erarbeitet. Darauf aufbauend wird ein Prototyp erstellt, der die grundsätzlichen Funktionalitäten sowie die Benutzeroberfläche verdeutlicht. Dieser findet anschließend Einsatz in einem A/B-Testverfahren, um die Entwicklungsresultate messbar zu bewerten. Die Ergebnisse der Usability-Tests zeigen eine deutliche Verbesserung in der Benutzerfreundlichkeit durch die Verwendung von TYPO3 Backend Apps. Somit wird bewiesen, dass durch ein aufgabenfokussiertes Arbeiten in Kombination mit einer Benutzeroberfläche nach aktuellen Konventionen ein besseres Nutzererlebnis gegenüber der aktuellen TYPO3-Version geschaffen werden kann. Zusammenfassend liefert das Konzept von TYPO3 Backend Apps einen belegbaren Produktivitätsgewinn für die Arbeit mit dem CMS und sollte zukünftig in das System integriert werden.

Abstract

In the last few years up to version 6.2 LTS (Long Term Support), the TYPO3 CMS has constantly been improved and expanded in terms of safety, function and features. However the state of usability in the system has remained deprecated. The upcoming changes in the user interface and the user experience of the CMS in the future version 7 are promising. The results of the present thesis can help additionally to improve the productivity and usability in handling for editors und integrators of TYPO3. It aims at developing a usability and software concept for task-focused work with Backend Apps. The details of the data visualization and of a filter component are analysed using the example of a list view. The foundations for the development of TYPO3 Backend Apps are created through an analysis of competition, state of affairs and existing ideas. By the help of use cases, the system integration, configuration and wireframes of data visualization and filter are being worked out. Based on this, a prototype is created which demonstrates the basic functionalities and the user interface. Afterwards the prototype is used in an A/B testing method to evaluate the development results measurably. The results of the usability tests show a significant improvement in the user-friendliness through the use of TYPO3 Backend Apps. Therefore it is proved that task-focused work in combination with a user interface according to current conventions can create a better user experience than the current version of TYPO3. In summary, it can be stated that the concept of TYPO3 Backend Apps provides a verifiable productivity gain for working with the CMS. Therefore it should be integrated into TYPO3 CMS in the future.

Vorwort

Die vorliegende Masterarbeit entstand in Zusammenarbeit mit der Firma TRITUM GmbH sowie der TYPO3 Association. Das Thema wurde zusammen mit Herrn Dipl.-Betriebsw. Björn Jacob (TRITUM GmbH) und Herrn Dipl.-Inf. Oliver Hader (TYPO3-Core-Entwickler) erarbeitet.

Diese Ausarbeitung richtet sich an die Leser, welche aktive oder interessierte Entwickler des TYPO3 CMS sind und sich mit der Benutzerfreundlichkeit des Systems auseinandersetzen möchten. Da diese Zielgruppe eine internationale Community im Rahmen des TYPO3 CMS umfasst, werden die Abbildungen in der vorliegenden Ausarbeitung in englischer Sprache dargestellt. Aufgrund der wissenschaftlichen Abhandlung stellt sich ein Nutzen für die TYPO3 Association heraus. Die analytischen Ergebnisse unterstützen die Weiterentwicklung des TYPO3 CMS und verbessern die zukünftige Arbeit damit.

An dieser Stelle bedanke ich mich bei Prof. Dr. Jürgen Heym für die wissenschaftliche Betreuung sowie die hilfreichen Anregungen. Mein Dank gilt ferner der TYPO3 Association und Mathias Schreiber, die es mir ermöglichten, diese Abschlussarbeit anfertigen zu können. Danke an die Entwickler der TYPO3-Community Felix Kopp, Daniel Siepmann und Gernot Schulmeister sowie an die Firma TRITUM GmbH für die Unterstützung. Darüber hinaus möchte ich den Korrekturlesern Michael Augustin, Nico Scherbarth und dem Ehepaar Gottschald für die investierte Zeit sowie meiner Frau für die Geduld bzw. Nachsichtigkeit danken.

Inhaltsverzeichnis

1 Einführung und Zielsetzung 1
1.1 Problemstellung 1
1.2 TYPO3 CMS – die Motivation 3
1.3 Leitlinien, Ziele und Konzepte der Arbeit 3
2 Grundlagen 6
2.1 Apps – Definition und Abgrenzung 6
2.2 Die Zukunft des TYPO3 CMS 7
2.3 User Experience (UX) bei Webanwendungen 9
2.3.1 Software-Ergonomie 9
2.3.2 Usability-Prinzipien 10
2.3.3 Interface-Design 13
3 Analyse 15
3.1 Zustandsanalyse TYPO3 CMS 15
3.1.1 Module und Aufgaben 15
3.1.2 Listenmodul 15
3.1.3 Lesezeichen im Backend 17
3.2 Notwendigkeitsanalyse für zusätzliche Backend Module 17
3.3 Konkurrenzanalyse 19
3.3.1 Konkurrenten und Kriterien 19
3.3.2 Datenvisualisierung und Usability am Beispiel von Listen 20
3.3.3 Verwendung von App-ähnlichen Konzepten 28
3.4 Konzeptvorschläge der TYPO3-Community 28
3.4.1 Focus Apps und andere App-Konzepte 28
3.4.2 Gedankensammlung zu TYPO3 Apps 31
3.5 Terminologie 33
3.6 Anwendungsfälle 34
4 Backend Apps im TYPO3 CMS 36
4.1 Listenmodul als TYPO3 Backend App 36
4.2 Systemintegration 37
4.3 Layout und Komponenten 39
4.3.1 Menü und Titel 41
4.3.2 Hierarchische Seitenstruktur 43
4.3.3 Presets, Filter und Ergebniskomponente 44
4.4 Aufbau der Konfiguration 44
4.5 App Engine 46
5 Visualisierung von Datensätzen 50
5.1 Datenvisualisierung im TYPO3 CMS 50
5.2 Tabellen im Backend 51
5.3 Einfluss der Visualisierungsaufgaben auf die benötigten Komponenten im UI-Layout 51
5.4 Aufbau der Ergebniskomponente als Datenvisualisierung 54
5.4.1 Kopfbereich 55
5.4.2 Inhaltsbereich 56
5.5 Computed Values 60
5.6 Optimierung der Darstellung 63
5.7 Visualisierung der Dateiverwaltung 66
5.8 Konfiguration der Ergebniskomponente 67

6 Filter 69
6.1 Definition 69
6.2 Prädikatsdefinition von Filtern 70
6.3 Facetten 71
6.4 Integration und Dynamik 75
6.5 Usability und Wireframes 77
6.6 Presets 80
7 Bewertung und Ausblick 84
7.1 Empirische Bewertung der Usability 84
7.1.1 Testverfahren 84
7.1.2 Auswertung 86
7.2 Zusammenfassung 87
7.3 Ausblick 88
8 Quellenverzeichnis 91
9 Anhang 94

Abbildungsverzeichnis

Abbildung 1: Geplante Releases von TYPO3 CMS; Quelle: http://typo3.org/typo3-cms/roadmap/ (Aufgerufen am 15.06.2015) 8

Abbildung 2: Beispiele für Elemente mit Aufforderungscharakter; Autor: A. Steiger, 2015 12

Abbildung 3: „Hamburger-Icon“ mit Label; Autor: A. Steiger, 2015 12

Abbildung 4: Listenmodul im Backend von TYPO3 CMS 7.3 (Unkenntlichmachung aufgrund Datenschutz); Autor: A. Steiger, 2015 16

Abbildung 5: Auswertung der Befragung bei den Agency Meetup Days 2014 (Auszug) zum CMS-Einsatz in mittelständischen Unternehmen; Quelle: [Veu15, S.6] 19

Abbildung 6: Listenansicht des CMS Drupal 7; Autor: A. Steiger 2015 22

Abbildung 7: Weitere Listenansicht des CMS Drupal 7; Autor: A. Steiger, 2015 23

Abbildung 8: Listenansicht im Backend des CMS Wordpress 4 (Unkenntlichmachung aufgrund Datenschutz); Autor: A. Steiger 2015 23

Abbildung 9: Listenansicht mit Kontextmenü in der Tabellenkalkulation Excel 2010; Autor: A. Steiger 2015 25

Abbildung 10: Listenansicht mit Kontextmenü in der Tabellenkalkulation Google Docs Tabellen; Autor: A. Steiger 2015 25

Abbildung 11: Listenansicht mit Ordnerbaum des Explorers in Windows 7; Autor: A. Steiger 2015 26

Abbildung 12: Listenansicht des Finders in Mac OSX; Quelle: http://www.macstories.net/mac/ extensibility-and-automation-changes-in-os-x-yosemite/ (Aufgerufen am 13.07.2015) .. 27

Abbildung 13: Wireframe zum ersten Backend-App-Konzept einer Listenansicht; Quelle: Anhang 13 29

Abbildung 14: Screenshot eines Backend Moduls mit fokussierter Aufgabenbearbeitung in TYPO3 CMS 6.2 LTS; Autor: G. Schulmeister, 2015 30

Abbildung 15: Integrationsübersicht der TYPO3 Backend Apps in einer Extension; Autor: A. Steiger, 2015 37

Abbildung 16: Namenskonventionen der Ordnerstruktur von TYPO3 Backend Apps; Autor: A. Steiger, 2015 38

Abbildung 17: Wireframe des Backend-App-Layouts mit flexibel erweiterbaren Bereichen; Autor: A. Steiger, 2015 40

Abbildung 18: Wireframe des Backend-App-Layouts mit beispielhaften Komponenten; Autor: A. Steiger, 2015 40

Abbildung 19: Wireframe des Header-Bereichs einer Backend App; Autor: A. Steiger, 2015 41

Abbildung 20: Wireframe der App-Einstellungen; Autor: A. Steiger, 2015 42

Abbildung 21: Wireframe des Backend-App-Titels mit Breadcrumb-Funktion; Autor: A. Steiger, 2015 ... 43

Abbildung 22: Grundgerüst der Konfigurationsdatei einer TYPO3 Backend App; Autor: A. Steiger, 2015 45

Abbildung 23: Klassenübersicht für den Registrierungsprozess einer TYPO3 Backend App; Autor: A. Steiger, 2015 47

Abbildung 24: Beispielhafte Übersicht der Fluid-Dateien einer TYPO3 Backend App und der Erweiterungsmöglichkeit mittels Partials; Autor: A. Steiger, 2015 48

Abbildung 25: Schema der Datenbanktabelle für benutzerdefinierte Presets der Ergebnisansicht; Autor: A. Steiger, 2015 48

Abbildung 26: Beispielhaftes Wireframe für Massendatenverarbeitung und aufgabenorientierte Visualisierung; Autor: A. Steiger, 2015 52

Abbildung 27: Wireframe als Beispiel für detaillierte Datenverwaltung und aufgabenorientierte Visualisierung; Autor: A. Steiger, 2015 53

Abbildung 28: Komprimierte Seitenbaumdarstellung für relevante Seiten als Dropdown; Autor: A. Steiger, 2015 54

Abbildung 29: Dropdown zur Spaltenauswahl; Autor: A. Steiger, 2015 55

Abbildung 30: Wireframe des Kopfbereiches der Ergebnisansicht am Beispiel einer Tabellenansicht; Autor: A. Steiger, 2015 56

Abbildung 31: Aktionsschaltflächen eines Datensatzes im Listenmodul von TYPO3 CMS 7; Autor: A. Steiger, 2015 57

Abbildung 32: Beispielhaftes Wireframe einer möglichen Position der Aktionsschaltflächen in einer Tabelle bzw. Liste; Autor: A. Steiger, 2015 58

Abbildung 33: Wireframe der reduzierten Aktionsschaltflächen am Beispiel des Veranstaltungsmanagements; Autor: A. Steiger, 2015 58

Abbildung 34: Wireframe des Inhaltsbereiches der Ergebniskomponente am Beispiel des Veranstaltungsmanagements; Autor: A. Steiger 2015 59

Abbildung 35: Konfiguration der Computed Values für Datensätze; Autor: A. Steiger, 2015 61

Abbildung 36: Konfiguration der Computed Values für einzelne Spalten; Autor: A. Steiger, 2015 62

Abbildung 37: Avatare bei TYPO3 CMS 7; Quelle: [Mac15] 63

Abbildung 38: Komprimierte Anzeige mehrerer Backend-Benutzer in einer platzoptimierten Spalte; Autor: A. Steiger, 2015 64

Abbildung 39: Screenshot der Tabellenkomprimierung bei Doodle; Autor: A. Steiger, 2015 65

Abbildung 40: Optimierter Hinweis auf weitere Spalten und horizontales Scrollen; Autor: A. Steiger, 2015 65

Abbildung 41: Wireframe des Inhaltsbereiches der Ergebniskomponente am Beispiel der Dateiverwaltung; Autor: A. Steiger, 2015 66

Abbildung 42: Wireframe spezieller Filtermöglichkeiten am Beispiel der Dateiverwaltung; Autor: A. Steiger, 2015 66

Abbildung 43: Konfiguration der Ergebnisansicht in der Datei: ListApp.yaml (App Engine); Autor: A. Steiger, 2015 67

Abbildung 44: Konfiguration der Ergebnisansicht in der Datei: EventApp.yaml (Event Manager); Autor: A. Steiger, 2015 68

Abbildung 45: UND-Verknüpfung von Filtern mit resultierender Schnittmenge; Autor: A. Steiger, 2015 . 69

Abbildung 46: Filterdarstellung am Beispiel einer SQL-Datenbankabfrage; Autor: A. Steiger, 2015 70

Abbildung 47: Konfiguration für intelligente Filter in einer TYPO3 Backend App; Autor: A. Steiger, 2015 72

Abbildung 48: Beispielbedingung eines Basistypen von intelligenten Filtern; Autor: A. Steiger, 2015 73

Abbildung 49: Beispielbedingung eines komplexen Typen von intelligenten Filtern; Autor: A. Steiger, 2015 74

Abbildung 50: Logische Verknüpfungen der Facetten im intelligenten Filter; Autor: A. Steiger, 2015 75

Abbildung 51: Venn-Diagramm einer Datensatzmenge und mehreren möglichen Filterbedingungen; Autor: A. Steiger, 2015 76

Abbildung 52: Intelligenter Ordner mit Filtermöglichkeiten im Apple Finder; Quelle: http://www.cultofmac.com/319619/10-essential-finder-tips-for-mac/ (Abgerufen am 26.08.2015) 78

Abbildung 53: Wireframe der Filterkomponente einer TYPO3 Backend App; Autor: A. Steiger, 2015 78

Abbildung 54: Markierung der Treffer-Strings in der Ergebniskomponente nach angewandtem Filter; Autor: A. Steiger, 2015 79

Abbildung 55: Auswahl und Anzeige des Seitenfilters; Autor: A. Steiger, 2015 79

Abbildung 56: Mehrfachauswahl einer Facette des intelligenten Filters; Autor: A. Steiger, 2015 80

Abbildung 57: Ausschnitt der Preset-Konfiguration einer TYPO3 Backend App; Autor: A. Steiger, 2015 .. 81

Abbildung 58: Wireframe der Preset-Komponente einer TYPO3 Backend App; Autor: A. Steiger, 2015 .. 82

Abbildung 59: Wireframe des Fensters zum Sichern eines Presets in einer TYPO3 Backend App; Autor: A. Steiger, 2015 82

Abbildung 60: Wireframe des Entfernen-Modus für eigene Presets in einer TYPO3 Backend App; Autor: A. Steiger, 2015 83

Abbildung 61: Mittelwerte der benötigten Zeiten über alle Testpersonen; Autor: A. Steiger, 2015 87

Tabellenverzeichnis

Tabelle 1: Gegenüberstellung der Möglichkeiten des Listenmoduls und individueller Backend Module; Autor: A. Steiger, 2015 18

Tabelle 2: Abgrenzung möglicher Manipulationen in unterschiedlichen Produktarten der Listenbearbeitung; Autor: A. Steiger, 2015 21

Tabelle 3: Beispieloperationen für datensatzbasierte Computed Values; Autor: A. Steiger, 2015 .. 61

Tabelle 4: Übersicht der enthaltenen Basistypen von intelligenten Filtern für die Spalten Dateien und Bilder; Autor: A. Steiger, 2015 74

Tabelle 5: Ausgewählte Features, welche mit bestimmten User Stories getestet werden; Autor: A. Steiger, 2015 85

Tabelle 6: Gegenüberstellung der Bewertungskriterien und den zu betrachtenden Nutzungsqualitätseigenschaften im A/B-Testverfahren; Autor: A. Steiger, 2015 85

Tabelle 7: Bewertungsmaßstab für die Ergebnisse des A/B-Testverfahrens; Autor: A. Steiger, 2015 86

Tabelle 8: Mittelwerte der Benotung mit Abschlussbewertung für die Testversionen A und B; Autor: A. Steiger, 2015 87

Abkürzungsverzeichnis

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
CMS	Content Management System
CSS	Cascading Style Sheet
CSV	Comma-Separated Values
ECM	Enterprise Content Management System
ERM	Entity-Relationship-Modell
HTML	Hypertext Markup Language
JSON	JavaScript Object Notation
LTS	Long Term Support
MIME	Multipurpose Internet Mail Extensions
PDF	Portable Document Format
PHP	PHP: Hypertext Preprocessor
QOM	Query Object Model
REST	Representational State Transfer
SQL	Structured Query Language
TCA	Table Configuration Array
TCE	TYPO3 Core Engine
UI	User Interface
UML	Unified Modeling Language
UX	User Experience
XML	Extensible Markup Language
YAML	YAML Ain’t Markup Language

1 EINFÜHRUNG UND ZIELSETZUNG

Das Open Source Enterprise Content Management System TYPO3 CMS gilt schon viele Jahre als einer der Marktführer zum Realisieren von Internetauftritten oder Webanwendungen. [Bre13, S.9] Ein Content Management System (CMS) bezeichnet eine Anwendersoftware zur Erstellung von Webseiten, ohne dass die Redakteure Programmier- oder HTML-Kenntnisse besitzen müssen. Der Fokus liegt auf der Inhaltsverwaltung sowie der Informationsaufbereitung einer Internetseite. Enterprise Content Management Systeme (ECM) unterstützen zusätzlich ein Unternehmen umfangreich bei dessen Geschäftsprozessen. Diese ECM bilden die funktional mächtigste Kategorie bei Content Management Systemen. [Bre13, S.7] Im Zusammenhang mit TYPO3 wird von einem Frontend und Backend gesprochen. Die Browseransicht einer Webseite, die vom CMS generiert wird und jeder im World-Wide-Web besuchen kann, gehört zum sogenannten Frontend. Das Backend hingegen stellt den Administrationsbereich und die Redakteursoberfläche dar. Es ermöglicht dem Nutzer mit entsprechenden Rechten die Verwaltung und Strukturierung aller Dateien, Seiten sowie Inhalte.

Von kleinen Seiten mittelständischer Unternehmen bis hin zu komplexen, großen Auftritten weltbekannter Firmen bietet TYPO3 CMS eine Vielzahl an Möglichkeiten. Zum Zeitpunkt der Erstellung dieser Arbeit erlebt die Software einen bedeutenden Wandel. Die Usability[1] in TYPO3 wird in allen Bereichen geprüft und hinterfragt. Die Produktivität der täglichen Arbeit steht im Mittelpunkt. Es ergeben sich neue Konzepte, welche in das System integriert werden und visionäre Ideen werden für die Zukunft des CMS konzipiert. Einer dieser Entwürfe befasst sich mit der Integration von Apps im TYPO3 Backend. Ein wegweisendes Konzept, welches das gesamte Backend des Systems revolutionieren kann.

1.1 Problemstellung

Die vorliegende Ausarbeitung beschäftigt sich mit den Aufgaben und Problemen der Endanwender des TYPO3 CMS. Diese Zielgruppe umfasst sogenannte Redakteure und Integratoren des Systems. In Bezug auf TYPO3 werden Autoren von Inhalten einer Webseite unter anderem als Redakteure bezeichnet, da dies ebenso als ein Online-Redaktionssystem klassifiziert werden kann. [Alt02, S.3] Ein

[1] Englisch für: Benutzerfreundlichkeit

TYPO3-Integrator sorgt für die Konfiguration des CMS, vergibt Zugriffsrechte für Backend-Benutzer und erstellt TYPO3-Templates[2]. Dieser programmiert dabei keine Erweiterungen und kümmert sich nicht um die TYPO3-Installation auf dem Server. [Lob14, S.8]

Das Internet verändert sich rasant. Besonders die Firma Apple prägte die Gesellschaft durch die Vorstellung des iPhones und ihrem hohen Anspruch an Usability und Design. Durch schnellere Breitbandverbindung, den Massenmarkt von Smartphones und neuen Mobilgeräten sowie durch etablierte Konventionen und Gewohnheiten im Internet, wurden in den letzten Jahren das Surfen und Browsen der Internetnutzer reformiert. TYPO3 CMS konnte in diesen Jahren hinsichtlich Sicherheit, Funktion und Features stetig verbessert und erweitert werden, jedoch ist der Stand der Benutzerfreundlichkeit im System veraltet. [Typ15] [Typ13a] Dies wird vor allem von Redakteuren besonders kritisiert. Um nicht nur gegenüber der starken Mitbewerber, wie Drupal, Wordpress oder Joomla die Marktanteile zu sichern, sondern auch ein angenehmeres Nutzererlebnis zu schaffen, wird in der kommenden TYPO3-Version 7 das Design modernisiert. Dabei erlangt die Usability eine hohe Bedeutung, welche einige signifikante Veränderungen für Funktionen und diverse Elemente mit sich bringt.

Das größte Problem jedoch fällt in der Betrachtung des sogenannten „Big Pictures“ auf. In mehreren Bereichen befindet sich die Benutzeroberfläche des Backends von TYPO3 CMS derzeit noch in einem recht komplexen, oft unübersichtlichen sowie veralteten Zustand. Dadurch ist die Bedienung für Integratoren und Redakteure nicht intuitiv. Die Nutzer kennen heute oft andere Arbeitswege, da sie geprägt sind von den Veränderungen und neuen Konventionen im modernen Internet. Gerade am Beispiel des Listenmoduls von TYPO3 CMS wird die Vielschichtigkeit und Problematik deutlich. Auf der einen Seite bietet es zu viele Möglichkeiten und es werden sämtliche Informationen über alle vorhandenen Datensätze angezeigt. Auf der anderen Seite ist die Funktionsweise so komplex und bietet einem Redakteur keine Darstellung der ausschließlich relevanten Daten. Einfache Funktionen, wie Sortierung oder Filter sind schwer zu bedienen oder nicht verfügbar.

Zusammengefasst bedeutet dies, dass TYPO3 CMS aktuell mehrere komplexe Module enthält, welche umfangreiche Aufgabenbereiche bearbeiten und dabei umständlich sowie mit hohem Erklärungsbedarf bedienbar sind.

[2] Englisch für: Vorlagen

1.2 TYPO3 CMS – die Motivation

Das System TYPO3 CMS 6.2 LTS (Long Term Support[3]) zeigt eine Vielzahl von Schwachstellen in der Benutzbarkeit. Die sich vollziehenden Veränderungen in der Benutzeroberfläche und der User Experience[4] (UX) des CMS der künftigen Version 7 geben Motivation. Die vorliegende Arbeit ermöglicht es, an einem innovativen und weltweit verbreiteten Softwareprodukt mitzuarbeiten. Dabei kann mit dem Ergebnis der Ausarbeitung die Produktivität der TYPO3-Anwender verbessert werden. Die Konzeption basiert auf einer stetigen Kommunikation mit der TYPO3-Community, um die Meinungen und Rückmeldungen mit in die Entwicklung von Backend Apps einzubeziehen.

1.3 Leitlinien, Ziele und Konzepte der Arbeit

Um eine Lösung der oben erwähnten Problemstellung für die Zielgruppe zu entwickeln, wurde die Idee der TYPO3 Backend Apps entworfen. Hierbei sollen die vielen Aufgaben großer Backend Module in kleine Aufgabenbereiche zerlegt werden. Zur Bearbeitung solcher fokussierten Aufgaben dienen Apps. Diese sollen die Komplexität des User Interfaces[5] (UI) verringern sowie eine übersichtliche Aufgabentrennung schaffen. Weiterhin ermöglicht dieses Konzept dem TYPO3-Integrator auf verständlichem Wege gezielt aufgabenspezifische Apps für Redakteure zu erstellen. Dadurch wird die momentan anspruchsvolle sowie zeitintensive Modulerstellung im Backend vereinfacht.

Es soll in dieser Arbeit ein Usability- und Software-Konzept für aufgabenfokussierte Backend Apps in TYPO3 CMS entwickelt werden. Hieraus entsteht ein Gesamtüberblick aller Komponenten und Möglichkeiten. Die Details der Datenvisualisierung und der Filterkomponente werden am Beispiel einer Listenansicht analysiert. Unterstützend soll davon ein Prototyp der Oberfläche implementiert werden, der die grundsätzlichen Funktionalitäten verdeutlicht. Die Lösung muss von TYPO3-Integratoren sowie -Redakteuren unmissverständlich verwendet werden können. Die erarbeiteten Konzepte und der Prototyp sollen eine Verbesserung in der Usability zeigen. Dies wird final mit Usability-Tests im A/B-Testverfahren gemessen und bewertet.

[3] Englisch für: Langzeitunterstützung
[4] Englisch für: Benutzererfahrung
[5] Englisch für: Benutzerschnittstelle

Es wird nach folgenden Leitlinien gearbeitet:

L1. Bestehendes erhalten und Innovationen einbringen
L2. Kontrollierbare Komplexität
L3. Produktivität erhöhen
L4. Fokus: Usability

Die Leitlinie L1 wurde vom TYPO3 CMS 7 übernommen („Embrace and Innovate“[6]), da in der vorliegenden Arbeit für diese Version entwickelt wird. Alle weiteren Leitlinien sind wichtig, um eine Software mit hoher Nutzungsqualität (siehe Kapitel 2.3.2) zu konzipieren. Daraus ergeben sich die folgenden Ziele:

Z1. Lernaufwand der Redakteure und Integratoren des TYPO3 CMS durch Backend Apps reduzieren
Z2. Rollenspezifische Komplexität
Z3. Minimale UI-Komplexität
Z4. Arbeitsschritte für das Bearbeiten von Inhaltselementen/Datensätzen reduzieren
Z5. Strong Defaults
Z6. Verwendbarkeit von Listenbearbeitung durch TYPO3 Backend Apps verbessern
Z7. Hohe Akzeptanz für TYPO3 Backend Apps erlangen
Z8. Listenmodul im Backend in fokussierte Aufgaben unterteilen

Den Lernaufwand reduzieren (Z1) bedeutet, weniger Schulungen anbieten zu müssen. Aktuell kommen Agenturen nicht umhin, TYPO3-Anfänger zu schulen, um einen Einstieg in das System zu ermöglichen. Diese sollen in Zukunft durch selbsterklärende Benutzbarkeit des Systems ohne lange Schulungszeit einen Schnelleinstieg schaffen. Bei der Reduzierung von Arbeitsschritten (Z4) geht es um konkrete Aufgaben, wie beispielsweise das Ändern eines Datensatzes. Für diese Aufgabe sind aktuell folgende Schritte notwendig:

Listenmodul öffnen -> Seite im Seitenbaum wählen -> Datensatztyp suchen -> Datensatz suchen -> bearbeiten.

Diese sollen auf wenige Aktionen reduziert werden:

App öffnen -> Datensatz suchen -> bearbeiten.

[6] Englisch für: festhalten und innovieren

Strong Defaults[7] ist ebenso ein Ziel der Version 7 von TYPO3 und soll dem Nutzer die Last der Entscheidung nehmen. Die zahlreichen Einstellmöglichkeiten des TYPO3 CMS werden dabei verlagert und bieten dem Anwender eine Vorauswahl als Bedienungshilfe an. Diese Vorschläge der Konfigurationsoptionen basieren auf den Erfahrungen, welche das TYPO3-Entwicklungsteam in den vergangenen Jahren sammeln konnte. Damit ist es möglich, die Konfiguration des Systems schneller und effizienter abzuschließen. [Mac14a]

Aus den genannten Zielen werden folgende zu entwickelnde Konzepte aufgestellt:

K1. Aufgabenfokussiertes Arbeiten
K2. Terminologie
K3. Integration der Backend Apps in TYPO3 CMS
K4. Konfiguration von TYPO3 Backend Apps
K5. Vererbung bei TYPO3 Backend Apps
K6. Visualisierung von Datensätzen in TYPO3 Backend Apps
K7. Benutzeroberfläche von TYPO3 Backend Apps (UI-Konzept)
K8. Bedienung von TYPO3 Backend Apps
K9. Intra-App-Kommunikation
K10. Filter in TYPO3 Backend Apps
K11. Ansichtsvorlagen (Presets) in TYPO3 Backend Apps

[7] Englisch für: starke Voreinstellungen

2 GRUNDLAGEN

2.1 Apps – Definition und Abgrenzung

Der Begriff „App“ ist die Kurzform für Applikation oder Anwendung. Softwareanwendungen sind auf dem Desktop-Computer sowie auf mobilen Geräten installier- und ausführbar. Jedoch ist eine Abgrenzung des Wortes Applikation sowie der Kurzform App erforderlich.

Apps unterscheiden sich im Vergleich zu Desktop-Anwendungen in ihren Funktionen, den Rechten und der Technik. Ein Programm auf einem Desktop-Computer zählt zu einer Software mit vollem Funktionsumfang (full-featured Software [New12]). Hingegen ist ein Programm auf dem mobilen Endgerät nur mit den wichtigsten, zielführenden Funktionen bestückt. Es bearbeitet fokussierte Aufgaben. Dies war der begrenzt leistungsfähigen Hardware der ersten mobilen Geräte geschuldet. Mittlerweile steigen die Anforderungen an mobile Apps, da die Hardwareentwicklung stark voranschreitet. Jedoch grenzen sich Desktop-Anwendungen weiterhin durch ihre Komplexität und dem uneingeschränkten Zugriff im Benutzerbereich des Computers ab. Full-featured Software benötigt entwickelte Installations- und Deinstallationsroutinen, die nur mit Administrationsrechten vom Benutzer ausgeführt werden können. Plattformunabhängig werden häufig einzelne Dateien und Einträge tief in das System eingebracht. Diese können nur durch Umwege entfernt werden. Ferner übergibt der Anwender seine Rechte der Software, womit diese vollen Zugriff auf den Computer erhält. In Bezug auf das Thema Sicherheit kommt es zu gefährlichen Lücken in vielen Bereichen.

Auch wenn bei mobilen Apps die Installations- und Deinstallationsroutinen vom Betriebssystem übernommen werden, fallen genauso Risiken an. Unkontrollierter Datenschutz und Sicherheitslöcher in der „Sandbox“ bringen Gefahren mit sich. Die „Sandbox“ bezeichnet einen Softwareaufbau. Dieser führt die Anwendung isoliert aus, damit die Software nur begrenzten Zugriff auf bestimmte Systembereiche und Ressourcen hat. Microsoft, Apple und andere Anbieter versuchen, diese und weitere Sicherheitsrisiken mit den eigenen Online-Geschäften für Apps, den sogenannten App-Stores, zu umgehen. Alle mobilen Anwendungen werden vor der Veröffentlichung im eigenen Hause geprüft. Diverse Sicherheitsrisiken und ein unerwünschtes Verhalten können damit vermindert werden. Darüber hinaus werden Apps durch verschiedene Berechtigungsschranken in ihrem Zugriff eingegrenzt. Dadurch kann der Anwender selbst bestimmen, ob die App auf Ortungsdienste, Fotos, das Mikrofon u. Ä. zugreifen darf.

Mittlerweile können Apps nicht mehr nur dadurch beschrieben werden, dass es Anwendungen auf mobilen Geräten sind. Durch Microsoft Windows 8 und die aktuellen Mac OS Betriebssysteme von Apple, hielten Apps auch Einzug in die Desktop-Computer und Notebooks. Weiterhin finden sie ihren Platz in Fernsehgeräten, Uhren und Brillen (Google Glass). Auch wenn auf all diesen modernen und leistungsfähigen Geräten viel komplexere Programme laufen könnten, wird das Konzept von der Bearbeitung dezidierter Aufgaben beibehalten. Außerdem ist es möglich, Apps in Desktop-Anwendungen, beispielsweise dem Browser Google Chrome, zu installieren und zu nutzen. Damit ergeben sich zwei grundsätzliche Einsatzgebiete:

- Betriebssystem
- Anwendung

Da eine App ursprünglich das Ziel verfolgt, kleine fokussierte Aufgaben zu bewältigen, liegt der Gedanke nicht fern, eine komplexe Anwendung in mehrere Apps zu unterteilen. Die Bedeutung von Apps nimmt dadurch stetig zu. Prinzipiell ist heutzutage unter dem Begriff „App“, ohne einen Wortzusatz, eine fokussierte Anwendung für ein Betriebssystem zu verstehen. Eine App in einer Anwendung selbst wird bisher nur selten eingesetzt. Interessant hierbei ist, dass in diesen Fällen immer die Möglichkeit besteht, eine separate Verknüpfung auf dem Desktop sowie an anderen Stellen im System (z.B. Taskleiste oder Startmenü) zu erzeugen. Damit kann ein natives App-Verhalten simuliert werden.

2.2 Die Zukunft des TYPO3 CMS

Als im Jahr 2014 die TYPO3-Version 6.2 LTS veröffentlicht wurde, gab es intensive Zusammentreffen der aktiven Kernentwickler (Active Contributer), um über die nächsten Schritte des CMS zu entscheiden. In dieser Zeit taten sich viele Visionen und Leidenschaften auf, die sich unter der Leitlinie „Embrace and Innovate“ zusammenfanden. Festhalten (Embrace) an dem, was sich über die Jahre bewährt hat und TYPO3 einzigartig macht: z.B. das TCA, der Seitenbaum oder die FormEngine TCEforms. Wie auch Innovationen in das System integrieren (Innovate), wo es möglich und notwendig ist. Damit sollen eine moderne und attraktive Webapplikation für Endanwender sowie solide Alleinstellungsmerkmale gegenüber anderen Content Management Systemen geschaffen werden. [Mac14]

Zum Zeitpunkt der Erstellung dieser Arbeit ist die Version 7 des TYPO3 CMS in Entwicklung. Die Backend-Oberfläche wurde von Grund auf mit dem Framework „Twitter Bootstrap“ realisiert und erscheint in einem modernen Flat-Design. Der Fokus liegt bei der Usability im täglichen Gebrauch sowie

der User Experience von Redakteuren und Integratoren. Da diese Aspekte im TYPO3 CMS in den letzten Jahren wenig Beachtung fanden, gilt nun die Vision, die Arbeitsabläufe der Endanwender im Alltag zu verbessern und angenehmer zu gestalten. Zu den Zielen der Version 7 wurden von den Kernentwicklern folgende Schwerpunkte herausgearbeitet:

- User Experience für Redakteure verbessern
- Einen schlanken und leichtgewichtigen Core entwickeln
- Strong Defaults (siehe Kapitel 1.3)

Ein weiterer Kernpunkt liegt in einer strukturierten sowie regelmäßigen Veröffentlichung von neuen Versionen. Heute und in Zukunft wird ein Plan verfolgt, der kontinuierliche Releases von TYPO3 vorsieht. Abbildung 1 zeigt den geplanten Ablauf der kommenden Versionen. Die großen Schritte der LTS-Versionen sind geprägt durch drei sich wiederholende Phasen: Entwicklung, Release und Feedback.

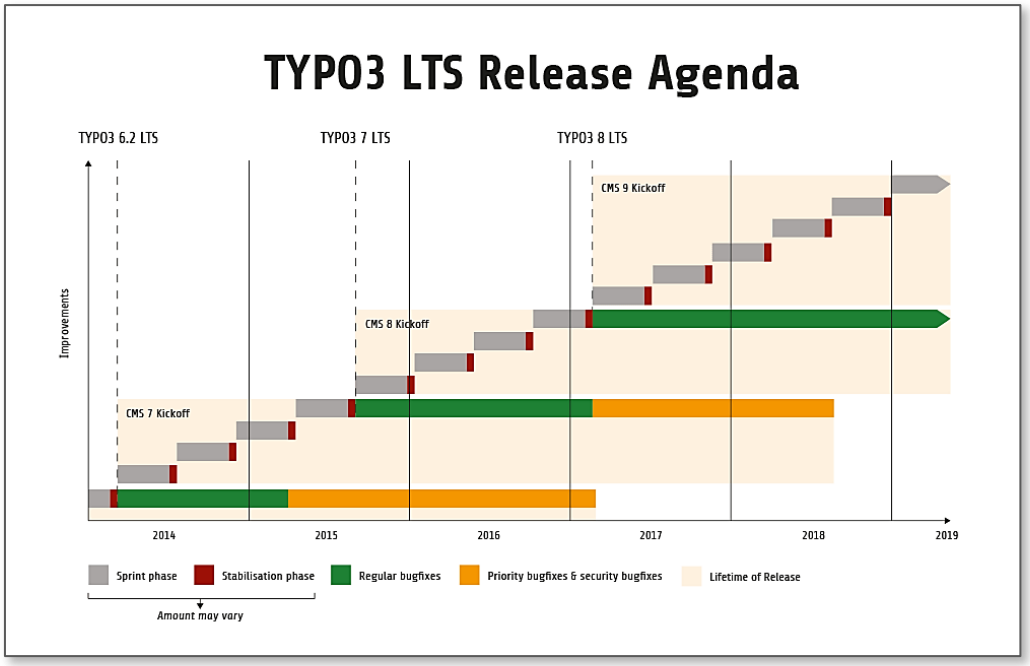


Abbildung 1: Geplante Releases von TYPO3 CMS;
Quelle: http://typo3.org/typo3-cms/roadmap/ (Aufgerufen am 15.06.2015)

Das Fazit der Zusammenkünfte 2014 in der TYPO3-Community ist deutlich: Die Zukunft des TYPO3 CMS legt den Fokus auf die Usability, die User Experience sowie das User Interface. [Bas14]

2.3 User Experience (UX) bei Webanwendungen

Bei der Entwicklung von Webanwendungen ist es wichtig, auf einem Fundament aus nachgewiesenen sowie erfolgreichen User Experiences aufzubauen. In den folgenden Abschnitten wird UX anhand der Themenkomplexe Software-Ergonomie, Usability und Interface-Design näher behandelt. Experten wie Donald A. Norman [Nor02], Steve Krug [Kru14] und andere haben in diverser Literatur viel Wissenswertes über ihre jahrelange Erfahrung festgehalten. In diesem Kapitel werden die wichtigsten Fakten kurz zusammengefasst.

2.3.1 Software-Ergonomie

Wenn es um die Thematik der Usability geht, darf eine Betrachtung der Software-Ergonomie nicht fehlen. Entwickler sollten sich über die Auswirkungen einer Software auf den Menschen bewusst sein. Sie müssen das Arbeitsumfeld des Nutzers kennen und in das Konzept des Produktes mit einbeziehen. Ein Unterbrechen des täglichen Arbeitsablaufes darf nicht aufgrund von mangelhafter Bedienbarkeit geschehen. Es liegt eine produktive Anwendung vor, wenn der Nutzer nicht genötigt wird, bei Internetsuchmaschinen zu suchen oder vor der Benutzung einen langen Beschreibungstext zu lesen. Dies beeinflusst das Wohlbefinden des Anwenders. Weiterhin sinkt das Vertrauen zu dem Softwareprodukt, wenn etwas nicht funktioniert, wie es erwartet wird. Ebenso können schlecht lesbare Inhalte, versteckte Elemente oder unstrukturierte Layouts zu Frustration führen. In den meisten Fällen gibt sich der Benutzer zusätzlich selbst die Schuld an den Schwierigkeiten im Umgang mit der Applikation [She15]. Die Ursachen hierfür liegen jedoch bei den Entwicklern und Herstellern. Grundsätzlich beschreibt Ergonomie einen wissenschaftlichen Ansatz, damit die höchste Zufriedenheit für den Nutzer erarbeitet werden kann. Es geht um die Anpassung der Arbeitsbedingungen an die menschlichen Fertigkeiten und Fähigkeiten.

1999 wurde eine Studie (Rage Against the Machine MORI-Studie) im Auftrag von Compaq an 1.255 britischen Arbeitnehmern durchgeführt [Bli05]. Das Ergebnis ist, dass sich die Hälfte der Befragten frustriert fühlt, wenn Zeit für das Beheben von Computerproblemen investiert werden muss. Rund ein Drittel sagte aus, dass IT-Probleme Grund für längere Arbeitszeiten und die Arbeit zu Hause sind. Demotivation, Unzufriedenheit sowie wirtschaftliche Einbußen sind hierbei die Resultate. Daher ist es entscheidend, bei der Softwareentwicklung auf eine angemessene Usability zu achten, um ein angenehmes sowie produktives Arbeiten zu gewährleisten.

2.3.2 Usability-Prinzipien

Usability ist ein Begriff, der nicht nur im Zusammenhang mit Softwareanwendungen oder Computerinteraktion verwendet wird. Er begegnet den Menschen auch in vielen Lebenssituationen. Das Öffnen einer Tür, das Auspacken eines Pakets oder das Bedienen eines Haushaltsgerätes, sind nur einige Beispiele, wo die Usability ausschlaggebend für die Verwendung ist. Die Benutzbarkeit entscheidet über die Nutzungsqualität von Soft- oder Hardware.

Wenn es um die Usability von Applikationen geht, müssen Informationen der User in Erfahrung gebracht und Feedback von ihnen eingeholt werden. Eine gut benutzbare Anwendung zeichnet sich besonders durch kontinuierliche Tests und Rückmeldungen von Anwendern bei der Entwicklung aus. Nur so kann ein effektives und effizientes Produkt für den Endnutzer entstehen. Über die Jahre wurde eine Vielzahl von Feedback in Bezug auf Webanwendungen eingeholt und gesammelt. Dadurch können einige allgemeine Usability-Prinzipien aufgestellt werden, die heutzutage bei jeder Webanwendung von Bedeutung sind. Zwei grundsätzliche Thesen können daher aufgestellt werden:

- Wenn von einer Software der Eindruck entsteht, dass die Bedienung einen hohen Zeitaufwand bedarf, minimiert dies die Wahrscheinlichkeit der Nutzung [Kru14, S.6]
- Anwender einer Software verfolgen immer ein Ziel, dass sie schnellstmöglich, effizient sowie effektiv erreichen wollen [Kru14, S.22]

Der allgemeine Computer-Nutzer ist konfrontiert mit Stress, vielen Aufgaben und wenig Zeit für IT-Probleme. Er ist gezwungen seine Arbeit produktiv zu erledigen. Beispielsweise möchte ein Redakteur auf einfachem und schnellem Wege Inhalte einer Webseite verändern. Um dies zu ermöglichen, wird eine anwenderfreundliche Applikation mit hoher Nutzungsqualität benötigt. Diese zeichnet sich durch folgende Eigenschaften aus:

- *Nützlich*: Kann die Software das, was der Anwender braucht?
- *Erlernbar*: Sind die Funktionen für den Nutzer verständlich?
- *Einprägsam*: Muss die Bedienung beim wiederholten Anwenden erneut erlernt werden?
- *Effektiv*: Arbeitet das System korrekt und dient der Zielerfüllung des Anwenders?
- *Effizient*: Kann der Anwender mit der Applikation seine Aufgaben in einer angemessenen Zeit und mit zumutbarem Aufwand erledigen?
- *Begehrenswert*: Liebt der Nutzer die Software und hat er Freude an der Nutzung?

  [Kru14, S.9]

Um diese Eigenschaften erreichen zu können, müssen grundsätzliche Usability-Prinzipien beachtet werden. Zusammenfassend sind im Folgenden einige wichtige Grundsätze aufgelistet:

- Inhalte strukturieren und deutlich voneinander abgrenzen
- Visuelle Hierarchien und klar definierte Bereiche erstellen
- Orientierungshinweise in der Navigation geben, um dem Nutzer jederzeit seine Position mitzuteilen
- Vorteile der bekannten Konventionen nutzen

Die Inhalte der Bereiche einer Seite oder Ansicht in einer Webanwendung müssen auf einem Blick erkennbar sein. Daher sind Strukturen, visuelle Hierarchien und Abgrenzungen von hoher Bedeutung. Außerdem erwartet der Nutzer Elemente an bestimmten Positionen mit dazugehörigen Funktionen. Diese Konventionen sind aus den Erfahrungen im Web entstanden und sollten implementiert werden. Weiterhin kann die Benutzeroberfläche übersichtlicher gestaltet werden, indem detaillierte Einstellungen oder Beschreibungstexte in Detailansichten ausgelagert werden. Hierfür können Icons oder Anrisstexte (Teaser) Verwendung finden, die zu einer weiteren Ansicht navigieren. Ebenso sollte dem Anwender jederzeit eine Information über die Position in der Applikation vermittelt werden. Es ist wichtig, dass der Nutzer einen nahezu gedankenlosen Weg zum Ziel seiner Aufgabe gehen kann. Er soll nicht darüber nachdenken müssen, für welche Funktion ein Label steht. Vielmehr muss dem Nutzer in jedem seiner Arbeitsschritte die Sicherheit gegeben werden, dass er weiß, wo er sich befindet und was er gerade bearbeitet.

Die genannten Usability-Prinzipien werden durch folgende Funktionen und Eigenschaften von Elementen oder Systemkomponenten erreicht:

- Affordances (Aufforderungscharakter)
- Constraints (Beschränkungen)
- Mapping (Beziehungen)
- Feedback (Rückmeldung)
  [Nor02, S.9 ff.]

Der Begriff Affordances wurde von Donald A. Norman geprägt und klärt die Funktion eines Elementes. Der Nutzer wird aufgefordert, damit zu interagieren. Ein deutlicher Aufforderungscharakter ist beispielsweise bei Formular-Textfeldern oder Schaltflächen zu erkennen (siehe Abbildung 2). Der Anwender kennt diese Elemente und weiß, welche Aktionen diese erwarten bzw. welche Funktion dahinter steht. Wenn mit dem Cursor in eine eingerahmte Box geklickt wird, kann ein Text eingege-

ben werden. Die Betätigung eines Buttons bestätigt eine Eingabe, führt eine Aktion aus (z.B. Suche) oder navigiert zu einer neuen Seite.

Abbildung 2: Beispiele für Elemente mit Aufforderungscharakter; Autor: A. Steiger, 2015

Beschränkungen (Constraints) helfen dem Anwender bei der Entscheidungsfindung und sind richtungsweisend für die Arbeit mit der Webapplikation. Auch wenn technisch mehrere Wege möglich wären, sollten die Entwickler die Möglichkeiten auf ein bis zwei reduzieren, um dem Nutzer die Bedienung zu erleichtern. Weiterhin ist es mit Beschränkungen möglich, die Oberfläche einfach und verständlich zu halten und die technische Komplexität des Systems zu verbergen.

Der Aspekt Mapping beschreibt die Beziehung zwischen zwei Komponenten. Praktisch bedeutet dies, dass beispielsweise eine Funktion eine Aktion auslöst oder eine Schaltfläche eine Aufgabe ausführt. Die Beziehungen sollten klar und verständlich für den Anwender sein. Beipielsweise kennt der Nutzer durch die Realität und heutige Konventionen das Mapping eines Buttons mit einem grünen Haken und der dazugehörigen Bestätigungsfunktion. Das Gegenstück durch eine Kennzeichnung mit dem roten „X“ und der dahinterliegenden Abbruchaktion ist ebenfalls geläufig. Icons statt Text werden immer beliebter und macht diese Beziehungen verständlicher. Durch aussagekräftige Farbwahl, ein Icon oder ein erklärendes Piktogramm kann langer Text entfernt werden, um eine Benutzeroberfläche aufgeräumter wirken zu lassen. Die Gefahr besteht jedoch im Missverstehen. Nur bestimmte Icons sind bekannt und können ohne darüber nachzudenken gedeutet werden. Manche Dinge sind klarer und der Natur des Menschen näher, sodass ein Verständnis vorausgesetzt werden kann (siehe weiter oben: bestätigen und abbrechen). Elemente, die nicht klar in ihrer Bedeutung sind, benötigen eine Erklärung, Hilfe oder ein Label. Das oft umstrittene „Hamburger-Icon“ für die Navigation ist bis heute noch nicht bei allen Webanwendern bekannt, sodass in diesem Fall ein Label nicht vernachlässigt werden sollte.

Abbildung 3: „Hamburger-Icon“ mit Label; Autor: A. Steiger, 2015

Eines der wichtigsten Features, die ein benutzerfreundliches Programm aufweisen sollte, ist das Feedback. Das System muss dem Anwender eine visuelle Rückmeldung über das liefern, was gerade in der Software geschieht. Ein Nutzer muss erkennen, ob gerade Daten verarbeitet werden oder wel-

che Teile der Software sich nach einer Aktion geändert haben. Dies sollte deutlich sichtbar und mit unterstützenden Hinweisen, Farben sowie Icons geschehen. Weiterhin ist wichtig, dass der Nutzer ein direktes und nahezu unverzögertes Feedback über seine Eingaben oder die aktuelle Systemverarbeitung bekommt.

Zusammenfassend bedeutet dies, dass die Anwendung und deren Funktionen selbsterklärend sein sollten. Der Nutzer darf nicht durch Unklarheiten, Verwechslungsgefahren sowie falschen Bezeichnungen zum Nachdenken gezwungen werden. Dadurch wird der Anwender in seinem Arbeitsfluss unterbrochen und benötigt mehr Zeit, um sein Ziel zu erreichen.

2.3.3 Interface-Design

Das Interface-Design von Webanwendungen beschäftigt sich mit Technologien, die eine Anwendung im Browser dem Verhalten einer nativen Applikation angleicht und dem Nutzer das Arbeiten damit erleichtert. Eine Browseranwendung ist oft geprägt durch wiederholtes Laden von neuen Seiten, zusätzlichen Formularen sowie der Isolation vom Betriebssystem. Das bedeutet beispielsweise, dass Dateien in ein Browserfenster bewegt werden können, dies aber zu keiner sinnvollen Interaktion führt. Um Programme im Web dynamischer zu gestalten, sind heutzutage folgende Integrationen entscheidend:

- Inline-Editing
- Drag-and-Drop
- Übergänge und Animationen
- Sofortiges Feedback (Autovervollständigung, Live-Vorschläge, Validatoren, Filter, usw.) [Sco09]

Im Hinblick auf die Usability wird damit eine Vielzahl an Erleichterungen und bekannter Verhalten einer Desktopanwendung geschaffen. Inline-Editing, Drag-and-Drop sowie Animationen werden in der vorliegenden Arbeit nicht weiter behandelt, da es den Kontext im Rahmen der Kernabhandlung verlassen würde.

Der Aspekt „sofortiges Feedback“ wurde hinsichtlich der Usability schon im vorherigen Kapitel 2.3.2 näher betrachtet. Im Kontext des Interface-Designs handelt es sich hierbei um sogenannte Lookup-Muster, die dem Nutzer schnelle Hilfe sowie Technologien für ein schnelles Arbeiten bieten. Autovervollständigung von Formularen, Live-Vorschläge und -Suchen analog zu Google sowie Filter, die bei der Eingabe der ersten Buchstaben sofort aktiv werden, sind heute weit verbreitet. Die modernen

Internetnutzer verwenden diese Technologien täglich und denken nicht mehr darüber nach. Erst wenn eines dieser Features fehlt, fällt es dem User auf. Sein Arbeitsfluss wird dadurch unterbrochen.

Beim Thema Interface-Design sind ebenso unterschiedliche Displaygrößen sowie Geräte, welche mit Handberührungen bedient werden, zu beachten. Bei touch-fähigen Geräten existieren kein Cursor und damit kein Rechtsklick oder Mouseover-Effekt. Dadurch können verschiedene Aktionen nicht ausgeführt werden. Dies darf bei einer Produktkonzeption nicht übersehen werden. Grundsätzlich sind die technischen Implementierungen der genannten Technologien nicht zu unterschätzen. Beim Betrachten der Usability werden einige Aspekte deutlich. Besonders durch Drag-and-Drop und das Inline-Editing wird eine hohe Anzahl an Möglichkeiten der Fehlbedienung/-eingaben geschaffen, welche durch die Software abgefangen werden müssen.

3 ANALYSE

3.1 Zustandsanalyse TYPO3 CMS

3.1.1 Module und Aufgaben

In den ersten Kapiteln wurde gezeigt, dass TYPO3 CMS als Content Management System klassifiziert werden kann. Das Backend eines CMS besteht aus verschiedenen Modulen, die für bestimmte Aufgaben zuständig sind. Abhängig vom Ziel, bietet ein Modul für das Erledigen dieser Aufgaben eine Visualisierung der vorhandenen Datensätze sowie Möglichkeiten zum Erstellen neuer. Ebenso werden Funktionen zur Modifikation und des Managements bereitgestellt. Eine Variante zur Darstellung von mehreren Daten kann eine Liste oder Tabelle sein. Ein Eintrag in einer Tabelle definiert sich beim TYPO3 CMS durch ein bestimmtes Objekt mit einem spezifischen Datentyp und mehreren Eigenschaften. Das Objekt „Seite“ ist ein solcher Typ. Im TYPO3 CMS wird jeder Datensatz zusätzlich einer dieser Seitenobjekte untergeordnet. Um dies in jedem Fall zu ermöglichen, stellt das System nach der Installation mindestens eine Root-Seite als Elternelement zur Verfügung. Zur Datenvisualisierung bietet das CMS im Backend zwei Module: Seite und Liste. Neben den Inhaltselementen in der Layout-Darstellung im Seitenmodul zeigt das Listenmodul mehrere sortierte Tabellenansichten.

3.1.2 Listenmodul

Das Listenmodul zeigt alle existierenden Datenbankeinträge einer Seite im System. Diese werden über die jeweilige Seite (Elternelement) entsprechend gefiltert. Die Navigation ist über eine hierarchisch strukturierte Seitenbaumansicht möglich. Weiterhin werden unterschiedliche Typen von Datensätzen jeweils separat gelistet, da bei einer gemeinsamen Tabelle kein greifbarer Informationsmehrwert erzielt werden würde. Bisher sind die Aufgabenbereiche eines einzelnen Backend Moduls recht umfangreich, um an zentralen Stellen eine Vielzahl von Funktionen im CMS zu bewältigen und Dateninformationen zur Verfügung zu stellen. Das Listenmodul ist eines der komplexesten Module im TYPO3 CMS und wird häufig, selbst von routinierten Anwendern, in dessen Vielschichtigkeit nicht durchdrungen. Individuelle Sortierungsmöglichkeiten der Tabellen sind vorhanden, jedoch führt die

Implementierung über mehrere Ebenen zu Verständnisproblemen. Das Anwenden der Sortierung leitet den User zu einer weiteren Ansicht, der sogenannten Single Table View[8]. An dieser Stelle erhält der Benutzer keinerlei Orientierungspunkte, auf welcher Ebene er sich momentan befindet. Diesen Weg, über eine zweite Ebene und einer neuen Darstellung mit nur einer Liste eines Datentyps, kann ein Anwender nur schwer nachvollziehen. Daher kommt es ohne Breadcrumb[9] oder einen Navigationshinweis zur Desorientierung. Weiterhin existiert keine Filteroption zum Eingrenzen der Ergebnisse. Selbst ein Integrator mit umfassenden Rechten hat lediglich begrenzte Möglichkeiten, das Listenmodul für Redakteure über das Rechtemanagement benutzerfreundlicher anzupassen. Er kann die Ansicht der verschiedenen Datentypen einschränken und den Endanwender damit unterstützen, seine relevanten Einträge besser zu finden. Jedoch ist dies nicht immer möglich. Beispielsweise darf ein Redakteur lediglich Inhalte, Kategorien und Neuigkeiten bearbeiten. Falls diese gemeinsam die gleiche Elternseite verwenden, werden alle weiteren Datensätze im Listenmodul angezeigt. Der Redakteur möchte jedoch ausschließlich Neuigkeiten sehen und editieren. Speziell für diesen Fall lassen sich die unwesentlichen Einträge nicht ohne weiteres ausblenden, um einen Fokus zu setzen. Nachfolgend ist ein Screenshot des Listenmoduls mit separater Ansicht der Datentypen dargestellt, welcher den eben beschriebenen Anwendungsfall visualisiert.

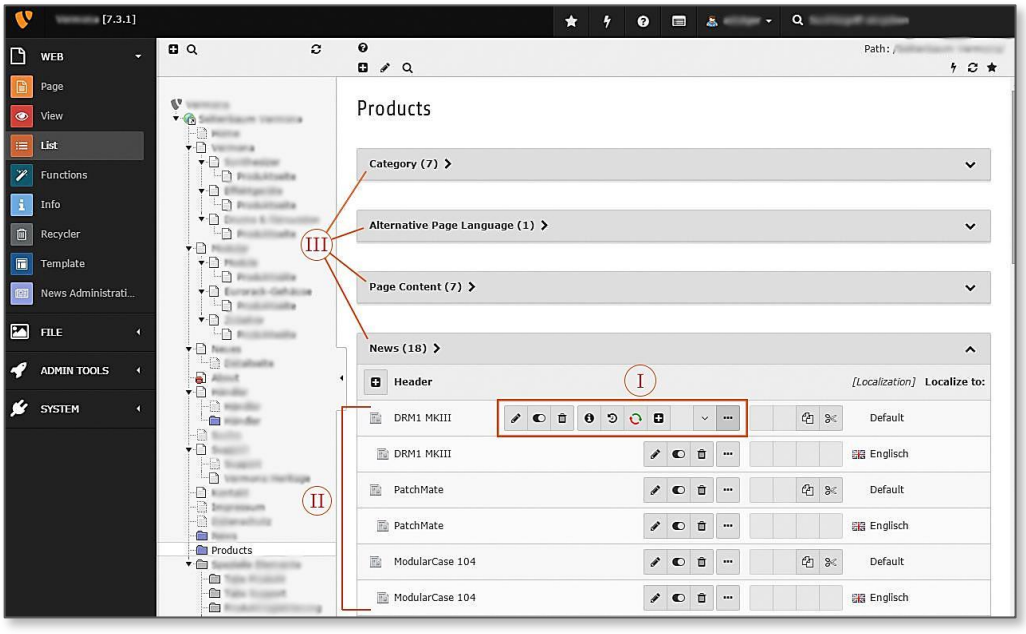


Abbildung 4: Listenmodul im Backend von TYPO3 CMS 7.3 (Unkenntlichmachung aufgrund Datenschutz); Autor: A. Steiger, 2015

[8] Englisch für: Einzeltabellenansicht
[9] Englisch für: Brotkrumenpfad

Durch die Vielzahl an Aufgaben, für welche das Listenmodul vorgesehen ist, existieren diverse Ansichten. Diese werden von der ersten Darstellungsebene aus über Aktionsschaltflächen (siehe I) oder ein Kontextmenü erreicht. Damit wird deutlich, dass hierbei ein Arbeiten auf mehreren Ebenen stattfindet. Die erste ist die Management-Ebene. Diese bildet die gelistete Visualisierung der Datensätze (siehe II) sowie -typen (siehe III) und wird in Abbildung 4 gezeigt. Die Single Table View kann über die jeweilige Tabellenbeschriftung (siehe III) erreicht werden und stellt eine gefilterte Management-Ebene dar. Die Zweite setzt sich aus den verschiedenen Bearbeitungsmodi sowie Massendatenverarbeitungsfunktion zusammen und kann daher als Persistenz-Ebene bezeichnet werden. Beispiele hierfür sind das Editieren des Verlaufes, des Zugriffs von Seiten oder der Eigenschaften eines Eintrages.

Die Single Table View ist eine fokussierte Visualisierung. Mit der Lesezeichen-Funktion im TYPO3 CMS Backend kann eine solche Ansicht für den späteren Gebrauch gespeichert werden und stellt damit ein App-ähnliches Verhalten dar.

3.1.3 Lesezeichen im Backend

TYPO3 CMS bietet mit sogenannten Lesezeichen die Möglichkeit, eine globale Sprungmarke zu setzen. Damit kann von jeder Stelle im CMS zu einer vordefinierten Ansicht gewechselt werden. Die Erstellung eines solchen Lesezeichens wird über das Stern-Symbol im Backend getätigt. Damit kann beispielsweise für die Bearbeitung von Neuigkeiten ein Lesezeichen auf die Single Table View dieser gesetzt werden. Mit nur einem Schritt wird somit die fokussierte Tabellendarstellung erreicht. Jedoch ist diese Ansicht in der Nutzungsqualität, je nach Anwender, gewöhnungsbedürftig und wenig intuitiv.

Es wird ersichtlich, dass durch die eben genannten Aspekte der Bedarf wächst, einem Redakteur zusätzliche, übersichtliche sowie individuelle Listenansichten mit hilfreichen Grundfunktionen bereitzustellen. Dies kann ausschließlich über zusätzliche Backend Module geschehen, was im nächsten Abschnitt näher beschrieben wird.

3.2 Notwendigkeitsanalyse für zusätzliche Backend Module

Im Austausch mit der TYPO3-Community und verschiedenen TYPO3-Redakteuren kommen regelmäßig die Benutzbarkeitsdefizite der TYPO3 CMS Version 6.2 LTS zur Sprache. Ob Integrator, Redakteur, Entwickler oder Administrator - jeder stößt im Laufe des Umgangs mit dem CMS auf die Probleme

der veralteten Usability. Entwickler bemühen sich, zusätzliche Backend Module mit benutzerfreundlicher Oberfläche zu entwickeln, um dem Kunden einen produktiveren Arbeitsablauf zu ermöglichen. Redakteure sind auf TYPO3-Schulungen angewiesen, um einen Einstieg in die Oberflächenbenutzung zu erlangen und sich zurechtzufinden. Lediglich Nutzer, die täglich mit dem TYPO3 CMS arbeiten, kennen das System gut und wissen, wie sie schnell zum Ziel kommen. Diese Anwender werden als Power User bezeichnet. Voraussetzungen für ein produktives Arbeiten mit dem CMS sind Erfahrung und regelmäßiges Nutzen der Software. Diese Anforderungen gilt es zu minimieren, so dass auch Nutzer, die keine Power User sind, effizient mit dem System arbeiten können.

Wie im Kapitel 3.1.2 beschrieben, sind Rechtevergaben und Modulanpassungen für eine fokussierte Redakteursarbeit nur bedingt möglich. Damit ist das Listenmodul von TYPO3 CMS in der Individualisierung stark eingeschränkt und es besteht oft die Notwendigkeit für ein zusätzliches Backend Modul. Entwickler haben hierbei eine Vielzahl von Möglichkeiten, um ein individuell angepasstes Modul für den Redakteur zu erstellen. Die Zustands- und Problemanalyse macht den wachsenden Bedarf eines individuellen Backend Moduls deutlich. Solch eine Implementierung erfordert jedoch einen hohen Zeitaufwand und damit einhergehende Entwicklungskosten. Dies liegt darin begründet, dass TYPO3 CMS keine Templates oder Komponenten anbietet, auf die Entwickler für eine Modulerstellung zugreifen können. Ein Backend Modul muss von Grund auf neu entworfen und programmiert werden. In der folgenden Tabelle werden die Defizite des Listenmoduls sowie die Vorteile eines individuellen, zusätzlichen Backend Moduls gegenübergestellt.

	Listenmodul	Individuelles Backend Modul
Individuelle, tabellenübergreifende Listen	✗	✓
Individuelle Ansichten (Diagramme, Kacheln, …)	✗	✓
Anpassbarkeit der Benutzeroberfläche	⊗	✓
Verbesserung der Produktivität im Arbeitsablauf	⊗	✓
Ausschließlich benötigte Datensätze anzeigen	⊗	✓
Nutzung ohne Entwicklungsaufwand	✓	⊗
Filterfunktion	✗	✓
Sortierfunktion	✓	✓
Nutzung der TYPO3-Zwischenablage	✓	✗

Legende: ✓ *= möglich;* ✗ *= nicht möglich;* ⊗ *= eingeschränkt möglich*

Tabelle 1: Gegenüberstellung der Möglichkeiten des Listenmoduls und individueller Backend Module; Autor: A. Steiger, 2015

Die Aspekte, welche durch „eingeschränkt möglich“ markiert sind, beziehen sich sowohl auf eine Einschränkung der technischen Möglichkeiten als auch in der wirtschaftlichen Betrachtung. Einige Punkte sind zwar technisch aufwendig umsetzbar, stehen jedoch nicht in einem wirtschaftlich angemessenen Kosten-Nutzen-Verhältnis.

Aus den genannten Fakten und der Übersicht wird die Notwendigkeit für zusätzliche Backend Module bei der Verwendung des TYPO3 CMS deutlich. Das Ziel hinter diesem Bedarf ist klar erkennbar: Die Produktivität der Arbeit mit dem System erhöhen. Bis heute müssen Entwickler versuchen, sich Wege zu schaffen, um angepasste Backend Module möglichst ohne großen Zeitaufwand umzusetzen. Dies soll durch die Einführung von TYPO3 Backend Apps fundamental verändert werden. Das TYPO3 CMS soll zukünftig Lösungen anbieten, um schnell sowie mit einfachen Mitteln ein zusätzliches, aufgabenfokussiertes Backend Modul zu erstellen.

3.3 Konkurrenzanalyse

3.3.1 Konkurrenten und Kriterien

In diesem Kapitel wird eine Konkurrenzanalyse vorgenommen und bewertet. Die Schwerpunkte in der Betrachtung liegen in der Datenvisualisierung mittels einer Liste sowie die verfügbaren Manipulationen. Ebenso wird die CMS-Konkurrenz auf Konzepte der aufgabenfokussierten Arbeit untersucht. Die Wahl der konkurrierenden Content Management Systeme basiert auf den Befragungsergebnissen der Agency Meetup Days 2014. Hierbei wurden Agenturen zur Produktwahl bei Internetauftritten für mittelständische Unternehmen befragt. Von 196 Befragten verschiedener Internetdienstleister wurden die Nennungen des Softwareproduktes gezählt. Jede Person hatte die Möglichkeit, mehrere Programme anzugeben. Das Ergebnis der am meisten genannten Content Management Systeme ist in der folgenden Abbildung dargestellt.

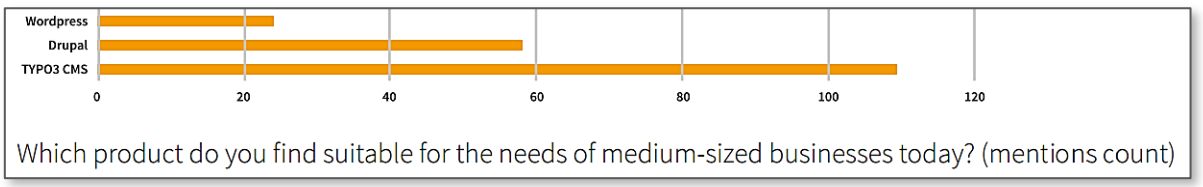


Abbildung 5: Auswertung der Befragung bei den Agency Meetup Days 2014 (Auszug) zum CMS-Einsatz in mittelständischen Unternehmen; Quelle: [Veu15, S.6]

Aufgrund des Befragungsergebnisses werden die Systeme *Wordpress* und *Drupal untersucht*. Weiterhin wird über den CMS-Sektor hinaus Software betrachtet, die Konventionen zur allgemeinen, digitalen Listenverarbeitung in der täglichen Büroarbeit prägen. In diesem Zusammenhang wird die Tabellenkalkulation *Excel* analysiert. *Excel* ist mit Abstand das meistgenutzte Programm für lokale Listen- sowie Tabellenbearbeitung und besitzt einen hohen Bekanntheitsgrad bei TYPO3-Endanwendern. Ebenso werden bei der Analyse Tabellenkalkulationen herangezogen, welche auf Internettechnologien basieren. Hierbei fällt die Entscheidung auf die Webdienste von *Google Docs* mit deren Tabellenprogramm.

Darüber hinaus werden die Dateiverwaltungsansichten der Betriebssysteme in die Konkurrenzanalyse mit berücksichtigt. Diese gebräuchlichen Benutzeroberflächen und Interaktionen sind bei Redakteuren sowie Integratoren aus ihren alltäglichen Tätigkeiten bekannt. Durch die hohen Marktanteile von Apple Mac OSX und Microsoft Windows [Pry14] werden der *Explorer* von Windows sowie der *Finder v*on Mac OSX untersucht.

Die Datenvisualisierung einer Listen- bzw. Tabellenansicht wird nach folgenden Kriterien bewertet:

1. Aufbau des Layouts
2. Erweiterte Ansichten
3. Navigation bzw. Orientierung
4. Speicherung
5. Tabellenmanipulationen:

5.1. Management-Ebene
- *5.1.1. Sortierung*
- *5.1.2. Filterung*
- *5.1.3. Größenanpassung*
- *5.1.4. Verschiebung von Spalten*
- *5.1.5. Ein- und Ausblendung von Spalten*

5.2. Persistenz-Ebene
- *5.2.1. Hinzufügung von Zeilen*
- *5.2.2. Hinzufügung von Spalten*
- *5.2.3. Ein- und Ausblendung von Zeilen*
- *5.2.4. Bearbeitung*
- *5.2.5. Entfernung von Zeilen*
- *5.2.6. Entfernung von Spalten*

Die Kriterien werden anhand der Grundinstallation und Standardkonfiguration der Systeme untersucht. Zusätzlich verfügbare Module finden keine Berücksichtigung.

3.3.2 Datenvisualisierung und Usability am Beispiel von Listen

Zunächst werden die genannten Softwareprodukte hinsichtlich der Datenvisualisierung und Usability integrierter Listenansichten untersucht. Hierbei muss als Erstes eine Abgrenzung in den Manipulatio-

nen sowie den Aufgabenbereichen der unterschiedlichen Produkttypen der Mitbewerber getroffen werden. Tabellenkalkulationen verfolgen andere Ziele in der Bearbeitung als Content Management Systeme. Diese dürfen in ihren Möglichkeiten sowie ihrem Kontext nicht gleich gestellt werden. Die uneingeschränkte Flexibilität und Editierfähigkeit einer Tabellenkalkulation (z.B. *Excel)* ist in einem CMS oder einer Dateiverwaltung eines Betriebssystems (z.B. *Explorer* oder *Finder*) oft nicht erwünscht oder notwendig. Außerdem werden Tabellenkalkulationsdateien im Gegensatz zu den öffentlichen Inhalten eines CMS für die interne Verwendung verfasst. Des Weiteren stellt die Listenansicht im Backend eines Content Management Systems nur eine Abstraktionsebene dar, die im Frontend keine visuelle Bedeutung hat. Die relevanten Möglichkeiten dieser verschiedenen Produkttypen erarbeitet die nachfolgende Tabelle.

Manipulation	Tabellenkalkulation	Listenbearbeitung im CMS	Listenbearbeitung in der Dateiverwaltung
5.2.1. Hinzufügung von Zeilen	✓	✓	✓
5.2.2. Hinzufügung von Spalten	✓	✗	✗
5.2.3. Ausblendung von Zeilen	✓	⊘	⊘
5.2.4. Bearbeitung	✓	⊘	⊘
5.2.5. Entfernung von Zeilen	✓	✓	✓
5.2.6. Entfernung von Spalten	✓	✗	✗

Legende: ✓ *= ja;* ✗ *= nein;* ⊘ *= eingeschränkt*

Tabelle 2: Abgrenzung möglicher Manipulationen in unterschiedlichen Produktarten der Listenbearbeitung; Autor: A. Steiger, 2015

Am Beispiel des TYPO3 CMS visualisiert Anhang 1 mit Hilfe eines Screenshots Zeilen und Spalten in einem CMS. Der in Tabelle 2 verwendete Begriff „eingeschränkt“ bezeichnet in diesem Zusammenhang die Einschränkung der Benutzeroberfläche, welche aus Gründen der Zugriffsrechte oder technischen Implementierung nicht alles ermöglicht. Als Schlussfolgerung dieser Auswertung werden in der folgenden Analyse die Punkte 5.2.2 und 5.2.6 nicht mit berücksichtigt. Das Hinzufügen und Entfernen von Spalten ist bei der Arbeit mit einem CMS nur in der Entwicklung und Administration sinnvoll.

CMS Drupal

Die erste Software, welche betrachtet wird ist *Drupal*. Hierbei handelt es sich um ein auf PHP-basierendes CMS, das sich besonders für Internetauftritte mittelständischer Unternehmen eignet. [Bre13, S.14] [Veu15, S.6] Das Besondere an diesem System ist, dass die Benutzeroberfläche des Ba-

ckends vielmehr einen Frontend-Editing-Modus darstellt. Eine separate Ansicht zur Inhalts- sowie Datensatzverwaltung analog zu TYPO3 CMS oder *Wordpress* wird nicht angeboten. Nach dem Login des Redakteurs werden zusätzlich zur normalen Webseitenansicht weitere Elemente, Menüs und Seiten angezeigt. Diese dienen zur Verwaltung der Webseite. Ein Usability-Nachteil dabei ist, dass ein Anwender in diesem Fall nicht sofort erkennt, ob er sich in der öffentlichen Frontend-Ansicht seiner Webseite oder im Backend befindet. Die Visualisierung dieser beiden Modi ist bei *Drupal* nur schwach abgegrenzt, sodass sich ein Unterschied nicht deutlich zeigt. Weiterhin besteht die Gefahr ungewollter Design-Änderung des Backends durch Frontend-CSS-Styles. Dadurch könnte das Drupal-Backend bei verschiedenen Webseiten unterschiedlich aussehen und sorgt damit für Mängel in der Benutzbarkeit. Die Analyse wird mit *Drupal* in der Version 7.35 an einem Demo-Projekt durchgeführt. In der nachfolgenden Abbildung sind eine Standard-Listenansicht des Systems sowie die Verwaltungsmenüpunkte am oberen Rand zusehen.

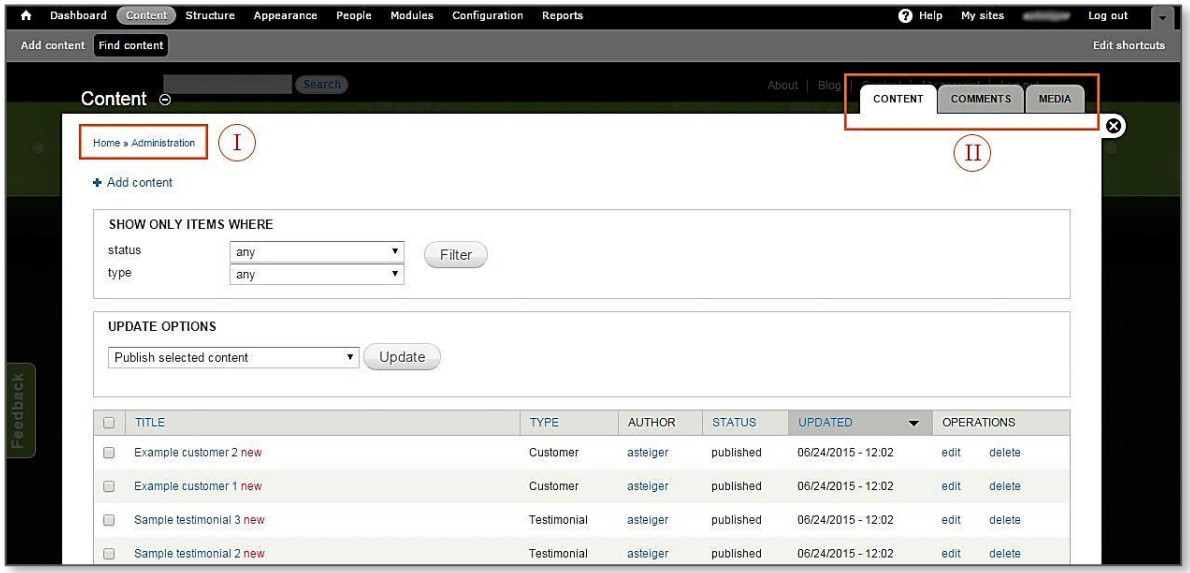


Abbildung 6: Listenansicht des CMS Drupal 7; Autor: A. Steiger 2015

Die Orientierung ist mittels Konventionen durch eine Breadcrumb (siehe I) und Reiter (siehe II) gut positioniert sowie verständlich. Der Anwender kann schnell erfassen, auf welcher Ebene er sich in der Webseite und im Backend befindet. Es fällt auf, dass die Tabellen der Inhaltselemente oder auch die Medien-/Dateiansichten keine Ordner oder Baumstrukturen aufzeigen. Alle Elementlisten können damit auf nur einer Navigationsebene verwaltet und ausschließlich über vordefinierte Filter in der Auswahl eingeschränkt werden. Das Gesamtlayout ist in der Anordnung der einzelnen Komponenten vertikal aufgebaut. Es befinden sich wenige Elemente an den Seiten. Dadurch kann ein horizontales Scrollen in den meisten Fällen unterbunden werden und schafft damit eine höhere Nutzungsqualität (siehe Kapitel 2.3.2). Diese richtet sich dabei nach dem natürlichen Workflow einer Webseite, wel-

cher vorrangig vertikales Scrollen als Konvention enthält. [Bed15] In der Datensatztabelle fehlt es bei vielen Aktionsschaltflächen an Aufforderungscharakter (siehe Kapitel 2.3.2). Alles wirkt wie eine Ansammlung von Verlinkungen und lässt das System unübersichtlich erscheinen. Eine erweiterte Ansicht für zusätzliche Spalten ist an der rechten oberen Ecke positioniert (siehe folgende Abbildung). Außerdem wird in einer anderen Tabellendarstellung eine Umschaltung für unterschiedliche Listen sowie Kacheln angeboten.

Abbildung 7: Weitere Listenansicht des CMS Drupal 7; Autor: A. Steiger, 2015

Im ersten Moment wirkt das Backend von *Drupal* relativ komplex. Nach einer kurzen Einarbeitungszeit wird die Bedienung verständlicher und die Inhalte erscheinen klar strukturiert. Die Grundfunktionen der Listenbearbeitung sind eher spärlich. Eine detaillierte Gegenüberstellung aller Kriterien wird im Anhang 2 präsentiert.

CMS Wordpress

Wordpress basiert ebenfalls auf der Skriptsprache PHP und ist derzeit das beliebteste CMS für private Webseiten und kleine Firmenauftritte. [Bre13, S.20] [Veu15, S.5] Die Untersuchung wird mit *Wordpress* in der Version 4.2.2 an einer Grundinstallation mit Dummy-Daten durchgeführt. Abbildung 8 zeigt das Backend sowie eine Standard-Datenvisualisierung des Systems in Tabellenform.

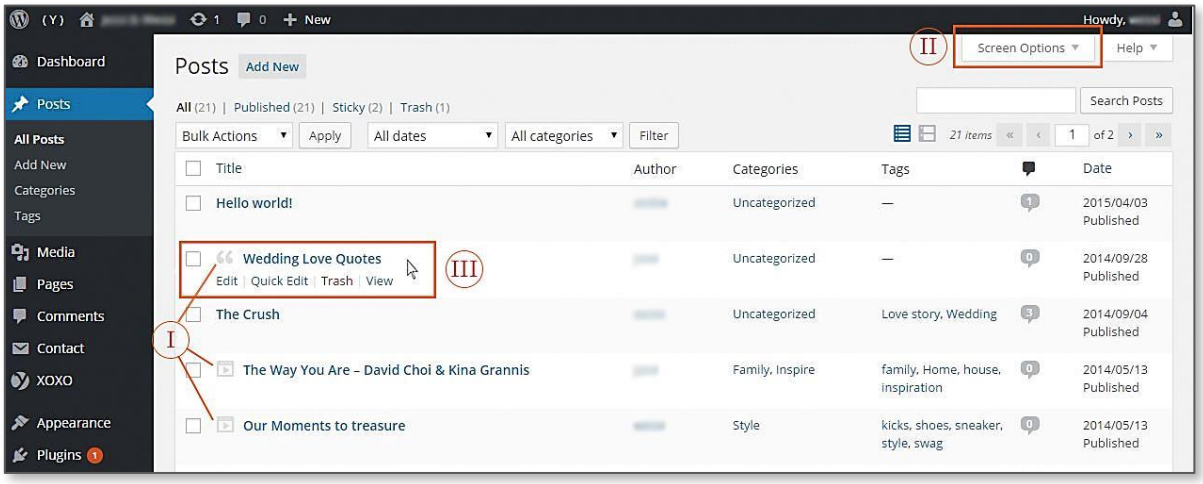


Abbildung 8: Listenansicht im Backend des CMS Wordpress 4 (Unkenntlichmachung aufgrund Datenschutz); Autor: A. Steiger 2015

Die visuelle Abgrenzung des Backends vom Frontend ist bei *Wordpress* deutlich. Es wird eine Verwaltungsoberfläche mit separatem und konsistentem Design analog zum TYPO3 CMS zur Verfügung gestellt. Schwierigkeiten gibt es bei der allgemeinen Orientierung im Backend. Auch wenn in *Wordpress* keine hierarchischen Strukturen aufgrund geringer Komplexität der Daten notwendig sind, wird in mehreren Ebenen analog zu TYPO3 CMS gearbeitet. Von der Management-Ebene kann zur Inhaltsbearbeitung in die Persistenz-Ebene gewechselt werden. Hierbei wird jedoch keine Breadcrumb oder Navigationshilfe angeboten, um dem Nutzer ein Gefühl für die Position im System zu geben. Des Weiteren werden gesetzte Filter der Ergebnisliste nicht in jedem Fall klar gekennzeichnet. Wenn beispielsweise das Kategorie-Icon vor dem Titel (siehe I) nach dem Datensatztyp filtert, wird der gesetzte Filter nicht angezeigt. Das Entfernen der aktiven Datensatzeingrenzung ist daher nicht verständlich und nur über Umwege möglich. Eigen definierte Filter können nicht erstellt werden. Beim Speichern ist die Platzierung der Schaltflächen nicht konsistent umgesetzt. Der Update-Button zum Veröffentlichen und Sichern befindet sich bei den Einstellungen links oder rechts unten bzw. in der Mitte rechts im Editiermodus von Beiträgen. Damit wird der Arbeitsfluss des unerfahrenen Anwenders unterbrochen und er muss nach der benötigten Schaltfläche suchen. Das Ein- und Ausblenden von Spalten ist über ein zusätzliches Menü möglich (siehe II). Die Anzahl der vorgegeben Spalten ist stark begrenzt. Dies bringt den Vorteil, dass der Anwender keine Tabelle mit vielen Spalten erzeugt und damit ein horizontales Scrollen notwendig macht. Jedoch wird damit der zusätzliche Informationsgehalt eingeschränkt.

Eine Besonderheit bei *Wordpress* stellt das Editieren eines Datensatzes in der Management-Ebene über Inline-Editing dar (Quick Edit: siehe III). Ohne einen Wechsel auf die zweite Ebene kann hierbei direkt die Bearbeitung stattfinden. Grundsätzlich bietet *Wordpress* in der Bedienung und in den Funktionen eine Vielzahl an Möglichkeiten und gestaltet dadurch die Arbeit mit dem Produkt angenehm. Anhang 2 bietet eine Gegenüberstellung aller Konkurrenzprodukte sowie Auflistung weiterer Details zu den Kriterien.

Tabellenkalkulationen Microsoft Excel und Google Docs

Das Tabellenkalkulationsprogramm *Microsoft Excel* bietet eine Vielzahl an Freiheiten in der Darstellungsbearbeitung und an der Manipulation. Der Umgang damit und die Funktionen sind den meisten TYPO3-Redakteuren und -Integratoren vertraut. Die Analyse wird mit *Microsoft Excel* 2010 durchgeführt. Abbildung 9 zeigt die Ansicht der Anwendung und die Möglichkeiten im Kontextmenü.

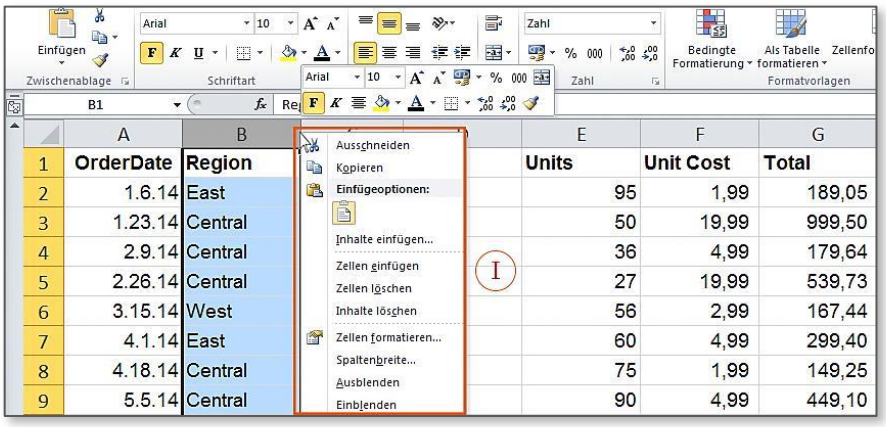


Abbildung 9: Listenansicht mit Kontextmenü in der Tabellenkalkulation Excel 2010; Autor: A. Steiger 2015

An dem umfangreichen Kontextmenü (siehe I) ist zu erkennen, dass es sich hierbei um eine Desktop-Anwendung handelt. Bei Mobilgeräten ist das Arbeiten über ein Kontextmenü nicht weit verbreitet und gestaltet sich schwierig in der Benutzung. Excel bietet für Tabellenmanipulationen und -aktionen viele Möglichkeiten, welche per Drag-and-Drop oder über das Kontextmenü angewandt werden können. Die Art der Benutzbarkeit bei Listenmanipulationen in *Excel* zeigt wichtige Ansätze für eine produktive Arbeit. Die Software ist vertikal im Layout orientiert: große Menüleiste, Datenvisualisierung und Navigation bauen sich von oben nach unten auf. Das Besondere an dieser Tabellenkalkulation ist, dass Management sowie Persistenz auf einer gemeinsamen Ebene verwaltet werden. Der Kontext wird bei Excel nie verlassen. Daher sind keine weiteren Orientierungshinweise notwendig. Allein die Navigation am unteren Rand führt den Benutzer über Reiter klar durch verschiedene Ansichten.

Die Tabellen-Webanwendung von *Google Docs* ist das Pendant zu *Excel*. Auffällig ist, dass selbst in dieser Browser-Applikation intensiv mit einem Kontextmenü gearbeitet wird. Deswegen bietet Google für Mobilgeräte zusätzliche Apps für ihre Produkte an. Abbildung 10 zeigt die Oberfläche des Systems. Die Ähnlichkeiten zu *Excel* in der Benutzung und Darstellung werden hierbei deutlich.

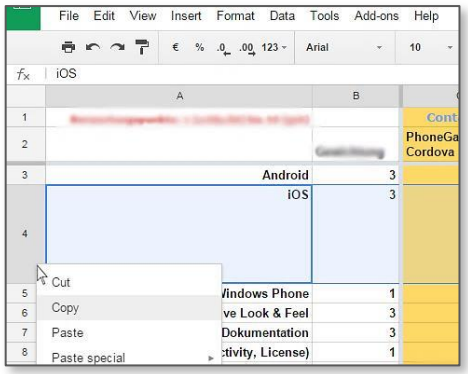


Abbildung 10: Listenansicht mit Kontextmenü in der Tabellenkalkulation Google Docs Tabellen; Autor: A. Steiger 2015

Bei den Spaltenköpfen kann das Kontextmenü nicht nur über den Rechtsklick aufgerufen werden, sondern auch per Dropdown. Diese Möglichkeit wird sichtbar, wenn sich der Anwender mit dem Cursor über dem Spaltenkopf befindet. Besonders auffällig ist das Fehlen der Schaltflächen zum Speichern von Änderungen. Google wagt hierbei einen neuen Ansatz und sichert alles automatisch ohne eine manuelle Funktion bereitzustellen. Um den Benutzer vor dem vergeblichen Suchen einer Speichermöglichkeit zu bewahren, wird ein Hinweis über das selbstständige Sichern der Änderungen am oberen Rand angezeigt. Weiterhin gibt dieser Hinweis Auskunft über den letzten Sicherungszeitpunkt.

Alle weiteren Funktionen sowie Möglichkeiten orientieren sich stark an *Excel* und werden an dieser Stelle nicht erneut aufgeführt. Eine ausführliche Auflistung zu weiteren Kriterien der Konkurrenzanalyse ist im Anhang 2 zu finden. Diese bestätigt die geringen Unterschiede zwischen *Google Docs* Tabellen und *Microsoft Excel,* obwohl ersteres eine Webanwendung und letzteres eine Desktop-Applikation ist.

Dateiverwaltungen Windows Explorer und Mac OSX Finder

Der Seitenbaum im TYPO3 CMS ist eines der bekanntesten Komponenten des Systems. Dieses Konzept ist bereits durch die Dateiverwaltung in einem Desktop-Betriebssystem bekannt. Microsoft Windows verwendet beim Explorer eine hierarchisch aufgebaute Ordnerstruktur analog zum Seitenbaum. Die Untersuchung wird mit dem *Explorer* in Windows 7 durchgeführt. Die Ansicht dieses Programms ist zweitgeteilt. Wie Abbildung 11 zeigt, befindet sich auf der linken Seite eine hierarchische Ordnerstruktur und auf der rechten eine Listenansicht der enthaltenen Dateien sowie Unterordner.

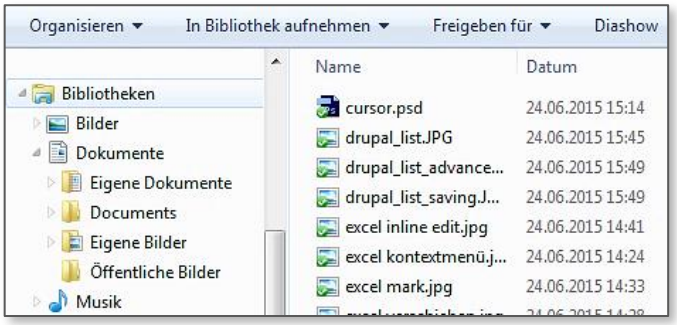


Abbildung 11: Listenansicht mit Ordnerbaum des Explorers in Windows 7; Autor: A. Steiger 2015

Diese strukturierte Art der Navigation hat sich über viele Jahre bewährt und stellt verständlich die hierarchischen Ebenen dar. Die Funktionen der Datenvisualisierung im *Explorer* sind erkennbar und flexibel in der Verwendung. Jedoch ist die Option zum Setzen von komplexen Filtern für den Anwen-

der versteckt. Der Benutzer erhält erst eine Schaltfläche für erweiterte Filter, wenn er den Cursor über einen Spaltenkopf bewegt. Jedoch zeigt diese nicht deutlich, welche Funktion hinter dem Button steckt. Da der Spaltenkopf die Möglichkeit der Sortierung (Linksklick), der Spalten-Einblendung (Rechtsklick) sowie der Filterung (Dropdown) besitzt, besteht hierbei die Gefahr der Verwechslung und des Nichtauffindens einer dieser Funktionen. Wird ein Bereichsfilter einer Spalte unterstützt, können Werte über ein Dropdown-Feld definiert werden. Die Kennzeichnung der gesetzten Filter wird mittels eines schlecht wahrnehmbaren und wenig aussagekräftigen Haken-Symbols angezeigt. Das Verschieben sowie Auswählen von Spalten, die Größenanpassung, das Filtern und Sortieren sind flexibel anwendbar. Ebenso erhält der Nutzer an unterschiedlichen Stellen Orientierungshinweise und Navigationsmöglichkeiten. Eine Änderung der Datensatzeigenschaften erfolgt über das Öffnen der Datei und folglich dem Betreten der Persistenz-Ebene. Lediglich der Name des Elementes kann in der Management-Ebene über Inline-Editing direkt geändert werden.

Der *Finder* von Mac OSX bringt ähnliche Ergebnisse der zu untersuchenden Kriterien. Jedoch ist im direkten Vergleich der Funktionsumfang des *Finders* höher als der des *Explorers*. Eine besondere Beachtung findet hierbei die komplexe Filtermöglichkeit. Mit den sogenannten Smart Folder können vordefinierte Einschränkungen der Anzeige durch diverse Vorgaben ausgewählt werden. Eine detaillierte Beschreibung dieser Funktion wird in Kapitel 6 erarbeitet. Weiterhin besteht ein auffälliger Unterschied bei der hierarchischen Darstellung. Diese befindet sich beim *Finder* im rechten Bereich des Fensters (siehe I, Abbildung 12). Im *Explorer* hingegen ist diese strukturierte Darstellung im linken Rahmen integriert (siehe II, Abbildung 12). Nachfolgend ist ein Screenshot des *Finders* abgebildet, der diesen Sachverhalt verdeutlicht.

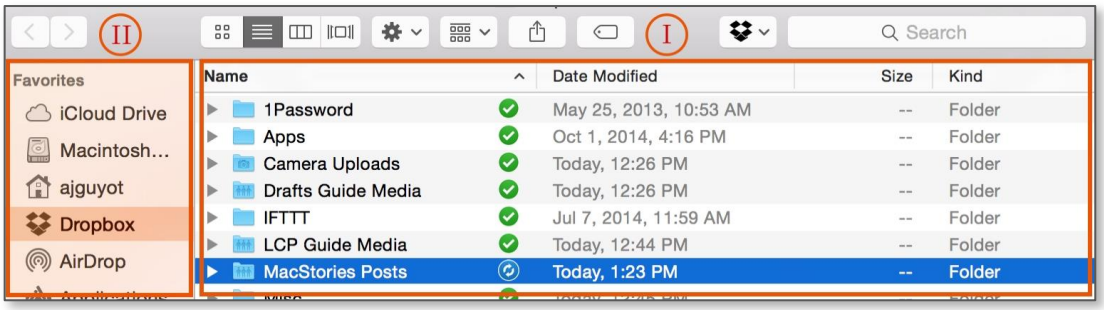


Abbildung 12: Listenansicht des Finders in Mac OSX;

Quelle: http://www.macstories.net/mac/extensibility-and-automation-changes-in-os-x-yosemite/ (Aufgerufen am 13.07.2015)

Bei der Betrachtung des Layouts beider Dateiverwaltungen fällt auf, dass diese sowohl horizontal als auch vertikal aufgeteilt sind. Der Aufbau ist abhängig von der jeweiligen Darstellung. Weitere Kriterien werden ebenfalls in der Gegenüberstellung im Anhang 2 genannt.

3.3.3 Verwendung von App-ähnlichen Konzepten

Unabhängig von den analysierten Datenvisualisierungen im vorherigen Kapitel werden nachfolgend die Content Management Systeme *Drupal* und *Wordpress* im Allgemeinen betrachtet. Hierbei wird untersucht, ob ein aufgabenfokussiertes Arbeiten mit separaten und kompakten Modulkomponenten im System möglich ist.

Bei *Wordpress* findet die Simplizität des Backends Beachtung. Die Datenvisualisierung der einzelnen Module weist durchaus fokussierte Inhalte auf. Je Modul werden Listen vom gleichen Datentyp ausgegeben. Auch wenn eine Tabelle Bilder, Videos oder Texte beinhaltet, sind alle jeweils vom Typ Beitrag. Es liegt nie der Fall vor, dass Formulare gemeinsam mit Beiträgen und Seiten in einer Liste kombiniert werden. Somit sind die Aufgaben der einzelnen *Wordpress*-Module klar vorgegeben. Wenn ein Benutzer beispielsweise Veranstaltungen bearbeiten möchte, wechselt er in das fokussierte Modul „Veranstaltungen“ und erhält dort eine Liste mit lediglich relevanten Einträgen. An dieser Stelle sei erwähnt, dass bei *Wordpress* grundsätzlich einfache Elementansichten ausgegeben werden. Komplexe Datenvisualisierungen aus unterschiedlichen Datenbanktabellen mit vielen Informationen und komplexen Relationen sind in der Standardinstallation nicht berücksichtigt.

Drupal zeigt sich im Verhalten ähnlich. Auch hier ist die Kombination mehrerer Datentypen in einer Tabelle nicht vorzufinden. Über Reiter können die Listen der unterschiedlichen Datentypen aufgerufen werden. In der Grundinstallation sind nur wenige listenartige Datenvisualisierungen vorhanden. Daher kann keine konkrete Aussage über ein App-ähnliches Verhalten im *Drupal*-Backend getroffen werden. Grundsätzlich sind bei beiden Content Management Systemen Ansätze für ein fokussiertes Arbeiten zu erkennen. Die Bezeichnung „App“ wird jedoch in keinem Fall verwendet.

3.4 Konzeptvorschläge der TYPO3-Community

3.4.1 Focus Apps und andere App-Konzepte

Anhang 13 beinhaltet unter Punkt 2 die Präsentationsfolien eines Vortrages der TYPO3 User Experience Week 2014 zum Thema: TYPO3 Backend-Modul-API. In dieser Ideenvorstellung kristallisiert sich die Arbeit mit fokussierten Apps im Backend heraus. Damit soll es für TYPO3-Anwender einfacher in der Konfiguration und Nutzung von kompakten Modulen werden. Der Ursprung der Idee liegt bei den immer wiederkehrenden Konzepten der verschiedenen Backend-Komponenten. 80 Prozent dieser

arbeiten nach dem gleichen Prinzip (siehe S.3, Anhang 13: „Präsentation der User Experience Week 2014 zum Thema: Focus Apps“). Entwickler müssen oft ein Backend Modul von Grund auf entwickeln, obwohl die Funktion des Moduls schon existiert. Es werden lediglich andere Daten abgefragt. Die sogenannten Focus Apps sollen einfach und schnell zu erstellen sein und eine bessere Usability bieten. Es werden folgende Komponenten dafür vorgeschlagen:

- Menüleiste
- Ordner-/Seitenbaum
- Filter
- Ergebniskomponente

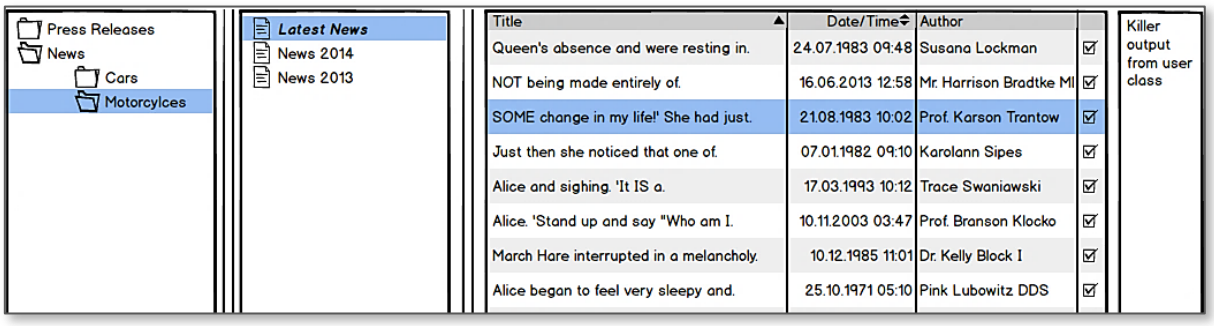


Abbildung 13: Wireframe zum ersten Backend-App-Konzept einer Listenansicht; Quelle: Anhang 13[10]

Die *Menüleiste* soll dem Nutzer die Möglichkeit bieten, zwischen verschiedenen Ansichten zu wechseln. Abbildung 13 zeigt in der linken Spalte eine hierarchische *Ordner-/Seitenstruktur*. Hierbei werden lediglich die relevanten Einträge gefiltert dargestellt. In der zweiten Spalte von links befindet sich eine Liste mit gespeicherten Filtern. Die darauf folgende Spalte hat die größte Breite und stellt die *Ergebniskomponente* dar. Hier befinden sich die Datensätze, die weiter bearbeitet, sortiert oder gefiltert werden können.

Eine weitere Idee des Vortrags der TYPO3 User Experience Week 2014 beschreibt die technische Umsetzung der Konfiguration einer Backend App. Hierbei sollen andere Technologien neben TypoScript und PHP in Betracht gezogen werden. In der Präsentation wurde vorgeschlagen, die Konfigurationssprache YAML zu verwenden.

Als Fazit des Focus-App-Konzeptes ist die Vereinfachung und Vereinheitlichung von Backend Modulen zu erkennen sowie eine Flexibilität in der App-Erstellung durch einzelne Komponenten. Besonders

[10] „Präsentation der User Experience Week 2014 zum Thema: Focus Apps“, S. 13

hervorgehoben wird die Konfiguration, welche über Konvention automatisiert vom System erkannt werden soll (siehe S.11, Anhang 13: „Präsentation der User Experience Week 2014 zum Thema: Focus Apps“). Damit kann die Notwendigkeit zusätzlicher Programmierung umgangen werden. Jedoch erfordert eine automatisierte Registrierung in TYPO3 CMS ein komplexes Cache-Speicher-Konzept sowie diverse Performance-Analysen bei einer Vielzahl installierter TYPO3 Extensions[11]. Dies würde die Dauer des automatisierten Suchprozesses beeinflussen und bedarf damit eines performanten Lösungsansatzes.

Neben dem eben beschriebenen Vortrag gibt es weitere Lösungen, welche App-ähnliche Verhalten aufweisen. Ein Team der TYPO3-Community implementierte eine Extension, welche oben benannte Möglichkeiten sowie einige der erwähnten Komponenten mit den vorhandenen Kernfunktionen des TYPO3 CMS 6.2 LTS umsetzt. Sie kombinierten Filter- und Sortierfunktionen mit einer fokussierten Listenansicht auf einer Management-Ebene (siehe auch: Kapitel 3.1.2). Die zweite Ebene wird lediglich für die Datensatzbearbeitung benötigt. Der nachfolgende Screenshot zeigt das kompakte Backend Modul dieser Erweiterung.

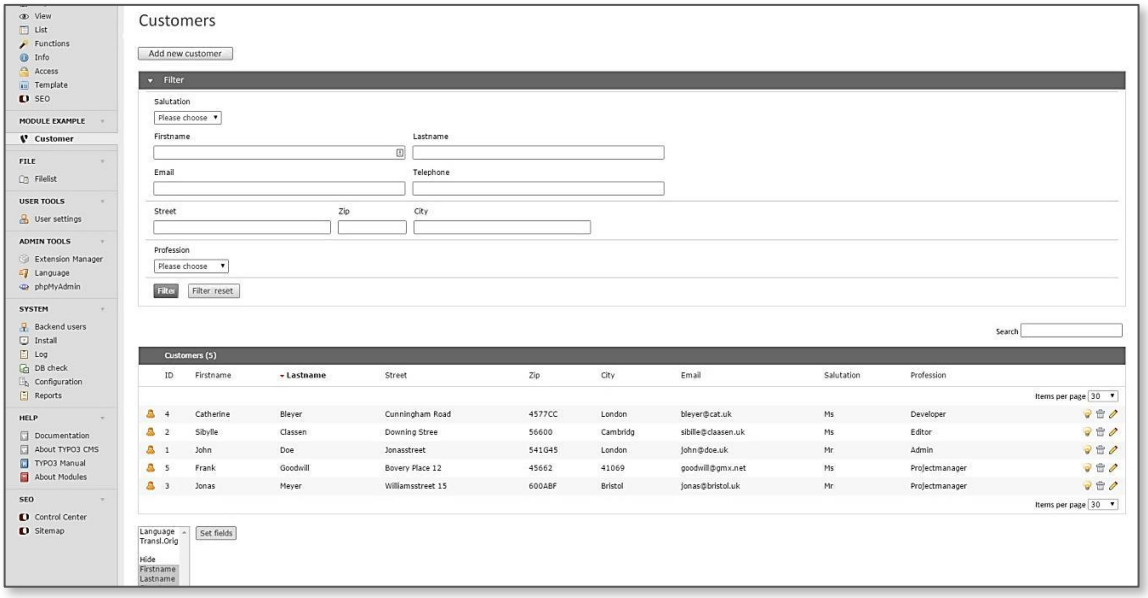


Abbildung 14: Screenshot eines Backend Moduls mit fokussierter Aufgabenbearbeitung in TYPO3 CMS 6.2 LTS; Autor: G. Schulmeister, 2015

[11] Englisch für: Erweiterungen

3.4.2 Gedankensammlung zu TYPO3 Apps

Daniel Siepmann, Entwickler in der TYPO3-Community, hat sich Ende des Jahres 2014 intensiv mit den ersten Konzepten der Focus Apps auseinandergesetzt. Er sammelte eine Vielzahl von Fakten basierend auf der ersten Präsentation (siehe Anhang 13: „Präsentation der User Experience Week 2014 zum Thema: Focus Apps“) und änderte in seiner Ausarbeitung den Terminus in „TYPO3 Apps“. Seine Zusammenfassung beinhaltet mehrere, nach Themen sortierte Textdateien und beschreibt weitere Ideen, die über die Kernabhandlung der vorliegenden Arbeit hinausgehen würde. Aus diesem Grund wird im Folgenden nur eine Aufstellung der relevanten Themen aufgezeigt.

Die Gedankensammlung ist technisch orientiert und zeigt Vorschläge über die Kernfunktionen von TYPO3 Apps. Es wird analysiert, welche bestehenden Backend Module von TYPO3 durch Apps ersetzt werden könnten. Daraus resultiert, alle vorhandenen Systemmodule zukünftig durch Apps abzulösen. Zudem werden die Konfiguration und die Anforderungen daran beschrieben. Das Konfigurieren einer Backend App sollte folgende Eigenschaften aufweisen:

- Editier- und erstellbar mit einfachem Texteditor
- Versionierbar
- Generierbar über externe Programme, wie z.B. Shell-Skripte oder Entwicklungsumgebungen
- Schnell umwandelbar (parsen)
- Verständlich und lesbar

In der weiteren Analyse werden verschiedene Formate zur Konfiguration diskutiert. Zur Auswahl stehen:

- PlantUML
- JSON
- XML
- YAML

PlantUML wird verwendet, um beispielsweise UML-Diagramme in einer Java-Anwendung zu erstellen. Der Vorteil dieses Formats besteht darin, dass aus dieser Konfiguration ein vollständiges UML-Diagramm ohne zusätzliche Arbeit erzeugt werden kann. Es wäre für diesen Fall zwar geeignet, nur ist die Unterstützung und Verbreitung in den Entwicklungsumgebungen und Texteditoren rar. *JSON* ist ein bekanntes Format in Zusammenhang mit JavaScript, welches auch von PHP verarbeitet und umgewandelt werden kann. *XML* ist ebenfalls weit verbreitet und wird schon an diversen Stellen im TYPO3-System verwendet. Jedoch gilt dieses Format als schwergewichtig, da viel Code für wenig Informationsgehalt benötigt wird. *YAML* ist heutzutage eine sehr beliebte Konfigurationssprache. Sie

hat ein unabhängiges Format und dient ausschließlich der Konfiguration. Allerdings wird hier ein zusätzlicher Parser zur Umwandlung benötigt und die Syntax ist streng sowie fehleranfällig. Als bevorzugtes Format zur Konfiguration von TYPO3 Apps entscheidet sich Daniel Siepmann daher in seiner Ausarbeitung für *JSON*. Was hierbei in Bezug auf Konfigurationen mit TYPO3 CMS zusätzlich Beachtung finden sollte, ist TypoScript. Diese gilt als interne Konfigurationssprache des TYPO3 CMS. [Wik15a] Jedoch ist TypoScript durch dessen Komplexität und die Möglichkeit der Integration von Applikationslogik sehr umfangreich und schwer zu durchdringen. Weiterhin erfordert das Parsen von TypoScript relativ viel Rechenzeit bzw. -leistung und damit einen nicht vernachlässigbaren Zeitaufwand. Gegenüber einer nativen Notation als Array oder als *JSON*, müssen bei TypoScript zusätzlich Bedingungen verarbeitet und evtl. benötigte Objekte erzeugt werden. Daher sollten effizientere Alternativen für eine Backend-App-Konfiguration im TYPO3 CMS vorgesehen werden.

Die Entscheidung für das *JSON*-Format ist zu hinterfragen. Als effiziente und leichtgewichtige Lösungen stehen *JSON* und *YAML* zur Diskussion. Da *JSON* grundsätzlich als Format zum Datenaustausch konzipiert ist und *YAML* dagegen als unabhängige Datenserialisierung einer vereinfachten Konfiguration entspricht, wird im Ergebnis dieser Analyse *YAML* als Konfigurationsformat vorgesehen. [Ecm99] [Eva09]

Darüber hinaus geht Siepmann auf die einzelnen Komponenten ein. Diese sollen eigenständig sein und flexibel in der Sichtbarkeit sowie Positionierung. Ein weiterer Aspekt setzt sich mit der Filter-Thematik auseinander. Es werden Wünsche über komplexe Filtermöglichkeiten sowie eine eigene Abfragesprache erwähnt. Hierbei orientiert sich Siepmann an der Projektmanagementsoftware Jira. Diese zeigt eine vielschichtige Filterung, die auch manuell mittels einer eigenen Abfragesprache in der erweiterten Ansicht angepasst werden kann. Die Vorstellungen der Filtervorgaben beschreiben aussagekräftige Bezeichnungen mit intelligenter Anwendung. Am Beispiel von Neuigkeiten einer Webseite könnten die Voreinstellung „Die letzten 5 Top-Neuigkeiten“ oder „Alle selbsterstellten Neuigkeiten“ sein. Diese werden konfiguriert und dann vom System angeboten. Ebenso besteht auch die Möglichkeit zum Setzen und Speichern eigen definierter Filter. [Sie14]

Es sei an dieser Stelle erwähnt, dass es sich bei der Gedankensammlung nicht um beschlossene Fakten handelt, sondern um Ideen sowie Vorschläge eines einzelnen Mitglieds der TYPO3-Community. Diese bekräftigen und unterstützen die Konzeptentwicklung der vorliegenden Arbeit.

3.5 Terminologie

Die Terminologie beim Thema TYPO3 Backend Apps ist essentiell für das Verständnis des Konzeptes. Gerade da es bei dieser Thematik nicht um die gebräuchlichen mobilen Apps oder Desktop Applikationen geht, sollten die Begrifflichkeiten hinterfragt werden. Um dies zu ermöglichen, muss eine genaue Beschreibung der Idee mittels Definition und Metaphern erarbeitet werden. Die Idee von Backend Apps ist es, einzelne CMS-Komponenten, die für mehrere Aufgaben zuständig sind, zu vereinfachen. Dies kann über das Aufteilen auf mehrere Komponenten mit fokussierten Aufgaben geschehen. Diese Komponenten sind darauf spezialisiert, den Anwender beim Erreichen eines definierten Zieles bestmöglich zu unterstützen. Dabei sollte der Fokus innerhalb einer Backend App nicht verlassen werden. Eine TYPO3 Backend App ist jedoch kein separates Programm, das von einem Startbildschirm eines Gerätes ausgeführt wird. Es handelt sich dabei vielmehr um eine Web-Applikation im Browser. Als bildhafte Umschreibung für diesen Sachverhalt kann als praktisches Beispiel metaphorisch der Baumarkt dienen. Dieser ist ein in sich geschlossenes Gebäude mit mehreren Abteilungen analog zu einem CMS mit einzelnen Modulen. Jede Abteilung repräsentiert ein fokussiertes Themengebiet und bietet dementsprechend Produkte an. Die Besucher eines Baumarktes orientieren sich am Eingang an den Abteilungsbezeichnungen und begeben sich danach auf den Weg, um ihr verfolgtes Ziel zu erreichen. Hierbei entstehen oft Probleme, da sich die Produkte nicht in einer geeigneten Abteilung befinden. Das Ergebnis ist Orientierungslosigkeit und Unzufriedenheit des Kunden. Im Umkehrschluss bedeutet dies, dass selbst Bezeichnungen von aufgabenfokussierten Modulen sehr präzise und aussagekräftig sein müssen. Ein Anwender sollte sofort wissen, mit welcher TYPO3 Backend App er welche Aufgabe erledigen kann, ohne sich im System zu verirren und unnötig lange zu suchen. Weiterhin ist es wichtig beim Verlassen des fokussierten Kontextes einen Übergang mittels Querverlinkungen zwischen den anderen Apps zu schaffen.

Die naheliegende Bezeichnung von kompakten Komponenten mit fokussierter Aufgabenbearbeitung lautet „App“. Dies ist ein Trend-Wort und bekannt in dessen Bedeutung. Es beschreibt den Sachverhalt des geplanten Entwurfes von TYPO3 Backend Apps. Im Kapitel 2.1 wurde eine ausführliche Definition des Begriffes vorgenommen, welche erkennen lässt, dass die gedankliche Verbindung bei dem Wort „App“ sich zuerst in Richtung mobile Apps oder auch Desktop-Apps bewegt. Das liegt darin begründet, dass der Ursprung einer App das mobile Betriebssystem ist und daher diese Benennung geprägt hat. Auch wenn die Firma Google in ihrer Browseranwendung Chrome Apps verwendet, sind

andere Hersteller eher vorsichtig in der Verwendung des Begriffs. Der Browser Firefox bezeichnet beispielsweise weiterhin die Erweiterungen mit Add-ons[12] und Microsoft änderte die ursprüngliche online-fähige Office-Bezeichnung von “Office Web Apps” zu “Office Online”. Die allgemeine Beschreibung aus Kapitel 2.1 lässt weiterhin erkennen, dass auch eine App in einer Software analog zu Chrome-Erweiterungen die Möglichkeit der Simulation einer nativen App bietet. Das Konzept von TYPO3 Backend Apps sollte daher zusätzlich einen Anreiz für die Entwicklung von REST-Schnittstellen in TYPO3 CMS geben. Mit dieser Technologie ist es möglich, mittels einer definierten Web-Adresse genau einen speziellen Seiteninhalt aufzurufen. [Wik15c] Damit könnten zukünftig Verknüpfungen von bestimmten Backend Modulen oder Backend Apps auf dem Desktop bzw. Home-Screen erstellt werden.

Um der Assoziation zu mobilen Apps entgegenzuwirken, ist es in jedem Fall notwendig, ein Begriffszusatz zu verwenden. Dieser sollte die Umgebung, in der die App arbeitet, genauer umschreiben. Beispiele hierfür wären „Online App“, „Web App“, „CMS App“, „TYPO3 App“, „Backend App“ oder „Modul App“. Die ersten zwei genannten Bezeichnungen sind sehr allgemein und führen ebenso zu einer Verwechslungsgefahr, da es bei mobilen Apps gleiche Benennungen gibt (Siehe Kapitel 2.1). Bei „CMS App“ und „TYPO3 App“ liegt zwar der korrekte Kontext vor, jedoch wird nicht ersichtlich, ob diese das CMS startet oder sich darin befindet. Die letzten beiden Bezeichnungen benennen sehr gut die Umgebung und lenken damit die Gedanken in die richtige Assoziation. „Modul App“ stellt jedoch eher zwei Synonyme dar, da eine App ebenfalls ein kompaktes Modul ist. Daher sind die Worte „Backend App“ in Kombination mit „TYPO3“ die treffendsten Benennungen. Hierbei werden zuerst das System und danach die allgemeine Umgebung genannt.

3.6 Anwendungsfälle

Um das Konzept in allen wichtigen Bereichen und in der Komplexität zu erfassen, werden spezielle Anwendungsfälle gewählt. Die Konzepte der Integration, Datenvisualisierung und des Filters finden anhand der zwei folgenden Beispiele statt:

1. Veranstaltungsmanagement
2. Dateiverwaltung

[12] Englisch für: Erweiterung

Beim Veranstaltungsmanagement soll eine TYPO3 Backend App für Veranstaltungen konzipiert werden, welche die Aufgaben der Information und Datendarstellung inklusive Bearbeitung einzelner Datensätze bietet. Das Entity-Relationship-Modell (ERM) im Anhang 3 zeigt die Entität der Veranstaltungstabelle sowie deren Felder und Beziehungen. Die Backend App soll auf diese Tabelle zugreifen. Die TYPO3 Extension für das Veranstaltungsmanagement trägt die Bezeichnung: event_manager.

Bei der Dateiverwaltung soll eine TYPO3 Backend App zur Visualisierung verschiedener Dateitypen vorgesehen werden. Dieser Anwendungsfall wird durch das ERM im Anhang 4 beschrieben. Als Quelle für diese Backend App, dient die TYPO3-Datenbanktabelle „sys_file“. Die verschiedenen Dateitypen wie Medien oder Dokumente sollen in der Visualisierung getrennt dargestellt werden.

4 Backend Apps im TYPO3 CMS

4.1 Listenmodul als TYPO3 Backend App

Im Kapitel 3.1.1 wurde die Bedeutung von Aufgaben und Modulen in einem CMS herausgearbeitet. In Bezug darauf zeigt Kapitel 2.1 und 3.5, wie wichtig die Konzentration auf die Kernaufgaben einer App ist. Die bisherigen Analysen zeigen, dass der Gedanke von Daniel Siepmann (siehe Kapitel 3.4.2), die System-Module durch Apps zu ersetzen, nicht ohne Weiteres möglich ist. Eine direkte Überführung der Aufgaben eines Moduls zu einer App würde den Grundprinzipien sowie der Bedeutung von fokussiertem Arbeiten widersprechen. Am praktischen Beispiel des mobilen Betriebssystems iOS von Apple wird die Aufteilung von Aufgabenbereichen verdeutlicht. Beim Arbeiten mit Fotos auf einem iPhone gibt es verschiedene Tätigkeiten zu erledigen. Diese werden aufgabenfokussiert auf drei separate Apps aufgeteilt. Zum Erstellen von Fotos dient die Kamera-App. Das Verwalten von Bildern kann mit der Foto-App getätigt werden und die Einstellungen sind in der Einstellungs-App des Systems zu finden.

Ein Kopieren aller Möglichkeiten des Listenmoduls in eine Backend App ist daher nicht zweckmäßig. In der vorliegenden Arbeit müssen die einzelnen Aufgabenbereiche in Bezug auf die Anwendungsfälle (siehe Kapitel 3.6) herauskristallisiert werden. Da ein Kernaspekt die Datenvisualisierung ist, wird eine flexible sowie benutzerfreundliche Anzeige von Datensatzinformationen in einer TYPO3 Backend App vorgesehen. Dies stellt einen von mehreren Aufgabenteilen des Listenmoduls dar. Das Konzept der Trennung von Management- und Persistenz-Ebene (siehe Kapitel 3.1.2) wird übernommen. Features der Massendatenbearbeitung und Selektion finden an dieser Stelle keinen Platz, da es den Fokus verlassen würde.

Beim Anwendungsfall des Veranstaltungsmanagements (siehe Kapitel 3.6) greifen die Datensätze der Veranstaltungstabelle in der Datenbank auf weitere Tabellen wie beispielsweise Künstler oder Teilnehmer zu. Bei einer Veranstaltungs-Backend-App sollte die Visualisierung und Bearbeitung lediglich von Veranstaltungen im Fokus stehen. Daher befindet sich die Bearbeitung von Teilnehmern oder Künstlern nicht im Aufgabenbereich und muss in separaten Apps erfolgen.

Bei der TYPO3 Backend App für die Dateiverwaltung, muss eine solche Aufgabentrennung nicht vorgenommen werden. Hierbei handelt es sich bei allen Datensätzen um Dateien und Informationen, die im gleichen Kontext stehen.

4.2 Systemintegration

Um TYPO3 CMS erweitern oder speziell Backend Module hinzufügen zu können, werden sogenannte Extensions benötigt. Diese basieren heutzutage auf dem Framework[13] Extbase. Für TYPO3 CMS liefert dieses einen Backport[14], um das existierende PHP-Framework Flow und das damit verbundene Model-View-Controller-Entwurfsmuster nutzen zu können. Weiterhin kann dadurch anwendungsdomänenorientiert entwickelt werden. [Lob14a, S.25 f.] Ein Backport schafft im Allgemeinen eine Kompatibilität zu anderen oder älteren Architekturen durch Software-Modifikation. [Wik13] Mittels einer TYPO3 Extension können Backend Module oder Frontend Plugins im System registriert und bereitgestellt werden. Ein Frontend Plugin ist in TYPO3 CMS ein spezielles Inhaltselement mit selbst definierter Logik und flexiblen Ausgabemöglichkeiten auf der Webseite bzw. dem Frontend. [Typ15] TYPO3 Backend Apps sollen eine weitere, zusätzlich registrierbare Einheit einer Erweiterung werden. Eine automatische Registrierung im System, ausschließlich über Namenskonventionen von Dateien, ist in diesem Zusammenhang nicht einfach möglich. Dies wurde in Kapitel 3.4.1 gezeigt. Aufgrund des Umfanges wird daher im weiteren Konzept der Weg über die manuelle Zuweisung von Backend Apps analog zum Registrierungsprozess von Frontend Plugins und Backend Modulen gewählt.

Grundsätzlich ist eine TYPO3 Backend App ein Bestandteil einer Extension. Die folgende Übersicht verdeutlicht diesen Sachverhalt.

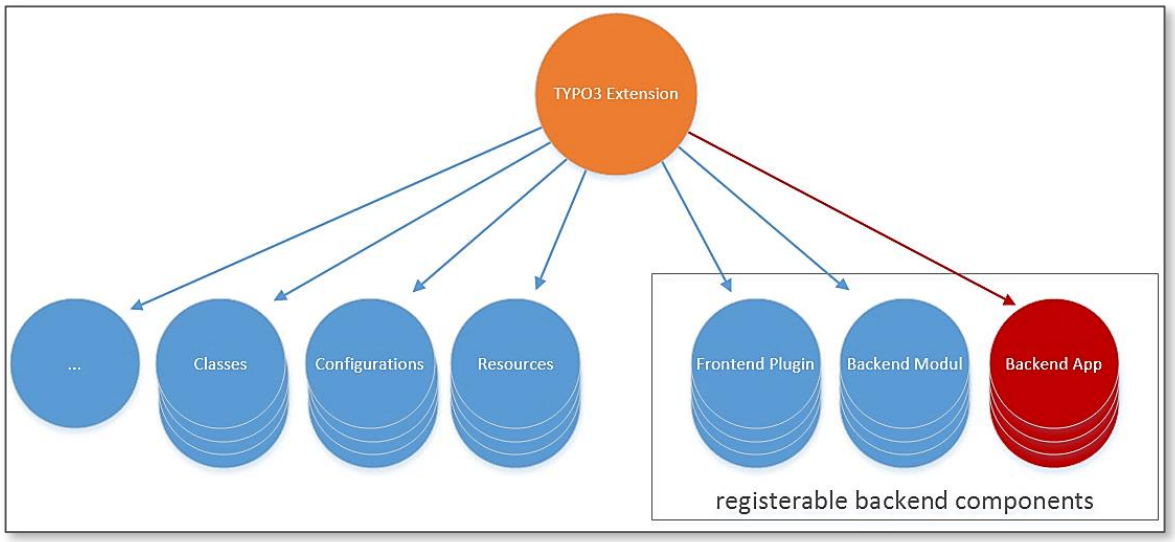


Abbildung 15: Integrationsübersicht der TYPO3 Backend Apps in einer Extension; Autor: A. Steiger, 2015

[13] Englisch für: Gerüst
[14] Englisch für: zurück portieren

Um die Systemintegration besser zu veranschaulichen, werden die nachfolgenden Beschreibungen beispielhaft am Anwendungsfall des Veranstaltungsmanagements durchgeführt. „Apps“ wird als Namenskonvention für eine TYPO3-Backend-App-Ordnerstruktur innerhalb einer Extension vorgesehen. Im Folgenden wird dies in einem Screenshot dargestellt.

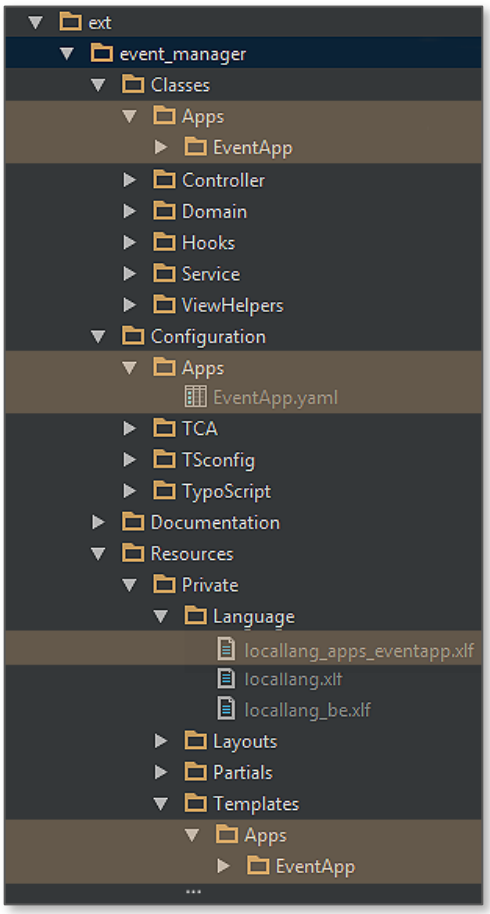


Abbildung 16: Namenskonventionen der Ordnerstruktur von TYPO3 Backend Apps; Autor: A. Steiger, 2015

Diese Ordnerstruktur gliedert sich in die bisherige Extbase-Ordnung (vgl. [Typ15a]) mit ein. Eine neue Hierarchie in der ersten Ebene einer Erweiterung aufzubauen, würde die Dateien einer Backend-App zu stark abkapseln und nicht dem Quasi-Standard des TYPO3 CMS entsprechen.

Der Registrierungsprozess einer TYPO3 Backend App wird von einer zentralen Komponente aus gesteuert. Dieser wird bei jedem Request zum Laden des TYPO3 Backends neu angestoßen. Um eine Backend App im System zu registrieren, muss diese der zentralen Komponente bekannt gemacht werden. Daraufhin wird sich um die Darstellung und die aufzurufenden Handler[15] im seitlichen Mo-

[15] Englisch für: Steuerungseinheit

dulbereich des Backends gekümmert. Die zentrale Komponente für die TYPO3 Backend Apps ist eine zusätzliche Extension und erhält die Bezeichnung: App Engine[16]. Um die Logik und Aufgaben dieser Erweiterung detailliert zu nennen, sind vorerst weitere Elemente einer Backend App zu definieren.

4.3 Layout und Komponenten

Eine Backend App besteht neben den in 4.2 genannten Elementen aus einem UI mit einem vordefiniertem Layout. Um das UI einer Backend App in TYPO3 CMS flexibel zu halten, ist dieses in einzelne Komponenten aufgeteilt. Aufbauend auf dem Kapitel 3.4.1 sind daher folgende Kernelemente für eine TYPO3 Backend App vorgesehen:

- Menü
- Titel
- Presets (Vorlagen vordefinierter und gespeicherter Ansichten)
- Hierarchische Seitenstruktur (Seitenbaum)
- Filter
- Ergebnis

Grundsätzlich kann eine App mindestens aus dem Menü und der Ergebniskomponente bestehen. Alle weiteren genannten sowie eigen definierten Elemente können bei Bedarf ergänzt werden. Das Layout einer App ist nach den Konventionen des natürlichen Workflows einer Webseite von oben nach unten und der Leserichtung von links nach rechts aufgebaut. [Bed15] Das Kernelement einer Backend App ist die Ergebniskomponente. Daher wird diese im rechten, unteren Bereich platziert und kann zur Eingabe oder Ausgabe dienen. Bei einer Backend App für Formulare bietet die Ergebniskomponente beispielsweise eine gerasterte Arbeitsfläche zum Erstellen eines Formulars, welche per Drag-and-Drop mit Formularfeldern und Labels gefüllt werden kann. Bei den Anwendungsfällen aus Kapitel 3.6 hingegen soll das Resultat die Visualisierung der gefilterten Datensätze sein. Das Layout kann durch weitere Einheiten im linken Bereich sowie oberhalb der Ergebniskomponente erweitert werden. Das nachfolgende Wireframe stellt diese Sektoren dar.

[16] Englisch für: Antriebs- / Funktionseinheit

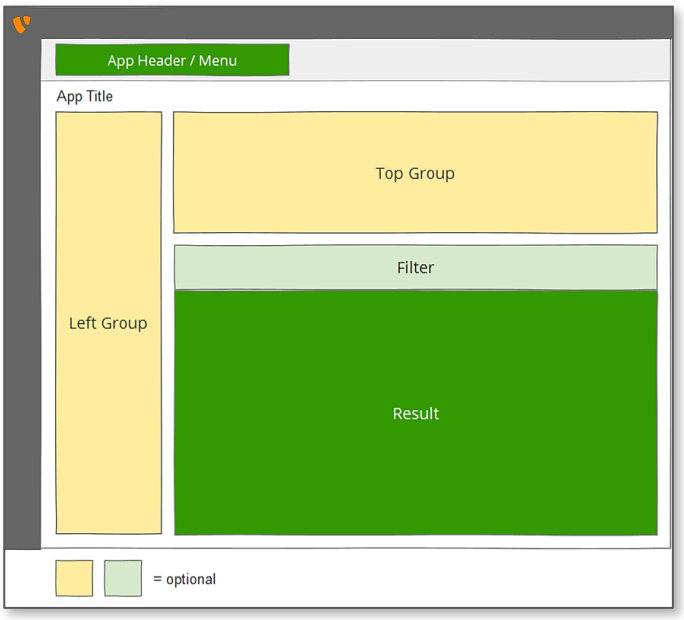


Abbildung 17: Wireframe des Backend-App-Layouts mit flexibel erweiterbaren Bereichen; Autor: A. Steiger, 2015

Auffällig ist hierbei die separate Filterkomponente. Die Konkurrenzanalyse aus Kapitel 3.3.2 zeigte in diesem Zusammenhang, dass die Filtereinheit für eine Liste nach heutigen Konventionen stets oberhalb der Datensatzvisualisierung positioniert wird. Aufgrund dessen ist das Filterelement ein Spezialfall und kann nicht variabel im oberen oder seitlichen Bereich platziert werden. Wird der Filter in einer Backend App aktiviert, ist die Position, direkt über der Ergebniskomponente, festgelegt. Eine Flexibilität der Platzierung würde hierbei zu eventuellen Usability-Problemen führen. Aus diesen Gründen sind ebenfalls die flexiblen Bereiche vordefiniert und geben damit eine Hilfestellung für ein benutzerfreundliches UI. In der nachfolgenden Abbildung werden beispielhaft einige Komponenten in den erweiterbaren Sektoren positioniert.

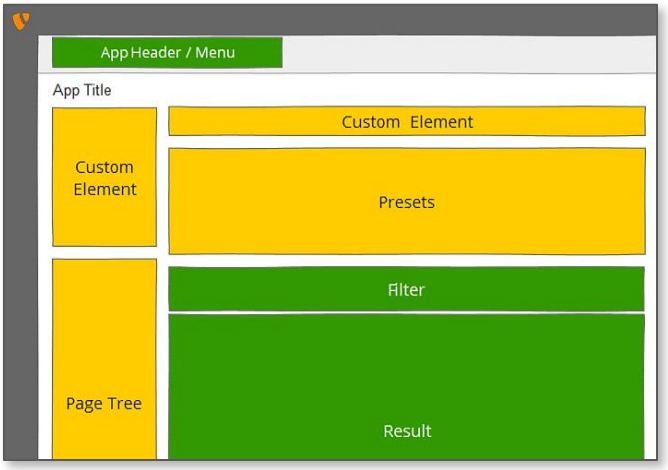


Abbildung 18: Wireframe des Backend-App-Layouts mit beispielhaften Komponenten; Autor: A. Steiger, 2015

Die aufgeführten Komponenten „Custom-Element“ stellen Objekte dar, welche mit eigen definierten Inhalten gefüllt werden können. Diese veranschaulichen hierbei die umfangreichen Möglichkeiten der flexiblen Sektoren und werden im weiteren Verlauf nicht näher in das Konzept mit einbezogen. Je nach Aufgabe und Fokus einer Backend App muss entschieden werden, in welchen Bereichen zusätzliche Elemente notwendig sind.

4.3.1 Menü und Titel

Im oberen Bereich einer Backend App, dem sogenannten Header[17] (siehe I, Abbildung 19), befinden sich einige Steuerelemente. Damit können Einstellungen, Hilfen oder spezielle Aktionen aufgerufen werden, welche die geöffnete App betreffen. Spezielle Aktionen können beispielsweise manuelles Aktualisieren oder Öffnen der Ansicht im Frontend sein. Die Entwicklung in der vorliegenden Arbeit wird aufgrund der Themenvielschichtigkeit in der Komplexität minimal gehalten und bietet für das Menü der TYPO3 Backend App daher lediglich die folgenden Möglichkeiten:

a) Öffnen der App-Einstellungen (siehe II, Abbildung 19)
b) Öffnen der App-Hilfe (siehe III, Abbildung 19)

Nachfolgend werden die einzelnen Elemente im Wireframe dargestellt.

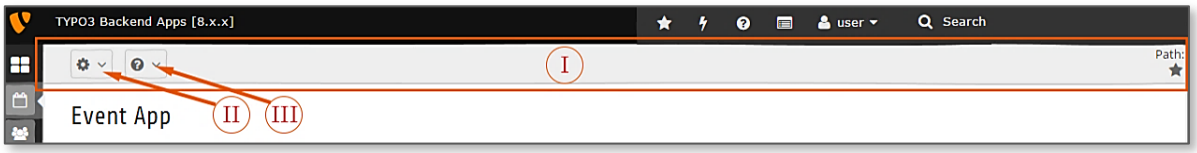


Abbildung 19: Wireframe des Header-Bereichs einer Backend App; Autor: A. Steiger, 2015

Die grundlegenden Einstellungen der Komponenten innerhalb der Backend App sind unter den App-Einstellungen (siehe II, Abbildung 19) zu finden. Damit sind die Optionen einer Backend App zentral an einem Ort im App-Kontext positioniert. Durch das Auslagern diverser Einstellungen der Hauptoberfläche in den Einstellungsbereich kann eine Übersichtlichkeit gewährleistet werden. Die Inhalte der App-Einstellungen sind beispielhaft im folgenden Wireframe gezeigt.

[17] Englisch für: Kopfzeile

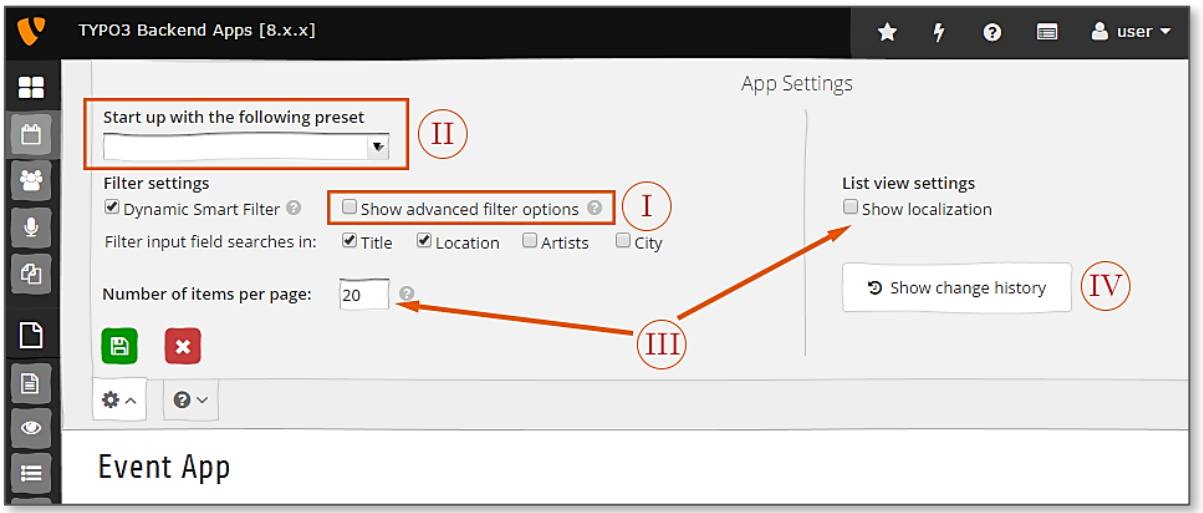


Abbildung 20: Wireframe der App-Einstellungen; Autor: A. Steiger, 2015

Der Rahmen der App-Einstellungen erweitert den oberen Header-Bereich innerhalb der Backend App und ermöglicht es, Einstellungen zu setzen und auf der aktuellen Seite zu bleiben. Hierbei können Daten mittels der AJAX-Technologie ohne ein erneutes Laden der Seite ausgetauscht werden. Mit diesen Steuerungen können beispielsweise erweiterte Ansichten aktiviert (siehe I), Optionen für den Start der App festgelegt (siehe II) sowie Werte für die Visualisierung definiert werden (siehe III). Zusätzlich besteht an dieser Stelle die Möglichkeit, den Verlauf aller inhaltlichen Änderungen innerhalb einer fokussierten Anwendung anzeigen zu lassen (siehe IV). Durch Fluid Partials[18] wird an dieser Stelle eine individuelle Erweiterbarkeit gewährleistet. Fluid ist die von TYPO3 CMS verwendete Template-Engine. [Typ14] Eine nähere Beschreibung dieser Anpassungsmöglichkeit mit Fluid ist im Kapitel 4.5 zu finden.

Die Änderungen an den Backend-App-Einstellungen durch den Benutzer werden individuell persistiert. Hierzu dient das TYPO3-interne Array: $BE_USER->uc. Dieses beinhaltet Konfigurationswerte der Benutzereinstellung und Werte, um den letzten benutzerdefinierten Zustand des Backends wiederherzustellen. [Typ15c] Beispielsweise wird in diesem Array gespeichert, welche Seiten im Seitenbaum aufgeklappt wurden. Das „$BE_USER->uc“ eignet sich daher genauso für das Persistieren der App-Einstellungen und sollte bei der Implementierung verwendet werden.

Die Hilfe-Schaltfläche b) (siehe III, Abbildung 19) im Headerbereich wird in einem ähnlichen Rahmen analog zu den App-Einstellungen dargestellt. An dieser Stelle befinden sich Informationen zur geöff-

[18] Englisch für: Teile

neten Backend App. Diese können Beschreibungen der App-Aufgaben oder Tutorials beinhalten. Weiterhin kann bei der Hilfe eine interaktive Bedienungsanleitung (User Guide) angeboten werden.

Der Titel einer TYPO3 Backend App ist im oberen Bereich unter dem Header platziert. Dieser besitzt zusätzlich die Funktion einer Breadcrumb, um einen Orientierungspunkt bei tieferen Ebenen einer Backend App zu schaffen. Aufgrund der fokussierten Aufgabenbearbeitung ist es für die Erstellung einer Backend App wichtig, dass die Tiefe der UI-Ebenen flach gehalten wird. Die folgende Darstellung zeigt ein Beispiel der Breadcrumb-Funktion.

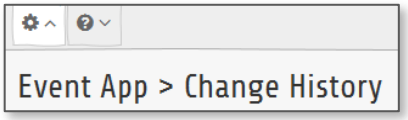


Abbildung 21: Wireframe des Backend-App-Titels mit Breadcrumb-Funktion; Autor: A. Steiger, 2015

4.3.2 Hierarchische Seitenstruktur

Die Komponente der hierarchischen Seitenstruktur, als eines der Hauptelemente von TYPO3 CMS, kann bei Bedarf ebenfalls Platz in einer Backend App finden. Der Seitenbaum ist eine Liste der bestehenden Seiten-Objekte und filtert je nach Auswahl die Ergebnisliste der Datensätze. Dabei werden alle Datentypen und Einträge angezeigt, die zur ausgewählten Seite gehören. Der Seitenbaum kann an einer beliebigen Stelle im seitlichen oder oberen Sektor positioniert werden. Die Darstellung und Logik der Komponente ist bereits vorkonfiguriert und wird von der App Engine zur Verfügung gestellt. Eine mögliche Darstellung der hierarchischen Seitenstruktur oberhalb der Ergebniskomponente könnte über ein spezielles Dropdown-Feld oder Overlay[19] gelöst werden, um nicht andere Elemente unnötig weit nach unten zu verschieben. Der Sachverhalt der Visualisierung einer hierarchischen Struktur ist umfangreich und wird an dieser Stelle der Vollständigkeit halber erwähnt. Eine weitere Abhandlung dieses Themengebietes in Bezug auf TYPO3 Backend Apps würde den Kontext der vorliegenden Arbeit verlassen.

[19] Englisch für: Überlagerung

4.3.3 Presets, Filter und Ergebniskomponente

Presets bezeichnen Vorlagen vordefinierter und persistierter Ansichtseinstellungen einer Backend App, die vom Benutzer manipuliert und abgespeichert wurden. Hierbei werden zwei Typen unterschieden. Zum einen gibt es Vorlagen, die für alle Nutzer zur Verfügung stehen und in der App-Konfiguration definiert werden. Zum anderen besitzt ein Benutzer die Möglichkeit, eigene Ergebnisansichten als Presets zu speichern. Letzteres kann ausschließlich von dem Backend-User verwendet werden, der es erstellt hat. Dies steht in Analogie zu den TYPO3-Lesezeichen (siehe Kapitel 3.1.3). Die eigenen Presets werden ebenfalls in der Datenbank persistiert. Auch wenn die Arbeitsschritte von TYPO3-Lesezeichen denen der Presets in den Backend Apps gleichen, wird aus Kontextgründen eine zusätzliche Datenbanktabelle benötigt. Da ein Preset auf den Informationen und Eigenschaften der Filter- sowie Ergebniskomponente basiert, werden die Details nach der Erarbeitung dieser Elemente behandelt (siehe Kapitel 6.6). Die Ergebnisansicht von Datensätzen und der Filter sind zwei komplexe Themengebiete, welche separat in den Kapiteln 5 und 6 bearbeitet werden.

4.4 Aufbau der Konfiguration

Um eine TYPO3 Backend App mit einem UI-Layout und dessen Komponenten sowie dem notwendigen Verhalten zu erstellen, muss diese konfiguriert werden. In Kapitel 3.4.2 wurde bereits erarbeitet, dass die Konfiguration in einer Datei definiert wird. Es wurden mögliche Konfigurationsformate auf Basis der Gedankensammlung von Daniel Siepmann gegenübergestellt mit dem Ergebnis, dass sich YAML als Format für die Konfiguration einer TYPO3 Backend App am besten eignet. Die Konfigurationsdatei einer TYPO3 Backend App befindet sich in einer Extension unter dem Pfad: „/Configuration/Apps“, und wird nach der Backend-App-Bezeichnung benannt. Die Datei gliedert sich grundsätzlich in zwei Teile, welche in der nachfolgenden Abbildung gekennzeichnet sind.

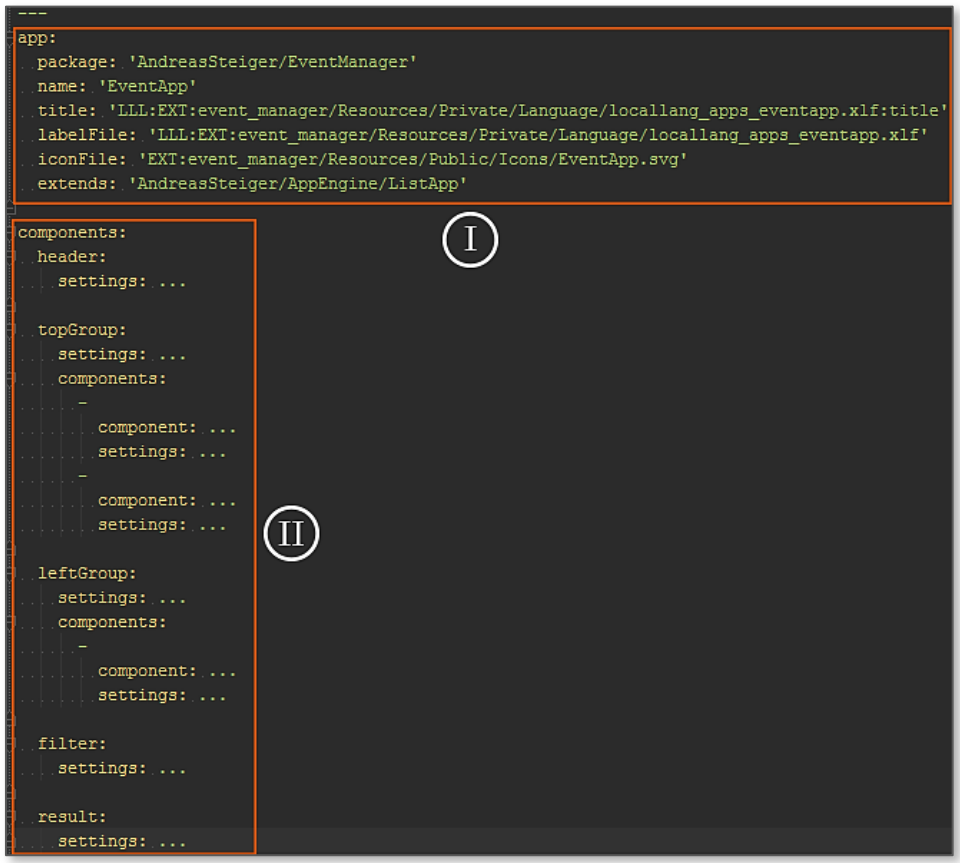


Abbildung 22: Grundgerüst der Konfigurationsdatei einer TYPO3 Backend App; Autor: A. Steiger, 2015

Der erste Bereich (siehe I) konfiguriert allgemeine Eigenschaften und benötigte Dateien einer Backend App. Hierbei wird ersichtlich, dass eine einfache Vererbung von Backend Apps möglich ist. Damit kann die Konfiguration einer bereits vorhanden TYPO3 Backend App kopiert und danach mit angepassten Werten überschrieben werden. Um eine App-Konfiguration einfach zu halten, bietet die App Engine verschiedene, vorkonfigurierte Backend Apps für diverse Aufgaben an. Diese können in einer neuen Backend App mittels Vererbung verwendet und mit nur wenig Aufwand ergänzt werden. Damit wird ein effizientes Erstellen einer TYPO3 Backend App gewährleistet.

Der zweite Teil der Konfiguration (siehe II) setzt Einstellungen für die Komponenten im UI-Layout einer Backend App. Darin sind die genannten Elemente aus Kapitel 4.3 wiederzufinden, welche an dieser Stelle in Layout-Sektor und Verhalten konfiguriert werden können. Die Komponenten in der „leftGroup“ und „topGroup“ stellen dabei die flexiblen Bereiche dar. Weiterhin gibt es Elemente, die lediglich benannt werden müssen und keine weitere Konfiguration bedürfen. Diese sind die Preset- und die Seitenbaumkomponente. Die App Engine erkennt die Elemente in der Konfiguration und lädt die entsprechenden Services, Controller sowie Ressourcen.

Um die Konfigurationen nicht ausschließlich über Editoren bearbeiten zu müssen, wird eine zusätzliche Extension im System benötigt: der App Builder[20]. Diese könnte zusätzlich eine Benutzeroberfläche zur Verfügung stellen, um neue TYPO3 Backend Apps produktiv mittels Drag-and-Drop oder eines Assistenten zu kreieren und zu bearbeiten.

4.5 App Engine

Backend Apps befinden sich in Form einer Konfiguration in verschiedenen TYPO3 Extensions und benötigen für eine Aktivierung sowie die Ausführung im CMS die App Engine. Diese verwaltet alle Backend Apps zentral, verringert den Aufwand einer App-Erstellung und verfügt über die notwendige Logik sowie Templates. Damit soll die Notwendigkeit von zusätzlicher Programmierung beim Erzeugen einer Backend App verringert werden. In der App Engine werden die folgenden Vorgänge für alle vorhandenen TYPO3 Backend Apps verarbeitet:

- Registrierungsprozess
- Konfigurationsverarbeitung und Vererbungsprozess
- Logikprozesse der Standard UI-Komponenten

Die Konfigurationsdatei einer Backend App wird in der entsprechenden Extension-Datei „ext_tables.php“ an die App-Engine weitergereicht. In der Datei „ext_tables.php“ werden ebenfalls Backend Module und Frontend Plugins im CMS registriert. Die Registrierung ist in dieser Extension-Datei notwendig, da diese als eine der letzten im Aufbauprozess des TYPO3 Backends verarbeitet wird. Damit ist sichergestellt, dass beim Auslesen der Konfigurationen alle benötigten Arrays und Objekte zur Verfügung stehen. Mit dem folgenden Funktionsaufruf kann eine Backend-App-Konfiguration an die App Engine weitergereicht werden:

```
ConfigurationService::create()->addConfiguration('EXT:event_manager/Configuration/Apps/EventApp.yaml');
```

Die übergebene Datei wird von der App Engine geparst. Danach können mittels PHP die ersten Informationen aus der Konfiguration entnommen werden, welche für die Registrierung im CMS notwendig sind. Hierbei findet der YAML-Parser vom PHP-Framework Symfony Anwendung, welcher bei

[20] Englisch für: Erbauer

TYPO3 CMS 7 im Core enthalten ist. Die App Engine führt mit den Daten der Backend App den Registrierungsprozess eines Backend Modules durch und erstellt damit einen Eintrag mit entsprechendem Handler im Modulbereich des Backends. In der nachfolgenden Systemübersicht sind die dafür notwendigen Komponenten und Klassen dargestellt.

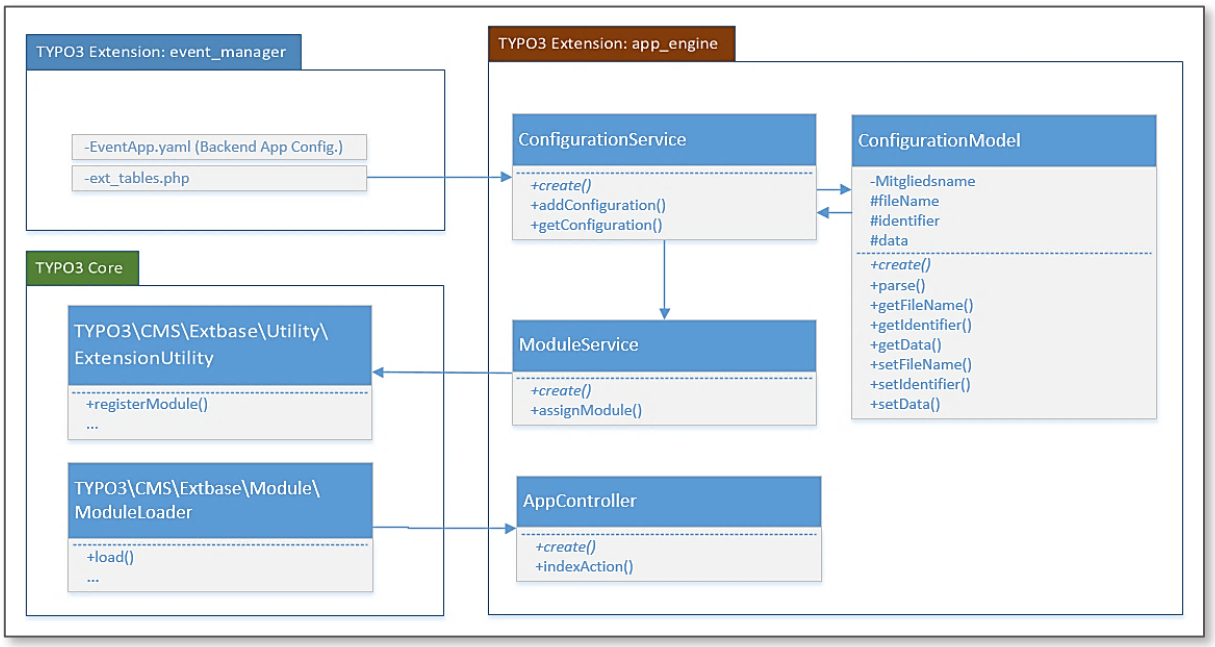


Abbildung 23: Klassenübersicht für den Registrierungsprozess einer TYPO3 Backend App; Autor: A. Steiger, 2015

Weiterhin werden in der App Engine die nachfolgenden Dateien als Hilfestellung für die Backend-App-Erstellung bereitgestellt:

- Fluid-Templates für die verschiedenen UI-Komponenten
- Unterschiedliche Basis-Backend-Apps mit Grundkonfigurationen für verschiedene Aufgaben
- Ressourcen: Stylesheets, JavaScript, Icons, Labels für Mehrsprachigkeit usw.

Eine Erweiterung der TYPO3 Backend App ist über die Fluid-Templates möglich. Hierzu werden leere Partial-Dateien in den Templates der App Engine vorgesehen, welche als Platzhalter für individuelle Ergänzungen dienen. Somit können in einer beliebigen Extension weitere Elemente an festgelegten Stellen im Layout hinzugefügt werden, ohne dazu die Fluid-Templates kopieren zu müssen. Ein weiterer Vorteil existiert durch die Fluid-Variable „settings“, die das gesamte Konfigurations-Array zur Verfügung stellt. Über „settings.configuration.“ kann auf die Werte aus der Konfigurationsdatei zugegriffen werden. Diese könnten beispielsweise in den App-Einstellungen Platz finden, um dem Benutzer

die Möglichkeit zum Ändern weiterer Werte zu geben. Die folgende Abbildung zeigt die Auftrennung der Fluid-Dateien und die vorgesehenen Partials, die bei Bedarf ergänzt werden können.

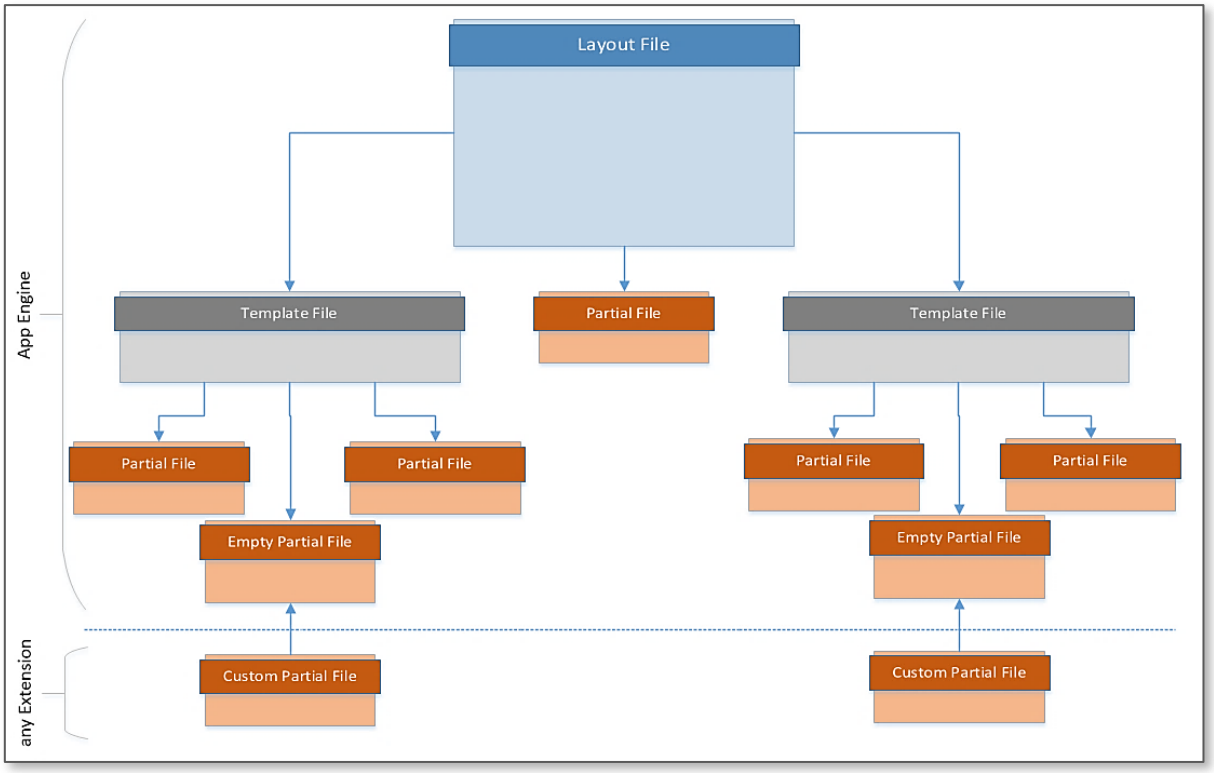


Abbildung 24: Beispielhafte Übersicht der Fluid-Dateien einer TYPO3 Backend App und der Erweiterungsmöglichkeit mittels Partials; Autor: A. Steiger, 2015

Wie in Kapitel 4.3.3 erarbeitet, benötigt die App Engine zusätzlich eine Datenbanktabelle für benutzerdefinierte Ansichten. Diese werden in der neuen Tabelle „tx_appengine_domain_model_userpresets“ persistiert. Der Aufbau wird in der folgenden Grafik visualisiert.

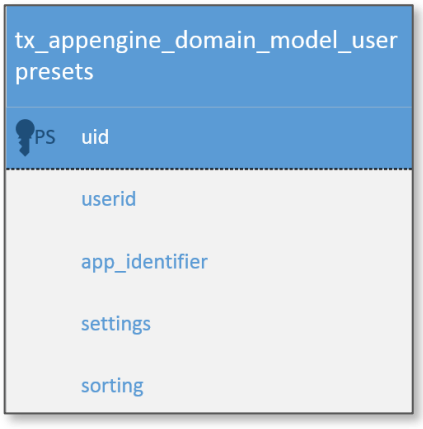


Abbildung 25: Schema der Datenbanktabelle für benutzerdefinierte Presets der Ergebnisansicht; Autor: A. Steiger, 2015

In dem Attribut „settings“ werden alle Einstellungen in einem JSON-String gebündelt. Welche Daten dort hinterlegt sind, wird in Kapitel 6.6 genauer beschrieben.

Unter oben genannten Voraussetzungen sind für die Erstellung einer neuen TYPO3 Backend App mindestens die folgenden Schritte notwendig:

- YAML-Konfigurationsdatei in einer Extension unter dem Pfad „Configuration/Apps“ anlegen
- Konfigurationen einer vorhandenen Backend App kopieren und anpassen
- Notwendige Icons und Dateien für die Mehrsprachigkeit in entsprechenden Ordnern erstellen
- Die App-Konfiguration in der Extension-Datei „ext_tables.php“ an die App Engine übergeben

Für die übersichtliche Verwaltung aller vorhandenen Backend Apps im System ist eine weitere Komponente im Backend erforderlich. Dabei handelt es sich um den App Manager. Die Bezeichnung orientiert sich am TYPO3 Extension Manager, in dem alle Erweiterungen des Systems verwaltet werden. Der App Manager sollte ein Teil der App Engine sein und als ein zusätzliches Backend Modul zur Verfügung stehen. Hier werden alle vorhandenen Backend Apps gelistet und Informationen sowie Möglichkeiten zur Bearbeitung angezeigt.

5 VISUALISIERUNG VON DATENSÄTZEN

5.1 Datenvisualisierung im TYPO3 CMS

Nachdem die Konzepte der Systemintegration und des Aufbaus einer TYPO3 Backend App definiert wurden, erfolgt nun die Entwicklung der Ergebniskomponente. Diese baut auf den bisherigen Kapiteln sowie Analysen auf und beschäftigt sich mit einer geeigneten Visualisierung von Datensätzen.

Die zentralen Kernaspekte eines CMS sind dessen Inhalte und Informationen. [Bre13, S.12] Diese werden von Benutzern mit entsprechenden Rechten manipuliert sowie komponiert und liegen bei TYPO3 CMS als Einträge in einer relationalen Datenbanktabelle vor. Eine Gruppierung von zusammenhängenden Inhalten oder Daten werden als Datensatz bezeichnet. [Wik15b] Aufbauend auf der hergeleiteten Definition eines Datensatzes in TYPO3 CMS (siehe Kapitel 3.1.1), werden dessen Aufgabenbereiche untersucht. Datensätze dienen der Information, der Darstellung von Beziehungen zwischen einzelnen Inhalten sowie der Informationsspeicherung. Um eine solche Datenbündelung anzuzeigen, bedarf es einer entsprechenden kontextbezogenen Visualisierung. Diese kann je nach Darstellung den Informationsgehalt eines Datensatzes erhöhen oder verringern. Beispielsweise kann eine Datenauswertung mittels Diagrammen und Trendlinien neue Zusammenhänge zwischen den gegebenen Datensätzen aufzeigen und damit zusätzlichen Mehrwert liefern. Eine Kachelansicht hingegen führt durch deren Visualisierungsart zu einer Komprimierung und Fokussierung des Informationsgehaltes.

Die Aufgaben der Datenvisualisierung sind damit auf der einen Seite analog zu denen eines Datensatzes. Auf der anderen Seite verfolgt die Visualisierung zusätzlich das Ziel, dem Anwendungsfall entsprechend eine bestimmte Anzahl an Informationen anzuzeigen. Hierbei liegen die Schwerpunkte bei der Darstellung der Zusammenhänge bis hin zu einer Übersicht der ausschließlich relevanten Informationen. Für eine möglichst geeignete, detaillierte Visualisierung von Daten aus einer Datenbanktabelle eignet sich die Tabellenansicht am besten, da diese eine identische Abbildung ohne Informationsverlust erzeugen kann. In einer Tabelle wird eine Vielzahl von Daten übersichtlich und sortiert dargestellt, wobei dies abhängig von der unterschiedlichen Informationsart sowie -größe der Inhalte innerhalb eines Datensatzes ist. Eine Herausforderung stellt beispielsweise die geeignete Darstellung eines einzeiligen Textfeldes neben einem mehrzeiligen Text dar, ohne den Informationsgehalt oder die Übersichtlichkeit zu verlieren.

Es zeigt sich damit, dass die Visualisierung von Datensätzen abhängig von der zu erfüllenden Aufgabe ist. Sollen Emotionen durch große Bilder geweckt werden, ist eine Kachelansicht geeignet. Bei Verwaltungstätigkeiten unterstützt eine sortierte Tabelle mit detaillierten Datensatzinformationen optimal. Werden Auswertungen benötigt, können diese mittels Diagrammen und berechneten Übersichten visualisiert werden.

5.2 Tabellen im Backend

Für die Anzeige bestimmter Datenbanktabellen im Backend von TYPO3 CMS ist das sogenannte Table Configuration Array (TCA) vorgesehen. Dieses ist ein globales Array, welches sich unabhängig von der Abfragesprache SQL um die Sichtbarkeit, Editierbarkeit und Validierung von Datenbanktabellen im Backend sowie um die Beziehungen zwischen mehreren Tabellen kümmert. [Typ15b] Damit definiert sich ein Datensatz in TYPO3 ebenso durch einen Eintrag im TCA. Ferner dient eine Liste oder Tabelle im TYPO3 Backend als Abbildung der persistierten, relationalen Datenbanktabelle mit Hilfe des TCA.

5.3 Einfluss der Visualisierungsaufgaben auf die benötigten Komponenten im UI-Layout

Um eine Darstellung von Informationen in Abhängigkeit der Visualisierungsaufgabe zu verdeutlichen, wird der Sachverhalt anhand zweier Beispiele erläutert. Diese zeigen ebenfalls die Notwendigkeit von einzelnen UI-Komponenten innerhalb einer TYPO3 Backend App auf. Wichtig ist hierbei die Verwendung von seitlichen Elementen zu untersuchen, da diese den Visualisierungsbereich der Ergebniskomponenten in der Breite einschränken und damit einer Tabelle weniger Raum für Informationen geben. Auch wenn die Möglichkeit für horizontales Scrollen bei mehreren nebeneinanderliegenden Daten eine Alternative bietet, schränkt dies die Nutzungsqualität ein (vgl. Kapitel 3.3.2) und ist nicht in jedem Fall wünschenswert.

Exemplarisch seien zwei allgemeine Visualisierungsaufgaben gegeben: Massendatenverarbeitung und Datenverwaltung. Ersteres muss dem Benutzer die Möglichkeit geben, entsprechende Datensätze zu finden und diese zu selektieren. Danach können Aktionen auf die ausgewählten Einträge angewendet werden. Da mit verschiedenen Datensatztypen von bestimmten Seiten gearbeitet wird, ist das Seitenbaumelement (siehe I, Abbildung 26) erforderlich. Wenn dieses seitlich eingebunden wird, steht der Information der Auswahltabelle (siehe II, Abbildung 26) weniger Raum zur Verfügung. Je-

doch dient in diesem Fall die Tabelle nicht als detaillierter Informationsgeber, sondern als Liste der vorhandenen Einträge. Die Kernaufgabe dieser Visualisierung ist daher die Selektion. Die Tabelle muss lediglich jene Informationen bieten, welche für eine Datensatzidentifikation notwendig sind. Damit können die benötigten Daten für die Aufgabenerfüllung in wenigen Spalten Platz finden, ohne auf ein horizontales Scrollen in der Backend App angewiesen zu sein. Die folgende Darstellung zeigt beispielhaft eine mögliche Komposition der beiden Komponenten bei einer Massendatenverarbeitung, wie es in ähnlicher Form bereits bei TYPO3 CMS vorzufinden ist.

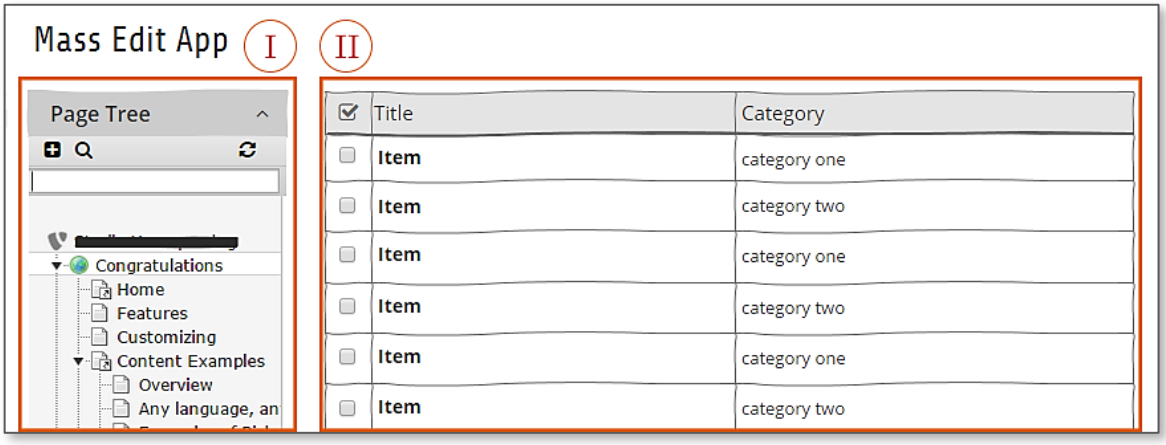


Abbildung 26: Beispielhaftes Wireframe für Massendatenverarbeitung und aufgabenorientierte Visualisierung; Autor: A. Steiger, 2015

Eine Visualisierung zur Datenverwaltung hingegen verlangt einen hohen Informationsgehalt. Dies erfordert eine ausführliche Tabelle mit detaillierten Informationen zu den Datensätzen. An dieser Stelle würde ein zusätzlich seitliches Element (z.B. der Seitenbaum) den Aufgabenfokus stören. Unterstützend wäre eine Tabellendarstellung, welche sich auf die gesamte verfügbare Seitenbreite einer TYPO3 Backend App erstreckt. Jedoch ist bei TYPO3 CMS eine Seitenauswahl in den meisten Fällen erforderlich (siehe Kapitel 3.1.1). Für eine qualifizierte Bewertung müssen die einzelnen Arbeitsschritte dieser Visualisierungsaufgabe näher analysiert werden. Bei einer Verwaltungs-App ist es wichtig, Datensätze gleichen Typs zu verwalten, um die Übersicht sowie den Fokus zu gewährleisten. Beispielsweise bietet die Veranstaltungs-App eine Tabelle aller Veranstaltungen mit sämtlichen dazugehörigen Informationen. Das bedeutet, dass ausschließlich die Datensätze vom Typ „Veranstaltung“ angezeigt werden. Diese können jedoch auf verschiedenen Seiten im CMS liegen. Ein Arbeitsschritt eines Benutzers wäre damit das Anzeigen aller Veranstaltungen im System mit einer Gruppierung nach den relevanten Seiten. Die Relevanz ergibt sich hierbei durch das Vorhandensein eines Datensatzes vom Typ „Veranstaltung“ auf einer Seite.

Damit werden die Unterschiede im Vorgehen beider Beispiele deutlich. Im ersten Fall ist ein Zwischenschritt der Backend App: zeige dem Benutzer alle Datensätze auf einer Seite. Bei der Datenverwaltung hingegen liegt der Fokus auf der Aufgabe: zeige dem Benutzer jene Seiten, welche Datensätze eines bestimmten Typs beinhalten. Dadurch wird lediglich eine komprimierte Darstellung der hierarchischen Seitenstruktur (siehe I, Abbildung 26) benötigt. Das nachfolgende Wireframe zeigt beispielhaft eine mögliche Visualisierung der kompakten Seitenbaumkomponente sowie einer detaillierten Tabelle (siehe II).

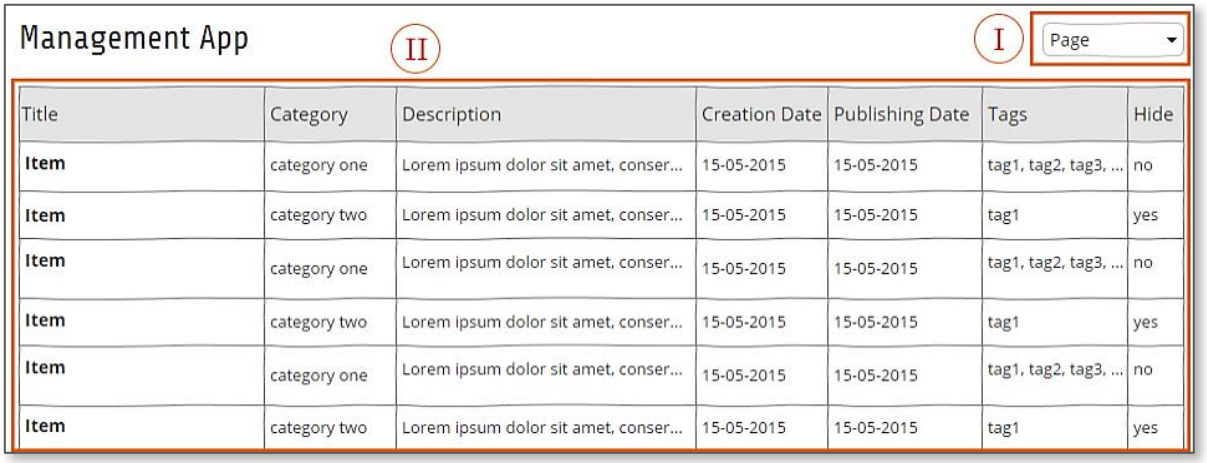


Abbildung 27: Wireframe als Beispiel für detaillierte Datenverwaltung und aufgabenorientierte Visualisierung; Autor: A. Steiger, 2015

Um eine komprimierte Darstellung einer hierarchischen Seitenstruktur in einem Dropdown-Feld zu visualisieren (siehe I), bedarf es einiger Optimierungen. Die Inhalte der minimierten Seitenauswahl sollten in einer schmalen Liste dargestellt werden und den notwendigen Informationsgehalt über Zuordnung und Hierarche liefern. Eine Möglichkeit ist, ausschließlich die relevanten Seiten mit Seitennamen, geordnet nach der jeweiligen Webseite (Root-Seite) darzustellen. Die Trennung nach der Webseite ist in TYPO3 CMS notwendig, da das System die Verwaltung mehrerer Webseiten in einer Installation erlaubt. Mit der Gruppierung kann eine konkrete Zuweisung bei gleichen Seitennamen im CMS gewährleistet werden. Weiterhin besteht die Möglichkeit, mehrere identische Seitenbezeichnungen innerhalb einer Webseite vorzufinden. In diesem Fall werden die jeweiligen Elternelemente mit angezeigt, um eine Verwechslungsgefahr zu vermeiden. Sollte dies ebenfalls eine Gleichheit in der Benennung aufzeigen, werden iterativ weitere Elternelemente ergänzt, bis eine Ungleichheit für eine Unterscheidung der Seiten vorliegt. Zusätzlich steht eine Schaltfläche zur Anzeige des kompletten hierarchischen Pfades mittels eines Informations-Icons zur Verfügung, um bei Bedarf detaillierte Informationen zu liefern. Die nachfolgende Abbildung zeigt diese mögliche Umsetzung der komprimierten Seitenbaumdarstellung.

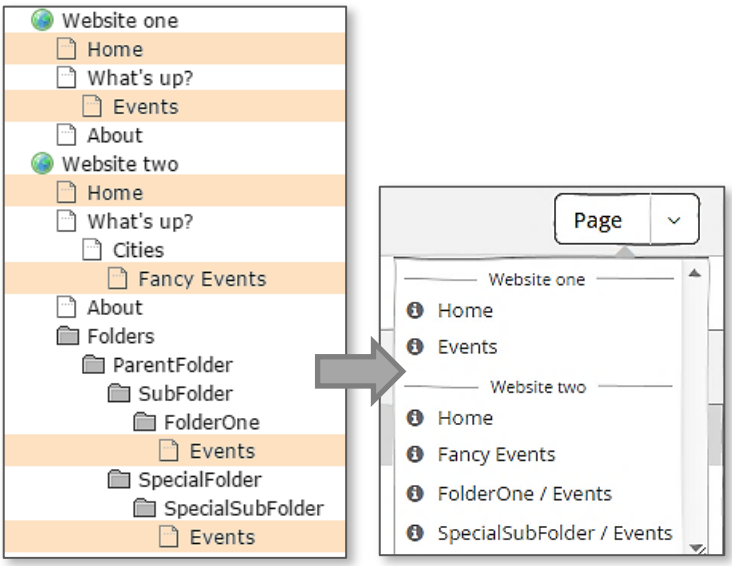


Abbildung 28: Komprimierte Seitenbaumdarstellung für relevante Seiten als Dropdown; Autor: A. Steiger, 2015

Diese mögliche komprimierte Darstellung steht dynamisch in dem Filterbereich zur Verfügung, sobald keine Seitenbaumkomponente in einer TYPO3 Backend App angezeigt wird. Des Weiteren ist das Dropdown-Feld erst aktiv, wenn sich die Datensätze auf mehr als einer Seite befinden.

5.4 Aufbau der Ergebniskomponente als Datenvisualisierung

Die Ergebniskomponente der TYPO3 Backend Apps basiert im Anzeige- bzw. Ausgabemodus (siehe Kapitel 4.3) auf einer Ergebnismenge von Datensätzen. Diese können frei visualisiert und dargestellt werden. Damit sind nicht nur Listen oder Layout-Abbildungen in der Ergebniskomponente möglich, sondern auch beispielsweise Kacheln, Kalenderansichten, Diagramme oder Trendlinien sowie moderne Cover-Flow-Illustrationen. Je nach Visualisierungsaufgabe sind bestimmte Ansichten notwendig und hilfreich für die Arbeit mit dem CMS. Für die weitere Ausarbeitung werden Details am Beispiel der Tabellenansicht bzw. Listenansicht und an den Anwendungsfällen (siehe Kapitel 3.6) erläutert. In diesem Zusammenhang wird die Ergebniskomponente im weiteren Verlauf der vorliegenden Arbeit als Anzeige- bzw. Ausgabeeinheit betrachtet. Diese Komponente wird aus den Erkenntnissen der Usability-Grundlagen (siehe Kapitel 2.3) sowie den Auswertungen der Konkurrenzanalyse (siehe Kapitel 3.3.2) entwickelt. Im Allgemeinen können zwei Bereiche unterschieden werden, die im Folgenden aufgeführt sind:

- Kopfbereich mit Steuer- und Navigationsschaltflächen
- Inhaltsbereich mit Datenvisualisierung

5.4.1 Kopfbereich

Im Kopfbereich der Ergebniskomponente können die verschiedenen Ansichten umgeschaltet, durch die Paginierung navigiert sowie neue Datensätze hinzugefügt werden. Weiterhin gibt es entsprechende Aktions- oder Manipulationselemente, welche je nach Ansicht des Inhaltsbereiches unterschiedlich zur Verfügung stehen. Diese werden dynamisch, je nach Ansicht ein- oder ausgeblendet. Unter anderem ist bei der Datenvisualisierung in Form einer Tabellenansicht zusätzlich eine Spaltenauswahl verfügbar. Diese bietet mittels eines Dropdown-Felds die Möglichkeit zur Ab- oder Anwahl von freigegebenen Spalten. Die Freigabe der Spalten wird durch drei Faktoren definiert:

- Persistiert in Datenbank
- Definiert im TCA
- Freigegeben in der Konfiguration der entsprechenden TYPO3 Backend App

Zusätzlich stehen in diesem Auswahlfenster errechnete Spalten (Computed Columns) zur Verfügung. Diese werden ausschließlich in der Backend-App-Konfiguration definiert und sind nicht persistiert. Dieser Sachverhalt wird in Kapitel 5.5 näher betrachtet. Der UI-Entwurf für die Spaltenauswahl ist in der folgenden Abbildung zu sehen.

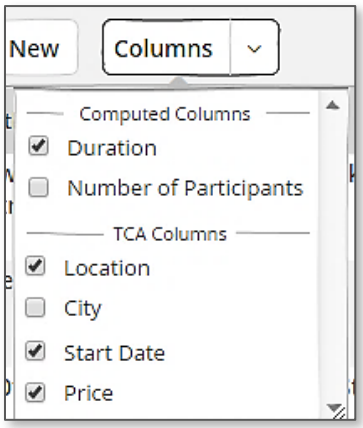


Abbildung 29: Dropdown zur Spaltenauswahl; Autor: A. Steiger, 2015

Die erste Spalte (Veranstaltungstitel) kann nicht aus- oder eingeblendet werden, da diese Sonderfunktionen enthält und somit in jedem Fall sichtbar bleiben sollte. Überdies ist bei der Selektion der Spalten zu beachten, dass es ab einer bestimmten Anzahl zum Informationsüberlauf in der Bildschirmbreite kommt. Damit wird ein horizontales Scrollen erforderlich. Um die Nutzungsqualität in diesem Fall zu verbessern, werden visuelle Optimierungen in Kapitel 5.6 vorgestellt.

Nachfolgend ist das Wireframe für den Kopfbereich am Beispiel einer Tabellen- bzw. Listenansicht abgebildet.

Abbildung 30: Wireframe des Kopfbereiches der Ergebnisansicht am Beispiel einer Tabellenansicht; Autor: A. Steiger, 2015

5.4.2 Inhaltsbereich

Die Herausforderung bei der Visualisierung von Datensätzen besteht darin, die unterschiedlichen Inhalte übersichtlich darzustellen. Um einen groben Überblick der Inhaltsvielfältigkeit bei TYPO3 CMS zu geben, werden beispielhaft die folgenden Datentypen aufgelistet:

- Einzeiliger Text
- Mehrzeiliger Text
- Auswahlliste (Dropdown)
- Auswahlkasten (Checkbox)
- Medien oder Dateien
- Benutzer und Benutzergruppe
- Datumsangabe
- Verlinkung
- Weitere Dimensionen des Datensatzes (z.B. Lokalisierung)
- Mehrzeilige TypoScript-Konfigurationen
- Weitere Inhaltselemente und Frontend Plugins

Es wird deutlich, dass eine übersichtliche und benutzerfreundliche Visualisierung dieser Vielschichtigkeit in einer gemeinsamen Tabelle intensive Analysen sowie Anpassungen benötigt. Wichtig für die Übersicht ist, eine nahezu einheitliche Höhe aller Zeilen zu erhalten. Die Schwierigkeit besteht darin, so wenig wie möglich Inhalte eines Datensatzes zu beschneiden und damit den Informationsgehalt zu bewahren. Für bestimmte Inhaltsfelder müssen jedoch Begrenzungen oder komprimierte sowie alternative Darstellungsmöglichkeiten mit ähnlichem Informationsgehalt verwendet werden. Vorteilhaft für eine kompaktere Visualisierung sind Abkürzungen, Icons oder kleine Vorschaubilder. Damit kann beispielsweise eine Kategorie eines Datensatzes über ein aussagekräftiges Icon beschrieben und Backend-Benutzer mittels Avataren (Profilbilder) oder Namensinitialen dargestellt werden. Fer-

ner kann die Datensatzkategorie ebenfalls über Reiter bzw. Tabs die jeweiligen Tabelleninhalte anzeigen. Somit können unter anderem Dateien gruppiert nach Medien und Dokumenten in zwei separat gefilterten Tabellen visualisiert werden. Durch diese Beispiele der Informationsverlagerung und -komprimierung wird Raum für weitere Inhalte geschaffen, ohne die Inhaltsaussage an diesen Stellen enorm reduzieren zu müssen.

Neben den Tabelleninhalten muss die Ebenentiefe einer TYPO3 Backend App geklärt werden. Interaktionen, Management- und Persistenz-Funktionen sollten in den entsprechenden Ebenen positioniert werden. Eine Datenliste bzw. -Tabelle kann aus einer streng getrennten Management- und Persistenz-Ebene (siehe Kapitel 3.1.2) oder einer übergreifenden Schicht bestehen. Wird eine Tabelle mit In-Page-Editing-Funktionen und der damit verbundenen direkten Bearbeitung auf der aktuellen Seite ergänzt, verschmelzen Persistenz und Management auf einer gemeinsamen Ebene. Es kann auch mehrere dieser Ebenen geben. Beispielgebend stellt eine Wetter-App für Smartphones die Daten zunächst in einer Liste der Tage mit einem errechneten Mittelwert der Temperaturen dar. Bei einem Klick auf den Tag navigiert die App zu einer tieferen Ebene, welche stundenweise die detaillierten Wetterdaten auflistet. Die TYPO3 Backend Apps spezialisieren sich in der vorliegenden Arbeit auf die Visualisierung, und nicht die Art der Bearbeitung. Somit werden Persistenz und Management in getrennten Ebenen behandelt. Funktionen wie das Sortieren und Filtern sind hierbei abzugrenzen. Diese werden nicht in einem Datensatz persistiert und gehören damit zur Management-Ebene. Weiterhin sind die Schaltflächen in der Ergebniskomponente zu untersuchen. TYPO3 CMS bietet bisher eine Vielzahl an Aktionsschaltflächen für einen Datensatz an. Eine Auswahl dieser ist nachfolgend gezeigt.

Abbildung 31: Aktionsschaltflächen eines Datensatzes im Listenmodul von TYPO3 CMS 7; Autor: A. Steiger, 2015

Diese Darstellung vermittelt viele Aktionsmöglichkeiten und kann auf diese Weise optisch exakt einem Datensatz zugeordnet werden. Jedoch befinden sich die Schaltflächen in zusätzlichen Spalten und engen mit der hohen Anzahl an Optionen den Platz für die Datensatzinformationen ein. Außerdem ist diese Vielzahl an Aktionsmöglichkeiten hinsichtlich der Notwendigkeit und des Fokus zu hinterfragen. Um an dieser Stelle mehr Platz in der Breite zu schaffen, könnten die Editier- bzw. Aktionsschaltflächen, analog zu Wordpress (siehe Kapitel 3.3.2), unterhalb in der Datensatzzeile angezeigt werden. Ein Beispiel zeigt das folgende Wireframe, welches anhand der Veranstaltungs-App die Visualisierung deutlich macht.

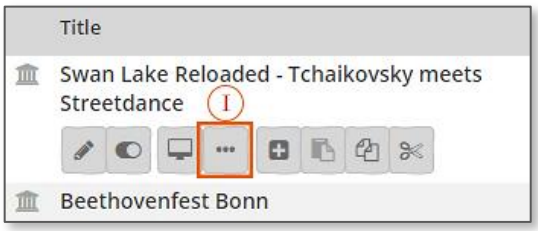


Abbildung 32: Beispielhaftes Wireframe einer möglichen Position der Aktionsschaltflächen in einer Tabelle bzw. Liste; Autor: A. Steiger, 2015

Hierbei müssen die Wichtigkeit und Aktionen hinter den Schalflächen analysiert werden. Beispielsweise sind Buttons, welche am häufigsten Verwendung finden, sofort bei einem Mouseover der Zeile sichtbar. Weitere Aktionen können über die Schaltfläche mit dem Drei-Punkte-Symbol (siehe I) in einem zusätzlichen Overlay gewählt werden. Ebenso sollte die Löschen-Funktion erst in einer zweiten Ebene erreicht werden, um somit ein versehentliches Entfernen zu vermeiden. Ein Nachteil dieser Visualisierung ist die Höhe der einzelnen Zeile, welche genügend Platz für mehrzeilige Informationen sowie die Schaltflächen bieten muss. Dies führt bei einer Vielzahl an Einträgen zu einer langen Listenansicht. Jedoch wird durch den Freiraum in den Zeilen auch der Wert der Übersichtlichkeit erhöht.

Bei allen genannten Möglichkeiten zur Visualisierung richtet sich der Einsatz bestimmter Elemente nach der Visualisierungsaufgabe und Notwendigkeit. Die TYPO3 Backend App unterstützt einen flexiblen Einsatz durch die Anpassung der Fluid-Templates (siehe 4.5). Somit kann ein TYPO3-Integrator über die vorgesehenen Fluid-Partial-Dateien eigenständig Buttons unterhalb des Titels eines Datensatzes oder in einer zusätzlichen Spalte hinzufügen. Für die fokussierte Aufgabe der Veranstaltungs-Backend-App werden nur wenige Aktionsschaltflächen benötigt. Hierbei wird sich auf das Editieren und Ausblenden beschränkt. Aufgrund einer platzsparenden Visualisierung wird die Schaltfläche zum Editieren mit einer hinterlegten Klickfunktion des Titels ersetzt (siehe I, Abbildung 33). Die Anzeige der Sichtbarkeit im Frontend bzw. des Veröffentlichungsstatus (siehe II, Abbildung 33) sowie die Schaltfläche dafür (siehe III, Abbildung 33) werden in der nachfolgenden Abbildung gezeigt.

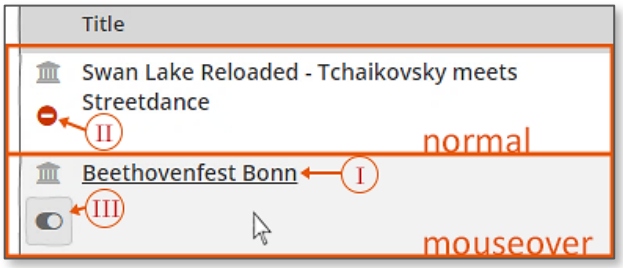


Abbildung 33: Wireframe der reduzierten Aktionsschaltflächen am Beispiel des Veranstaltungsmanagements; Autor: A. Steiger, 2015

Die Option zum Löschen sowie alle weiteren Informationen sind in der Persistenz-Ebene zu finden, die über einen Klick auf den Titel des jeweiligen Datensatzes erreicht wird.

Unabhängig vom Einsatz der speziellen Aktionsschaltflächen sind grundsätzlich in einer Tabellenansicht einer TYPO3 Backend App folgende Interaktionen im Inhaltsbereich vorgesehen:

I. Bearbeiten von einzelnen Datensätzen in einer zusätzlichen Persistenz-Ebene
II. Verschieben und Skalieren von Spalten
III. Sortieren nach Spalten
IV. Filtern nach vordefinierten Werten (z.B. Kategorie)

Um die nachfolgenden Erläuterungen besser zu veranschaulichen, zeigt Abbildung 34 den entwickelten Aufbau am Beispiel der Veranstaltungs-Backend-App mit zusätzlicher Beschriftung analog zu den eben genannten römischen Zahlen.

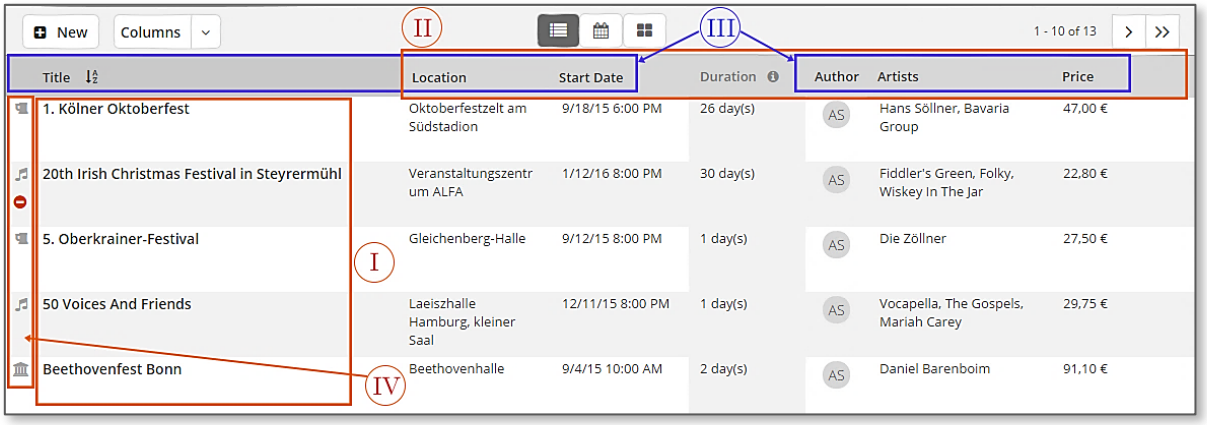


Abbildung 34: Wireframe des Inhaltsbereiches der Ergebniskomponente am Beispiel des Veranstaltungsmanagements; Autor: A. Steiger 2015

Das Verschieben und Skalieren von Spalten (siehe II) ist mittels Drag-and-Drop-Unterstützung vorgesehen. Hierbei fällt auf, dass die erste Spalte nicht mit dieser Funktion versehen ist. Die erste Spalte in der Tabelle ist ein Sonderfall, da sich innerhalb dieser zusätzliche Aktionsschaltflächen befinden können. Weiterhin identifiziert der Titel oder Name in den meisten Fällen den Datensatz. Bei einer hohen Anzahl an Spalten wird die Erste beim horizontalen Scrollen fixiert, um somit die Zugehörigkeitserkennung der Inhalte zu den einzelnen Einträgen sicherzustellen.

Wenn die Sortierung angewendet wird (siehe III), bestimmt dies die Reihenfolge der Zeilen. Standardmäßig wird bei TYPO3 CMS nach dem Erstellungsdatum sortiert. Dieser Fall liegt daher im Normalzustand vor, wenn keine bestimmte Sortierung aktiviert wurde. Eine Anordnung nach einer Spalte kann durch einen Klick darauf aktiviert werden. Hierbei wird zunächst aufsteigend und beim erneu-

ten Betätigen absteigend sortiert. Klickt der Benutzer ein drittes Mal auf den Spaltenkopf wird die Ordnung wieder deaktiviert. Die Sortierung wird durch ein entsprechendes Icon signalisiert (siehe Spaltenkopf „Title“). Auffällig ist, dass bei der Anordnungsfunktion die Spalte „Duration“ ausgeschlossen wird. “Duration“ ist eine errechnete Spalte (siehe Kapitel 5.5), deren Werte nicht persistiert sind. Daher ist ein Sortierprozess dessen mit hohem Bearbeitungsaufwand verbunden. Hierbei müssten zunächst alle Datensätze abgerufen, die Spalteninhalte errechnet und danach sortiert werden. Bei über 10.000 Einträgen würde dies zu nicht vernachlässigbaren Performanceproblemen führen und wird deshalb in der Funktion eingeschränkt. Das Filtern nach bestimmten Werten wird in diesem Beispiel über Icons ermöglicht, die der jeweiligen Datensatzkategorie entsprechen (siehe IV). Diese Funktion steht jedoch nur zur Verfügung, wenn die Filterkomponente in der Backend App aktiviert wurde.

5.5 Computed Values

Hinsichtlich der TYPO3 Backend Apps sind Computed Values[21] nicht persistierte Werte, die aus Datensätzen berechnet werden. Der Inhaltsbereich kann mit diesen Werten unabhängig vom TCA ergänzt werden. Computed Values können in Form von Spalten, Zeilen oder formlosen Elementen als zusätzliche Information zu den vorhandenen Daten hinzugefügt werden. Grundsätzlich dienen diese Werte ausschließlich der Auswertung und Information im Backend. Je nach Inhalt der existierenden Datensätze sind verschiedene Berechnungen möglich:

- Summe
- Differenz
- Prozentsatz
- Anzahl
- Mittelwert

Diese Operationen entsprechen den Standard-Berechnungen der Tabellenkalkulationen Excel und Google Docs Tabellen (siehe Kapitel 3.3.2).

Die Berechnung über eine Zeile bezieht sich auf die Inhalte innerhalb eines Datensatzes und erzeugt eine neue Spalte mit den jeweiligen Ergebniswerten. Die benannten Operationen werden von der

[21] Englisch für: berechnete Werte

App Engine verarbeitet. Die nachfolgende Tabelle zeigt Beispiele für Computed Values eines Datensatzes.

Bezeichnung der berechneten Spalte (Computed Column)	Operation	Berechnung	Beispielergebnis
Countdown	Differenz	$Startzeit - Jetzt$	12 : 30 : 25
Dauer	Differenz	$Endzeit - Startzeit$	2 Tage
Anzahl der Teilnehmer in Prozent	Prozentsatz	$\frac{Anzahl\ der\ Teilnehmer}{Maximalwert}$	20 %

Tabelle 3: Beispieloperationen für datensatzbasierte Computed Values; Autor: A. Steiger, 2015

Hierbei fällt auf, dass es sich bei den Operanden um Spaltenwerte, benutzerdefinierte Werte (z.B. Maximalwert) oder dynamische Größen handelt. In der Konfiguration der Backend App könnten diese Beispiele wie in der folgenden Abbildung definiert werden.

```
computedColumns:
  countdown:
    label: 'EXT:event_manager/Resources/Language/locallang_apps_eventapp.xlf:result.list...
    type: 'datetime'
    operation: 'difference'
    arguments:
      -
        column: 'startdate'
      -
        provider: 'AndreasSteiger\\EventManager\\AppController\\DataProvider->getDateTimeNow'
    resultFormat: 'hh : mm : ss'

  duration:
    label: 'EXT:event_manager/Resources/Language/locallang_apps_eventapp.xlf:result.list...
    type: 'number'
    operation: 'difference'
    arguments:
      -
        column: 'enddate'
      -
        column: 'startdate'
    resultUnit: 'days'

  numberOfParticipants:
    label: 'EXT:event_manager/Resources/Language/locallang_apps_eventapp.xlf:result.list...
    type: 'number'
    operation: 'percent'
    arguments:
      -
        column: 'participants'
      -
        value: 150
    resultUnit: '%'
```

Abbildung 35: Konfiguration der Computed Values für Datensätze; Autor: A. Steiger, 2015

Mittels „column“ und „value“ werden mit Daten einer bestimmten Spalte oder eines definierten, statischen Wertes gearbeitet. „provider“ verweist auf sogenannte Data Provider[22], die dynamische Größen, wie beispielsweise das Datum und die Uhrzeit des jetzigen Zeitpunktes zurückgeben.

Die gleichen Operationen können für zeilenübergreifende Berechnungen innerhalb einer Spalte verwendet werden. Dazu wird eine neue Zeile unterhalb der Ergebnisdatensätze erstellt, welche eine inhaltsbezogene Zusammenfassung errechnet. Dies kann über die gesamte Ergebnismenge aller Paginierungen oder dem Resultat der aktuellen Paginierung geschehen. Jedoch muss die Performance bei großen Mengen an Datensätzen beachtet werden. Hierfür sollte eine Deaktivierung dessen oder ein manuelles Berechnen vorgesehen werden. Die Operationen richten sich nach den definierten Typen im TCA und müssen daher nicht in der Backend-App-Konfiguration angegeben werden. Der nachfolgende Screenshot zeigt eine beispielhafte Definition für eine Preiszusammenfassung in der Event-App und eine Dateigrößenberechnung in der File-App.

```
computedRows:
  settings:
    paginationSummery: true
    totalSummery: true
  columns:
    price:
      label: 'EXT:event_manager/Resources/Language/locallang_apps_eventapp.xlf:result.list...
      operation: 'sum'
      resultUnit: '€'

-------------------------------------------------------------------------------------------

    fileSize:
      label: 'EXT:event_manager/Resources/Language/locallang_apps_eventapp.xlf:result.list...
      operation: 'sum'
      provider: 'AndreasSteiger\\EventManager\\AppController\\DataProvider->getFileSizeInGb'
      resultUnit: 'GB'
```

Abbildung 36: Konfiguration der Computed Values für einzelne Spalten; Autor: A. Steiger, 2015

Bei dieser Konfiguration können ebenfalls Data Provider zum Einsatz kommen (siehe „fileSize“). Dies gewährleistet eine Erweiterbarkeit und flexible Anpassung der Werte.

[22] Englisch für: Datenanbieter

5.6 Optimierung der Darstellung

Bei einem CMS liegen häufig große Datenmengen mit viel Informationsgehalt vor. Für den Benutzer im Backend ist es daher oft wünschenswert, eine Vielzahl der Spalten in einer Tabellenansicht anzeigen zu lassen, um einen Gesamtüberblick zu erhalten. Dadurch muss auf der Benutzeroberfläche in den meisten Fällen horizontal gescrollt werden. Um diese Fälle zu verringern oder dem User das mögliche Scrollen deutlich zu machen, können diverse Optimierungen vorgenommen werden.

Zunächst werden Verbesserungen in der Spaltenbreite behandelt. Hier gilt es, die Inhalte individuell zu analysieren und dementsprechend kompakte Visualisierungsmöglichkeiten zu finden. Durch die neu eingeführten Avatare für Backend-Benutzer in der kommenden TYPO3 CMS Version 7, wird eine minimierte Darstellung für die Identifikation bereitgestellt. [Mac15] Somit können lange Namen der User durch kleine Grafiken ersetzt werden und mittels Tooltip (Schnellinfo) steht die gesamte Bezeichnung zur Verfügung. Die nachfolgende Abbildung zeigt die neuen Avatare an einem Beispiel.

Abbildung 37: Avatare bei TYPO3 CMS 7; Quelle: [Mac15]

An dieser Stelle kann angeknüpft werden. Wenn ein Backend-Benutzer kein Avatar im System festgelegt hat, können unter anderem die Initialen des Users in der kreisförmigen Visualisierung dargestellt werden. Ferner kann es bei einer Tabelle vorkommen, dass verschiedene User in einer Spalte enthalten sind. In diesem Fall kann ebenfalls eine platzsparende Lösung über einen zusätzlichen Button sowie eine Information über weitere Datenobjekte verwendet werden. Damit werden nicht alle verfügbaren Benutzer in der Tabelle nebeneinander aufgereiht, sondern zusammengefasst in einem Overlay-Fenster beim Klick auf den Button angezeigt. Das folgende Wireframe zeigt eine beispielhafte Umsetzung, wobei „+7“ eine Schaltfläche darstellt.

Abbildung 38: Komprimierte Anzeige mehrerer Backend-Benutzer in einer platzoptimierten Spalte; Autor: A. Steiger, 2015

Das gleiche Prinzip könnte für Medien-Spalten oder Spalten mit mehreren Objekten verwendet werden. Oft findet eine Schaltfläche mit drei Punkten Verwendung, um weitere Elemente anzuzeigen. Der Vorteil bei der Lösung, die im Wireframe gezeigt ist, liegt in dem Informationsgehalt. Im Gegensatz zu den drei Punkten wird hierbei der Nutzer darüber informiert, wie viele weitere Elemente sich dahinter verbergen.

Überdies können Optimierungen für überbreite Tabellen bzw. Ergebnisanzeigen vorgenommen werden. Der Benutzer muss im Falle einer solchen Tabelle oder Liste deutlich sehen, dass weitere Informationen außerhalb des Bildschirmbereiches liegen. Mit diesem Problem beschäftigte sich unter anderem das Entwicklungsteam von Doodle. Doodle ist ein Webservice, über den Umfragen und Terminplaner erstellt werden können. Bei diesen Umfragen kommt es schnell zu breiten Listen, welche nicht ohne Scrollen auf der Webseite angezeigt werden können. Um dies dem Nutzer erkenntlich zu machen, findet zuerst eine komprimierte Tabelle mit besonderer Visualisierung Verwendung. Hierbei wird mittels einer gefalteten Spaltendarstellung (siehe I, Abbildung 39) sowie über einen zusätzlichen Button (siehe II, Abbildung 39) die überbreite Ansicht hervorgehoben. Diese Elemente sind jedoch in dieser ersten Visualisierung optimiert, so dass kein horizontales Scrollen erforderlich ist. Klickt der User auf den Button oder die gefalteten Bereiche, wird die Tabelle in voller Breite dargestellt. Durch die bewusste Interaktion des Benutzers bekommt dieser den deutlichen Hinweis, dass nun weitere Inhalte außerhalb des Anzeigebereiches vorhanden sind. Der folgende Screenshot zeigt diese Lösung im kompakten Anfangszustand.

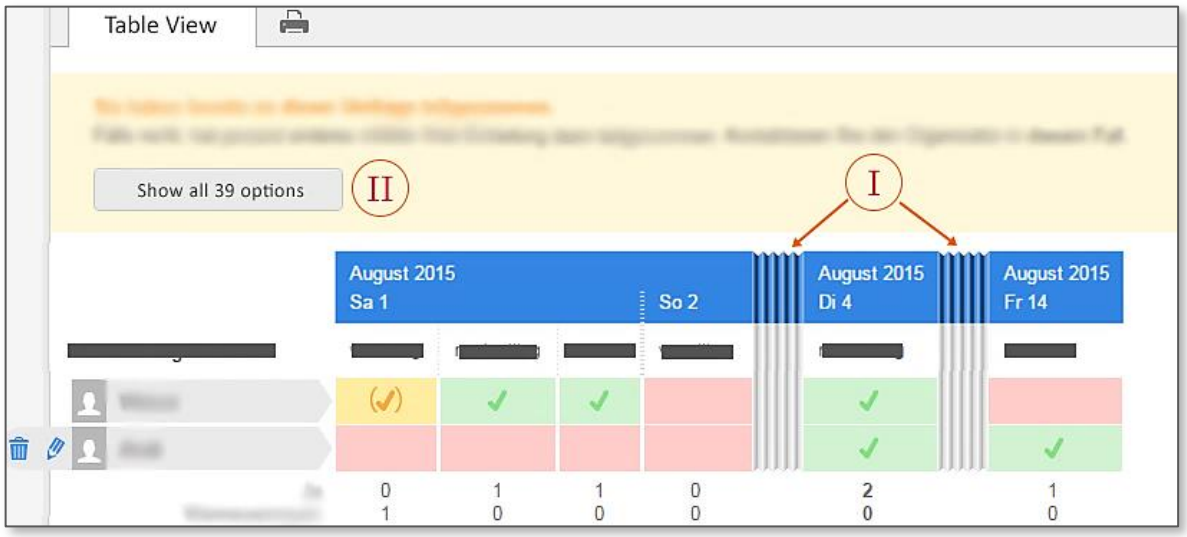


Abbildung 39: Screenshot der Tabellenkomprimierung bei Doodle; Autor: A. Steiger, 2015

Diese Umsetzung kann ebenfalls für die Tabellenansicht einer TYPO3 Backend App übernommen werden, um dem User aussagekräftige Hinweise für die Benutzung zu geben. Die folgende Abbildung enthält ein Wireframe, das eine mögliche Umsetzung einer Schaltfläche (siehe I, Abbildung 40) zum Aufklappen weiterer Tabellenspalten zeigt.

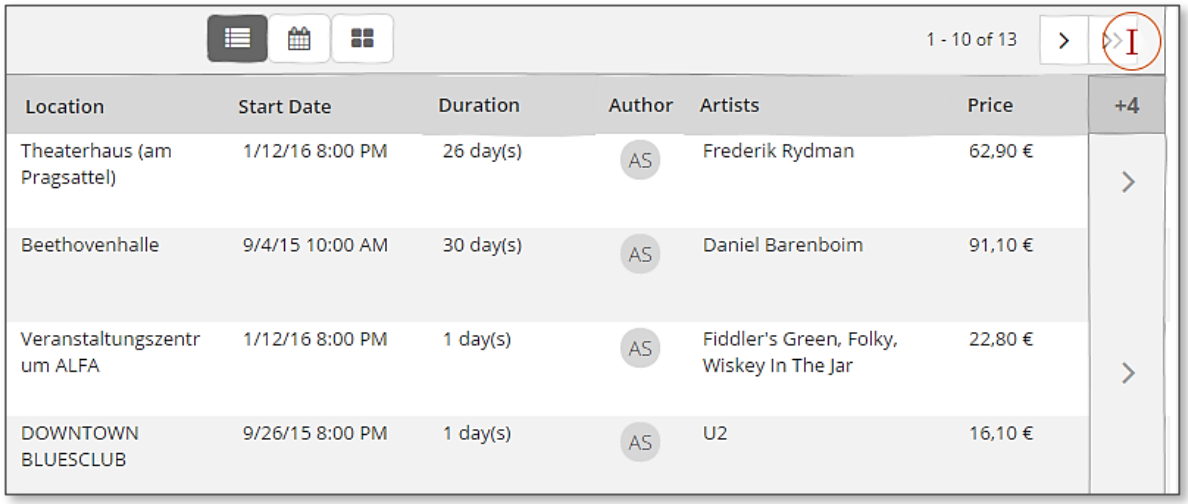


Abbildung 40: Optimierter Hinweis auf weitere Spalten und horizontales Scrollen;
Autor: A. Steiger, 2015

Zusammenfassend lassen diese Beispiele erkennen, wie wichtig es ist, die angebotenen Funktionen in Bezug auf die Usability zu analysieren und optimierte Lösungen anzubieten. Dadurch kann die Nutzungsqualität der Software erhöht werden.

5.7 Visualisierung der Dateiverwaltung

Beim Anwendungsfall der Dateiverwaltung liegt eine ähnliche Visualisierungsart analog des Veranstaltungsmanagements vor. Es wird eine Vielzahl an Informationen über mehrere Spalten bereitgestellt. Das Besondere an dieser gelisteten Ansicht sind hierbei die hohen Zeilen. Diese werden durch die Größe der Vorschaubilder (siehe I, Abbildung 41) bestimmt, welche bei der Arbeit mit Dateien hilfreich sind. Falls keine Bilder vorhanden sind, werden gleichgroße Symbolbilder des Dateitypen an dieser Stelle dargestellt. Da diese Elemente durch die Höhe der Zeile Freiräume schaffen, können Informationen untereinander platziert werden. Dies ist in Abbildung 41 im markierten Bereich I zu sehen, wobei der Dateiname, Titel sowie die Pfadangabe gemeinsam in der ersten Spalte der Tabelle positioniert sind. Weiterhin kann in dieser Visualisierungsart über die Typbezeichnung der Datei ein Filter gesetzt werden (siehe I, Abbildung 41), um somit schneller die Auswahl einschränken zu können. Das Wireframe der Dateiverwaltung ist nachfolgend abgebildet.

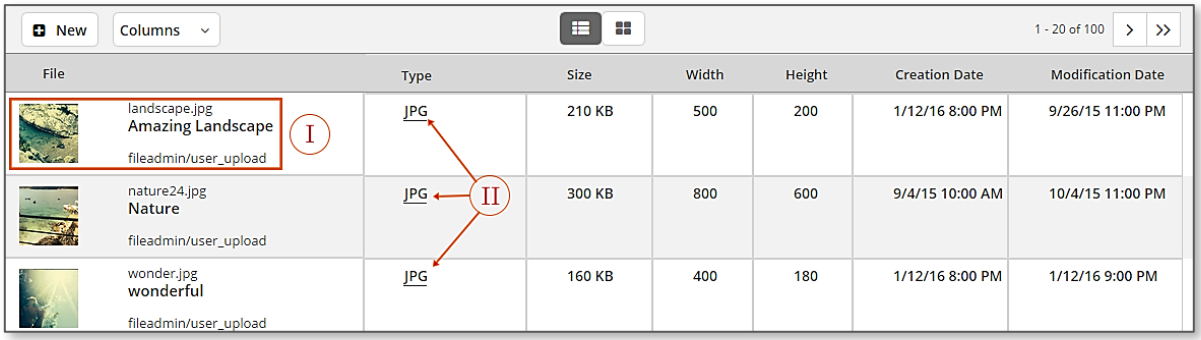


Abbildung 41: Wireframe des Inhaltsbereiches der Ergebniskomponente am Beispiel der Dateiverwaltung; Autor: A. Steiger, 2015

Um die Dateien in Dokumente und Medienelemente zu gruppieren, können zusätzliche Schaltflächen verwendet werden. Diese befinden sich analog zum Seitenwahl-Dropdown in der Filterkomponente, da an dieser Position visuell der richtige Kontext vorliegt. Nachfolgend ist die Schaltflächenplatzierung neben dem kompakten Seitenbaum dargestellt.

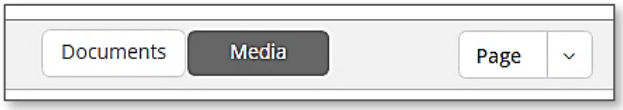


Abbildung 42: Wireframe spezieller Filtermöglichkeiten am Beispiel der Dateiverwaltung; Autor: A. Steiger, 2015

5.8 Konfiguration der Ergebniskomponente

Basierend auf den Komponenten und Funktionen des UI wird die TYPO3-Backend-App-Konfiguration der Ergebniskomponente erstellt. Dabei findet das Vererbungskonzept Anwendung, welches auf eine Grundkonfiguration aus der App Engine zugreift. Die „ListApp.yaml“ stellt unter anderem eine Basiskonfigurationsdatei für Tabellen- bzw. Listenansichten zur Verfügung. Andere Backend Apps haben die Möglichkeit, die Einstellungen einer Basis-Backend-App zu erben und diese anzupassen. Der nachfolgende Screenshot stellt den Bereich für die Ergebniskomponente der „ListApp.yaml“-Datei dar.

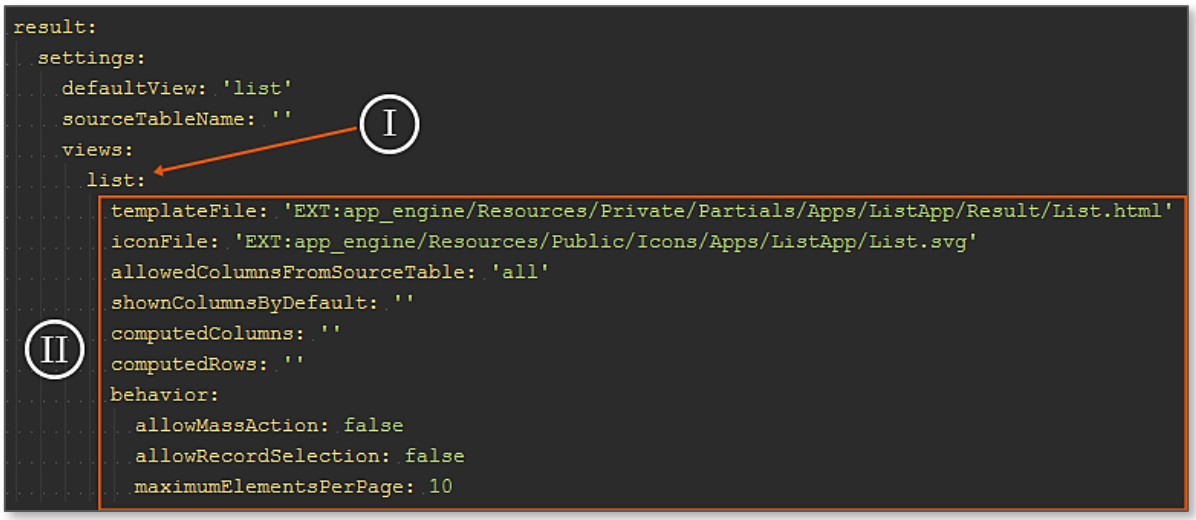


Abbildung 43: Konfiguration der Ergebnisansicht in der Datei: ListApp.yaml (App Engine); Autor: A. Steiger, 2015

Die „ListApp.yaml“ kann am Beispiel des Veranstaltungsmanagements von der Event App geerbt und manipuliert werden. Diese Datei beinhaltet ein Grundgerüst für weiterführende Einstellungen. Die sogenannte „list“-Ansicht (siehe I) stellt eine von mehreren Datensatzvisualisierungen dar. Weiterhin könnten in diesem Bereich ebenso Kachel- oder Kalenderdarstellungen definiert werden. Innerhalb der Listenansicht sind spezifische Einstellungen zu konfigurieren und Dateipfade zu setzen (siehe II). Die Konfigurationsdatei „EventApp.yaml“ aus dem Event Manager kann nun mit detaillierten sowie veranstaltungsrelevanten Informationen gefüllt werden. Dies zeigt der nachfolgende Screenshot.

Abbildung 44: Konfiguration der Ergebnisansicht in der Datei: EventApp.yaml (Event Manager); Autor: A. Steiger, 2015

Die bedeutendste Konfigurationseinstellung für die jeweilige TYPO3 Backend App ist die spezielle Quelltabelle, auf die zugegriffen wird (siehe I). Diese Information gibt an, von welcher Datenbanktabelle die Datensätze abgerufen werden. Daraufhin sind die anzuzeigenden sowie verfügbaren Spalten zu konfigurieren (siehe II). Weiterhin finden an dieser Stelle die Definitionen der Computed Values (siehe Kapitel 5.5) Platz. Die Dateien ListApp.yaml und EventApp.yaml sind auf dem Datenträger (siehe Anhang 13) finden.

6 FILTER

6.1 Definition

Der Begriff „Filter“ ist im Allgemeinen, je nach Kontext, unterschiedlich definiert. In einer TYPO3 Backend App wird ein Filter durch die Reduzierung einer Ergebnismenge beschrieben. Da eine Ergebnismenge in TYPO3 CMS als eine Ansammlung von mehreren Datensätzen vorliegt, führt ein Filter zu einer Teilmenge von Datenbankeinträgen. Dabei wird eine bestimmte Menge analysiert und schließt alle Inhalte aus, welche nicht den festgelegten Kriterien entsprechen. [Whi14] Des Weiteren wird dem Benutzer damit eine Unterstützung beim Auffinden der relevanten Informationen in der Datenvisualisierung gegeben. Technisch betrachtet handelt es sich hinsichtlich der Daten einer relationalen Datenbank um eine Datenbankabfrage mit einer Selektivität kleiner Eins. Die Selektivität beschreibt in der Informatik ein Maß für den Anteil der gefilterten Teilmenge, welcher durch eine Selektivitätsbedingung ausgewählt wird. Die Selektivitätsbedingung kann ebenfalls als Selektionsprädikat bezeichnet werden und benennt damit den Vergleichsausdruck. [Wik13a]

$$Selektivität = \frac{Anzahl\ der\ selektierten\ Einträge}{Anzahl\ der\ existierenden\ Einträge}$$

Weiterhin kann ein Filter mit logischen Operatoren verknüpft werden: UND-, ODER- und NICHT-Verknüpfung. Der UND-Operator schränkt bei der mehrfachen Verwendung iterativ die Ergebnismenge ein und bildet eine Schnittmenge der angewandten Filter. In der nachfolgenden Abbildung ist dieser Sachverhalt visualisiert.

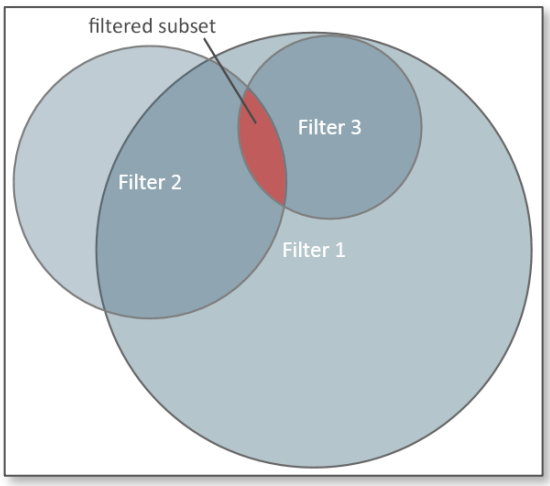


Abbildung 45: UND-Verknüpfung von Filtern mit resultierender Schnittmenge; Autor: A. Steiger, 2015

Die rot-markierte Teilmenge ist das Resultat aus den logischen UND-Verknüpfungen:

$$Filter\ 1\ \wedge\ Filter\ 2\ \wedge\ Filter\ 3$$

Eine andere Ergebnismenge würde entstehen, wenn die Filter aus Abbildung 45 mit dem ODER-Operator logisch verbunden werden:

$$Filter\ 1\ \vee\ Filter\ 2\ \vee\ Filter\ 3$$

Aus einer solchen Filterzusammenstellung resultiert nicht nur die Schnittmenge derer, sondern die gesamte Vereinigungsmenge aus allen drei Eingrenzungen. In Verbindung mit dem logischen ODER wird somit jeder einzelne Filter wiederkehrend auf die Anfangsmenge angewendet. Der Operator NICHT ist ein Spezialfall und kann lediglich mit einem Operanden und damit ausschließlich einem Filter verknüpft werden. Diese Verknüpfung könnte wiederum mit dem UND- bzw. ODER-Operator und weiteren Filtern kombiniert werden und würde damit eine komplexe Komposition erzeugen.

TYPO3 CMS arbeitet standardmäßig mit der Datenbanksprache SQL. Mit der „select“-Anweisung in SQL können Spalten von Datenbanktabellen mittels einer Abfrage ausgewählt und ausgegeben werden. Ohne jegliche Zusätze würde dies eine Selektivität von Eins ergeben, da die Anzahl der Einträge einer Ergebnismenge gleich der Anzahl der existierenden Datensätze ist. In diesem Fall wird die Menge nicht gefiltert. Die Selektivität wird beeinflusst, wenn eine SQL-Abfrage die „where“-Klausel enthält. Schematisch wird dies in der folgenden Abbildung beschrieben.

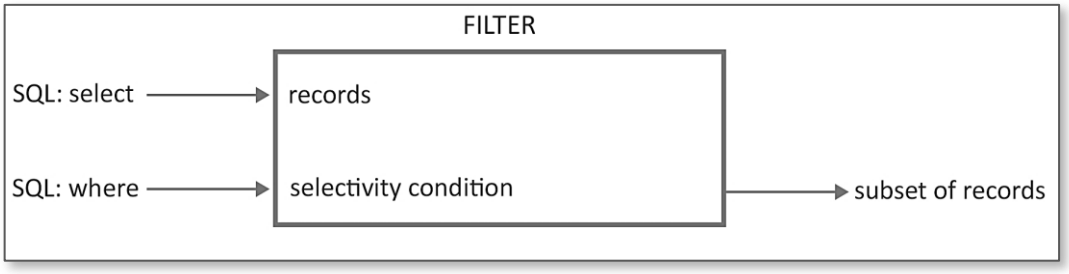


Abbildung 46: Filterdarstellung am Beispiel einer SQL-Datenbankabfrage; Autor: A. Steiger, 2015

6.2 Prädikatsdefinition von Filtern

Für die Definition des Prädikats bzw. der Bedingung eines Filters in einer TYPO3 Backend App sind verschiedene UI-Eingabemethoden vorgesehen:

a) Texteingabefeld für eine zu suchende Zeichenkette

b) Auswahlfeld für einen Filter nach dem Elternseitenobjekt

c) Vorkonfigurierte, intelligente Filter zum Auswählen und Definieren

Beim Eingabefeld aus Punkt a) wird die eingegebene Zeichenkette als Teil-String in vorher definierten Spalten gesucht. Bei Suchtreffern werden die jeweiligen Datensätze im Ergebnisbereich angezeigt. Die Spalten, welche mit dem Selektionsprädikat aus a) verglichen werden, sind in der Konfigurationsdatei der Backend App definiert. Außerdem wird die Möglichkeit in den App-Einstellungen gegeben, bei Bedarf einzelne Spalten für den Filter an- oder abzuwählen (siehe Kapitel 4.3.1). Das Auswahlfeld b) ist die kompakte Seitenbaumdarstellung der relevanten Seiten (siehe Kapitel 5.3). Hierbei wird eine Filterbedingung gesetzt, bei der das Elternelement eines Datensatzes der gewählten Seite entsprechen muss. Die genannte Eingabeart c) bietet intelligente Filter an, welche vorkonfiguriert sind und verschiedene Eingabemöglichkeiten erlauben. Hier können Bereichsfilter, Bedingungen in Verbindung mit vordefinierten Werten oder Prädikate über String-Vergleiche erstellt werden. Diese sind miteinander kombinierbar und beschreiben über unterschiedliche Eingrenzungen die gesuchten Inhalte. Bei dieser vielschichtigen Art der Ergebnismengenbeschreibung wird von sogenannten Facetten gesprochen.

6.3 Facetten

Facetten bestehen aus mehreren Einzelfiltern, welche ein Objekt bzw. einen Datensatz in unterschiedlichen Aspekten beschreiben. Dadurch kann eine präzise Eingrenzung der gesuchten Inhalte sowie eine genaue Identifizierung erreicht werden. [Whi14] Durch die detaillierte Umschreibung eines Eintrages erfordert die Integration jedoch eine komplexe Datenanalyse. Hierfür werden zunächst Inhaltselemente der TYPO3-Tabelle „tt_content“ sowie weitere Objekte, die mit der TYPO3-Erweiterung Extension Builder erstellt wurden, verglichen. Diese sind hinsichtlich der Wiederverwendung von Attributen zu betrachten. Damit zeigen sich folgende, filterrelevante Spalten eines allgemeinen Datensatzes in TYPO3 CMS:

- Erstellungsdatum („crdate“)
- Datum der letzten Bearbeitung („tstamp“)
- Backend-Benutzer, der den Datensatz erstellt hat („cruser_id“)
- Kategorien („categories“)
- Tags („tags“)
- Veröffentlichungsstatus („hidden“)
- Datum des Veröffentlichungsbeginns („starttime“)
- Datum des Veröffentlichungsendes („endtime“)

Es wurden die Identifikationsnummer des Datensatzes („uid“) und des Elternelementes („pid“) sowie weitere „t3...“-Attribute nicht aufgeführt. Diese stellen Spezialfälle dar und können bei Bedarf nachkonfiguriert werden. Weiterhin wurde die Sprachenzuordnung („sys_language_uid“) von der Liste ausgeschlossen, da die Themen der Mehrsprachigkeit und der Dimensionen von Inhaltselementen nicht zum Fokus der vorliegenden Arbeit gehören. Die oben aufgeführten Spalten sind grundsätzlich in einer Basis-Backend-App (z.B.: app_engine/Configuration/Apps/ListApp.yaml) der App Engine als Filterfacetten vorkonfiguriert. Weitere spezielle Attribute der eingesetzten Tabellen können in der Konfiguration einer Backend App (z.B.: event_manager/Configuration/Apps/EventApp.yaml) definiert werden. Die Facetten werden in diesem Zusammenhang als intelligente Filter (Smart Filter) bezeichnet. Die Konfiguration ist beispielhaft in der folgenden Abbildung gezeigt.

App Engine: ListApp.yaml

```
filter:
  settings:
    pageFilter: true
    inputField:
      allowedColumns: ''
      enabledColumns: ''
    smartFilter:
      columns:
        - crdate
        - tstamp
        - cruser_id
        - categories
        - tags
        - hidden
        - starttime
        - endtime
```

Event Manager: EventApp.yaml

```
filter:
  settings:
    inputField:
      allowedColumns:
        - tx_eventmanager_domain_model_event.title
        - tx_eventmanager_domain_model_event.location
        - tx_eventmanager_domain_model_event.artists
        - tx_eventmanager_domain_model_event.city
      enabledColumns:
        - tx_eventmanager_domain_model_event.title
        - tx_eventmanager_domain_model_event.location
    smartFilter:
      addColumns:
        - tx_eventmanager_domain_model_event.title
        - tx_eventmanager_domain_model_event.location
        - tx_eventmanager_domain_model_event.city
        - tx_eventmanager_domain_model_event.eventstart
        - tx_eventmanager_domain_model_event.eventend
```

Abbildung 47: Konfiguration für intelligente Filter in einer TYPO3 Backend App; Autor: A. Steiger, 2015

Bei der Konfiguration werden ausschließlich die Spaltenbezeichnungen angegeben, da die Inhaltstypen und die Label-Dateien bereits im TCA definiert sind. Die App Engine greift daher auf dieses Array zu, um alle weiteren Informationen zu erhalten. Damit kann die YAML-Datei einer Backend App übersichtlich bleiben und die Werte werden nicht mehrfach im System konfiguriert. Des Weiteren werden die drei Hauptfiltereinheiten (siehe Kapitel 6.2) beschrieben. Die Definition der Event App gibt exemplarisch für den String-Filter („inputField“) die im Backend anwählbaren Spalten über „allowedColumns“ an. Weiterhin werden mittels „enabledColumns“ jene Attribute aufgeführt, welche standardmäßig für den String-Filter verwendet werden.

Über den Spaltentyp kann die App Engine entsprechende Smart Filter zur Verfügung stellen. Hierbei müssen zunächst die unterschiedlichen Datentypen einer Spalte in TYPO3 CMS hinsichtlich einer Filterbedingung untersucht werden. Es kristallisieren sich folgende Filterarten heraus:

a) Einzeiliger Text (String)
b) Mehrzeiliger Text / Inhalt
c) Boolescher Wert
d) Numerischer Wert
e) Datum / Uhrzeit
f) Auswahl
g) Bilder
h) Dateien

Die Punkte a) bis f) sind die Basistypen. Bilder und Dateien fallen in die Kategorie der komplexen Typen, da diese wiederum in den einzelnen Facetten durch Basistypen beschrieben werden. Daher sind zunächst die Basistypen näher zu betrachten.

Um eine konkrete Filterbedingung zu definieren, werden dem Benutzer im UI mehrere Auswahlmöglichkeiten oder Texteingabefelder zur Verfügung gestellt. Diese erweitern nach einer Wertangabe iterativ das Prädikat, bis ein vollständiger Ausdruck vorliegt. Die folgende Abbildung zeigt ein beispielhaftes Schema für die Filterbedingung: Zeige alle Einträge, bei denen der Veranstaltungsbeginn innerhalb der letzten drei Wochen liegt.

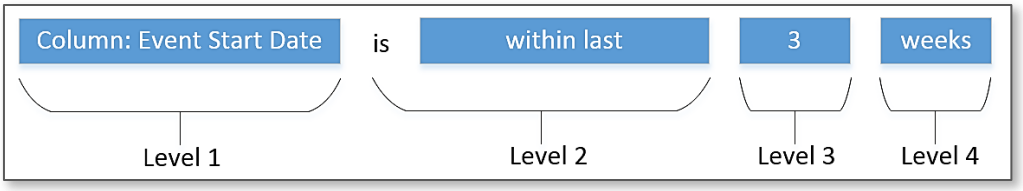


Abbildung 48: Beispielbedingung eines Basistypen von intelligenten Filtern; Autor: A. Steiger, 2015

In Abbildung 48 wird der Anwendungsfall eines Datumstyps verdeutlicht. Level 1, 2 und 4 sind an dieser Stelle austauschbare Felder. In der ersten Ebene wird die Spalte gewählt, auf der das Prädikat angewendet werden soll. Ebene 2 legt einen Vergleichsausdruck oder Gültigkeitsbereich fest. Level 3 dient bei diesem Beispiel der numerischen Wertangabe und Ebene 4 gibt die Einheit an. Dieser Sachverhalt lässt erkennen, dass individuelle Felder abhängig vom Spaltentyp benötigt werden. Die Tabelle in Anhang 5 listet alle Basistypen nach Ebenen und den jeweiligen, vordefinierten Elementen auf.

Komplexe Filterarten, wie Dateien oder Bilder, greifen auf die Basistypen zu. Soll beispielsweise nach bestimmten Inhalten der Spalte „Dateien“ gefiltert werden, ergeben sich entsprechende Auswahlkriterien für den Benutzer, um den Filter zu präzisieren. Die nachfolgende Abbildung zeigt ein mögliches Beispiel dafür.

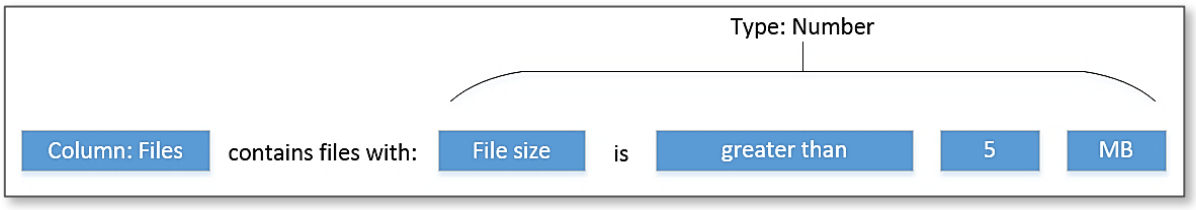


Abbildung 49: Beispielbedingung eines komplexen Typen von intelligenten Filtern; Autor: A. Steiger, 2015

Für das Kriterium der Dateigröße können ebenso andere, dateispezifische Eigenschaften gewählt werden. Demnach werden die Felder der folgenden Ebenen an die Inhaltsart angepasst. Die nachfolgende Tabelle zeigt eine mögliche Auswahl mit den dazugehörigen Basistypen.

Spaltentyp	Auswahl	Basistyp
Dateien	Dateiname Titel Dateigröße Pfad Dateityp (MIME-Type) Meta-Information	String String Numerischer Wert String String Text
Bilder	Dateiname Titel Dateigröße Pfad Dateityp (MIME-Type) Meta-Information Breite Höhe	String String Numerischer Wert String String Text Numerischer Wert Numerischer Wert

Tabelle 4: Übersicht der enthaltenen Basistypen von intelligenten Filtern für die Spalten Dateien und Bilder; Autor: A. Steiger, 2015

Bei diesen Typen können sich die Folgeschritte bis in die fünfte Ebene erstrecken. Weiterhin sind die Filterbedingungen mehrfach und in verschiedenen Ausführungen kombinierbar. Hierfür müssen die notwendigen logischen Verknüpfungen und Operanden definiert werden.

6.4 Integration und Dynamik

Um aus den verschiedenen Facetten am Ende einen Abfrageausdruck zu formulieren, müssen die einzelnen Operanden mit logischen Operatoren korrekt verknüpft werden. Die nachfolgende Abbildung zeigt beispielhaft vier Bedingungen sowie die dazugehörigen Verknüpfungen.

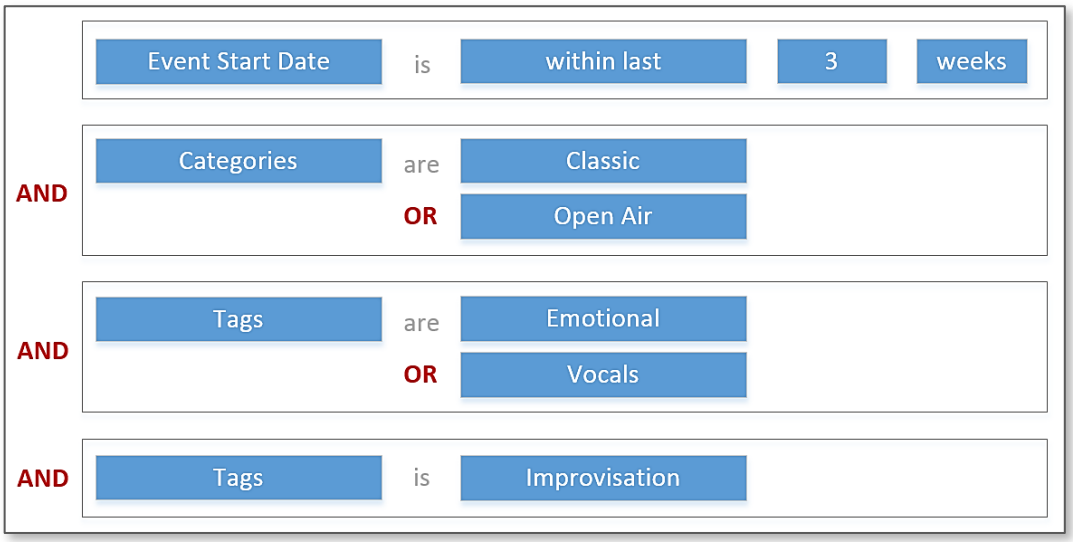


Abbildung 50: Logische Verknüpfungen der Facetten im intelligenten Filter; Autor: A. Steiger, 2015

Daraus ergibt sich der folgende, komplexe Ausdruck:

$$(Veranstaltungsbeginn < Heute \ \wedge \ Veranstaltungsbeginn > (Heute - 21\,Tage)) \wedge$$
$$(Kategorie == "Classic" \ \vee \ Kategorie \ == "Open\ Air") \wedge$$
$$(Tag == "Emotional" \vee \ Tag == "Vocals") \wedge$$
$$Tag == "Improvisation"$$

Nach diesem Prinzip werden die Filterbedingungen als Datenbankabfrage formuliert. Hinsichtlich der Integration in TYPO3 CMS könnten diese Ausdrücke in ähnlicher Form als SQL-Abfragen über eine PHP-Klasse zusammengestellt werden. Es ist jedoch zu beachten, dass eine konkret formulierte SQL-Abfrage nicht skalierbar ist und damit in der Austauschbarkeit und den Anwendungsbereichen eingeschränkt wird. Ein empfehlenswerter Weg wäre über das Query Object Model (QOM) Pattern[23] zu arbeiten. Hierfür bietet TYPO3 CMS mit Extbase (siehe Kapitel 4.2) das sogenannte Query Object an, welches auf dem Query Interface (TYPO3\CMS\Extbase\Persistence\QueryInterface) basiert.

[23] Englisch für: Muster

[Typ15d] Diese PHP-Schnittstelle bietet alle notwendigen Funktionen sowie Operationen für eine Abfrageformulierung und kann für die Erstellung eines skalierbaren Query Objects verwendet werden. Mit dieser Lösung wird eine Implementierung mit Bindung an SQL-Datenbanken umgangen und eine Abstraktion für einen möglichen Technologiewechsel geschaffen.

Ferner werden die Optionen der intelligenten Filter bei Bedarf dem Benutzer dynamisch nach der Selektivität angezeigt. Das bedeutet, wenn eine Abfrage die Selektivität gleich Null aufzeigt, werden die Möglichkeiten für diese Filterfacette nicht angeboten. Dies erfordert eine Ergebnismengenprüfung der wählbaren Filter aus der ersten Ebene, um bei einer leeren Schnittmenge diese auszublenden. Die nachfolgende Grafik beschreibt den Sachverhalt mit Hilfe des Venn-Diagramms.

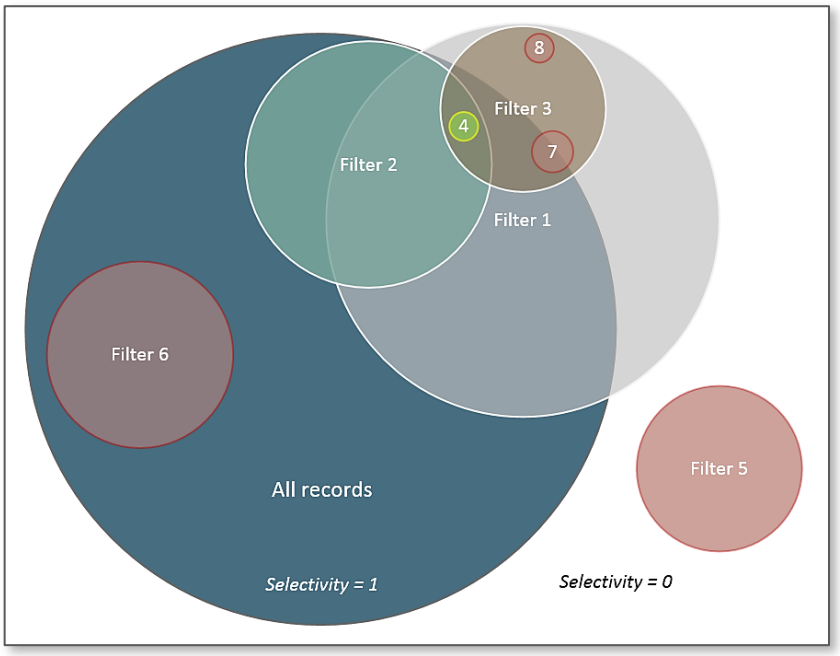


Abbildung 51: Venn-Diagramm einer Datensatzmenge und mehreren möglichen Filterbedingungen; Autor: A. Steiger, 2015

Durch die grundsätzlichen Filterverknüpfungen mit dem UND-Operator verbleibt am Ende die Schnittmenge. Somit würden nach der Anwendung von Filter 1, 2 und 3, die Filter 5, 6, 7 und 8 zu einer leeren Schnittmenge führen. Mit der Verwendung einer benutzerfreundlichen Dynamik im Filterangebot, könnte daher Filter 5 von Beginn an ausgeblendet werden. Nach dem Anwenden von Filter 1 und 2 würden zusätzlich die Filter 6, 7 und 8 entfallen. Das Ausblenden der Bedingungen bezieht sich ausschließlich auf die erste Ebene der intelligenten Filter, welche die Spalte definiert. Sind unter anderem bei einer Datensatzmenge keine Kategorien oder Tags gesetzt, würden diese Auswahlmöglichkeiten nicht angeboten werden. Es würde keine Schnittmengen geben.

Zusätzlich ist zu beachten, dass der Filter im CMS nicht der Suche eines Onlineshops gleichzustellen ist. Beim Einkaufen im Internet möchte der Benutzer in jedem Fall über eine Facettennavigation relevante Produkte finden. Im Gegensatz dazu dient der Filter bei einem CMS nicht nur der Suche von Einträgen, sondern ebenso der Überprüfung von Daten. Aus diesem Grund kann beispielsweise ein Prädikat für Dateien mit einer Größe über fünf Megabyte der Aufgabe nachgehen, das System nach großen Dateien zu durchsuchen. Dabei ist das Ziel des Benutzers nicht spezielle Einträge zu finden, sondern zu überprüfen, ob derartige Datensätze vorhanden sind. Aufgrund dessen wird eine Option in den Einstellungen der TYPO3 Backend App vorgesehen, welche es dem Anwender die Möglichkeit gibt, das dynamische Ausblenden von intelligenten Filtern ein- oder auszuschalten (siehe Kapitel 4.3.1, Abbildung 20: „Dynamic Smart Filter“).

Die genannten Aspekte benötigen weiterhin eine Entscheidungskomponente für ein Cache-Handling, um die Performance für die Abfragen zu erhöhen. Hierbei wird mit einem Data Provider gearbeitet, der bei Anfragen entscheidet, ob eine neue Datenbankabfrage gestartet wird oder die Ergebnismenge aus dem Cache geladen werden kann.

6.5 Usability und Wireframes

Die Herausforderung beim UI-Konzept liegt in dem Zusammenspiel der Komplexität und Usability. Auf der einen Seite soll es vielseitige Optionen für Facettenfilter unterstützen. Auf der anderen Seite gilt es, die Nutzungsqualität für Einsteiger und Redakteure hoch zu halten und damit eine übersichtliche Benutzeroberfläche zu schaffen. Es muss abgewogen werden, welche Elemente sofort sichtbar sind oder erst nach dem Betätigen einer Schaltfläche erscheinen. Als Leitbilder für die intelligenten Filter der TYPO3 Backend App dienen die Webanwendung Jira (siehe Kapitel 3.4.2) und die sogenannten Smart Folder im Apple Finder (siehe Kapitel 3.3.2). Im folgenden Screenshot sind die Benutzeroberfläche sowie die Filtermöglichkeiten des Finders abgebildet.

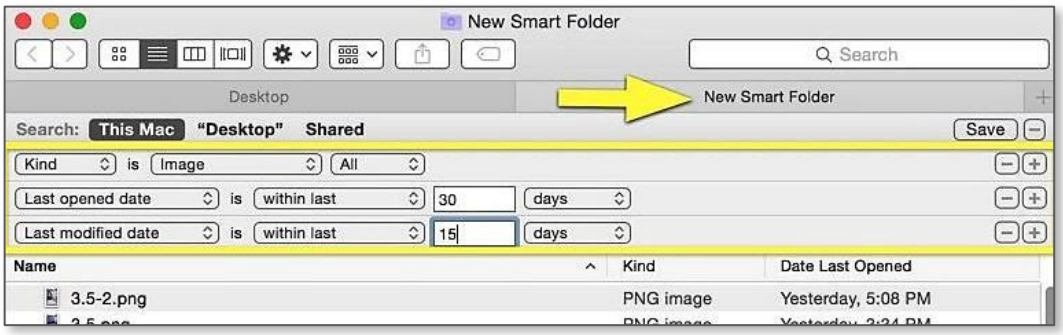


Abbildung 52: Intelligenter Ordner mit Filtermöglichkeiten im Apple Finder;
Quelle: http://www.cultofmac.com/319619/10-essential-finder-tips-for-mac/ (Abgerufen am 26.08.2015)

Im Apple Finder können beliebig viele Filter durch das Plus- oder Minussymbol hinzugefügt oder entfernt werden. Ebenso ist das Löschen aller bereits gesetzten Filter über das erste Minuszeichen neben der „Save"-Schaltfläche möglich. Die Anordnung und Kombination aus Dropdown-, Text- und Eingabekomponenten sind selbsterklärend sowie verständlich. Das Konzept von Apple entspricht dem Aufbau der intelligenten Filter in einer TYPO3 Backend App und dient damit unterstützend der Wireframe-Entwicklung.

Für die erarbeiteten Funktionen einer Filterkomponente werden verschiedene Aktionsschaltflächen benötigt:

- **I.** Schaltfläche zum Anwenden des String-Filters
- **II.** Schaltfläche zum Ausklappen der intelligenten Filtermöglichkeiten
- **III.** Schaltfläche zum Ausklappen der Seitenauswahl
- **IV.** Schaltfläche zum Entfernen gesetzter Filter
- **V.** Schaltfläche zum Hinzufügen einer Filterfacette
- **VI.** Schaltfläche zum Entfernen einer Filterfacette
- **VII.** Schaltfläche zum Anwenden des intelligenten Filters

Gemäß der Nummerierung sind die einzelnen Elemente im folgenden Wireframe markiert, welches den Aufbau der Filterkomponente zeigt.

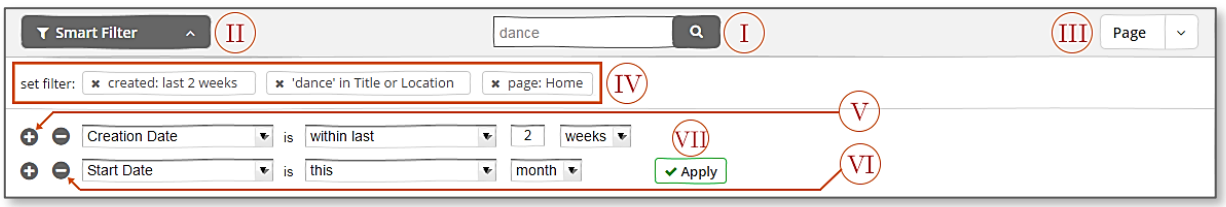


Abbildung 53: Wireframe der Filterkomponente einer TYPO3 Backend App; Autor: A. Steiger, 2015

Die Abbildung zeigt einen erweiterten Modus mit Einstellmöglichkeiten der intelligenten Filter. Im minimierten Zustand der Komponente sind lediglich die drei Hauptelemente sichtbar: Filter nach String, Filter nach Seite und Filter nach Facette (siehe Kapitel 6.2). Im Bereich IV werden alle angewandten Filter gelistet. Grundsätzlich weist die Filterkomponente eine Intra-App-Kommunikation auf, da diese den Ergebnisbereich beeinflusst. Sowohl die Ergebnismenge der Datensätze, als auch die Visualisierung kann durch einen Filter verändert werden. Beispielsweise sind beim Filtern nach einer bestimmten Zeichenkette (siehe I) zusätzlich die Treffer-Strings im Ergebnisbereich markiert. Das folgende Wireframe zeigt diese Beeinflussung der Ergebnisanzeige durch einen angewandten Filter.

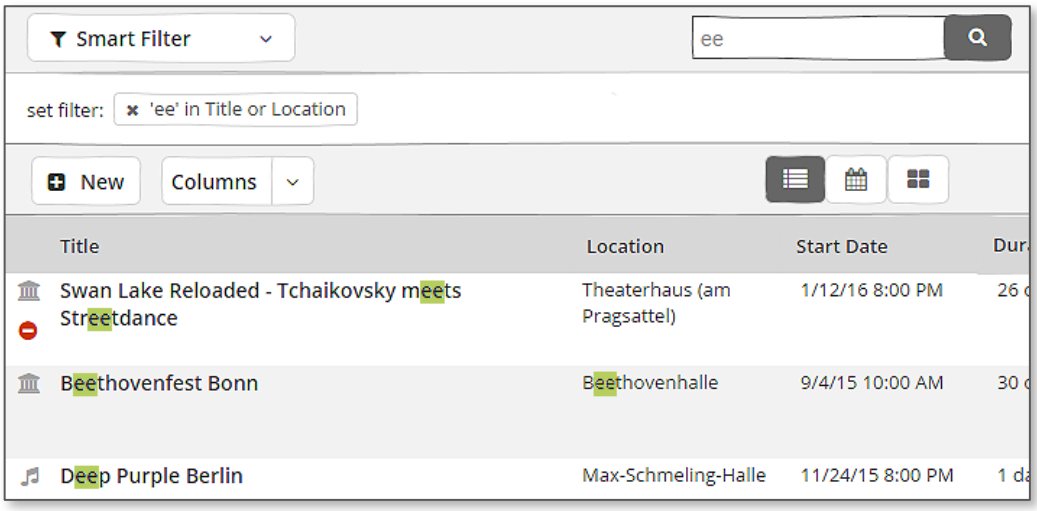


Abbildung 54: Markierung der Treffer-Strings in der Ergebniskomponente nach angewandtem Filter; Autor: A. Steiger, 2015

Ferner wird zusätzlich eine gewählte Elternseite in der Seitenauswahl (siehe III, Abbildung 53) als Filterbedingung in die Liste der gesetzten Filter aufgenommen, um somit dem Benutzer transparent Informationen über die eingeschränkte Auswahl der Ergebniskomponente zu geben. In der folgenden Abbildung wird dies verdeutlicht.

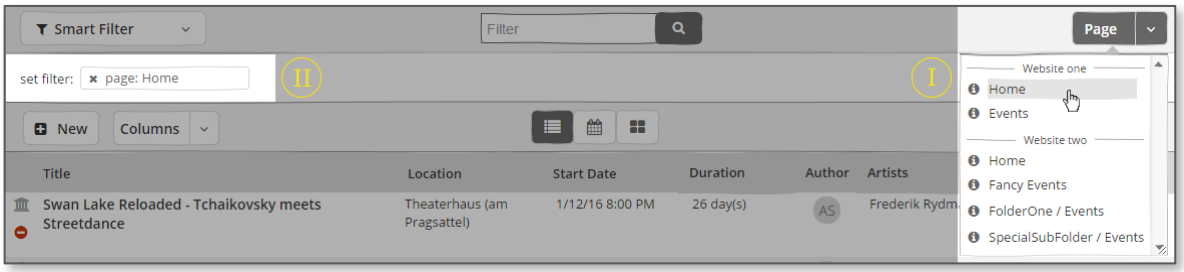


Abbildung 55: Auswahl und Anzeige des Seitenfilters; Autor: A. Steiger, 2015

Beim Wählen einer Seite im Seitenfilterbereich (siehe I) wird die Ergebnisliste entsprechend eingeschränkt und zusätzlich wird ein gesetzter Filter hinzugefügt (siehe II). An dieser Stelle kann ausschließlich ein Seitenfilter analog zum Seitenbaum gesetzt werden. Eine Mehrfachwahl der einzelnen Seiten ist hier nicht vorgesehen. Jedoch wird diese Funktion bei dem Basistypen Auswahl (siehe Anhang 5) der intelligenten Filter angeboten. So ist es unter anderem bei Kategorien oder Tags möglich, mehrere Elemente zu wählen. Diese Möglichkeit ist in der nachfolgenden Abbildung dargestellt.

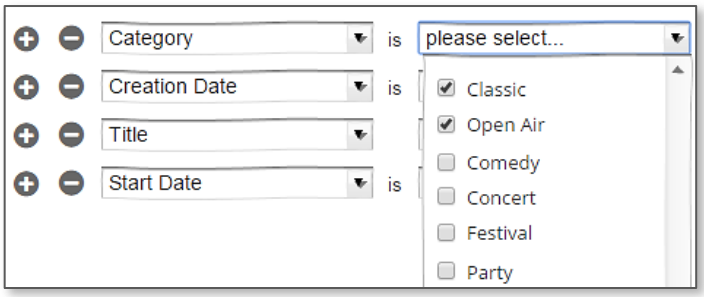


Abbildung 56: Mehrfachauswahl einer Facette des intelligenten Filters; Autor: A. Steiger, 2015

Die logische Verknüpfung innerhalb des Auswahlfeldes wird mit einem ODER realisiert. Zusätzlich kann der Benutzer eine weitere Facette hinzufügen und erneut auf Kategorien bzw. Tags die bereits angewandten Filter einschränken. Da jede neue Facette mit einem logischen UND verknüpft wird, können dadurch beispielsweise komplexe Kategoriefilter mit verschiedenen logischen Operatoren erstellt werden.

6.6 Presets

Nachdem alle Backend-App-Komponenten erläutert wurden, können die konkreten Inhalte der Presets entwickelt werden. Aufbauend auf den erarbeiteten Ergebnissen der Kapitel 4.3.3 und 4.5 sind die Konfigurationen dieser Ansichtsvorlagen zu definieren. Am Beispiel der Tabellendarstellung werden in einem Preset die folgenden Ansichtseinstellungen gespeichert:

- Alle gesetzten Filter
- Gewählte Ansicht bzw. Datenvisualisierung (Tabelle, Kacheln, Kalender, ...)
- Ansichtsspezifische Eigenschaften:
    - Sichtbare Spalten
    - Spaltenanordnung
    - Spaltenbreite
    - Aktive Sortierung

Diese Eigenschaften sind für Presets einzelner Backend-Benutzer in der Datenbank persistiert oder global für alle User in der Backend-App-Konfiguration festgelegt. Ersteres sind die Vorlagen, die von Benutzern individuell erstellt werden können und in der vorgesehenen Datenbanktabelle (siehe Kapitel 4.5, Abbildung 25) im Attribut „settings“ als JSON-String definiert werden. Dieser ist für den Anwendungsfall des Veranstaltungsmanagements mit Beispieldaten im Anhang 6 aufgeführt. Es sind die Inhalte von genau einem Preset aufgeführt. Grundsätzlich werden die Einstellungen in die Bereiche „name“, „filter“ und „view“ aufgeteilt. Die Bezeichnung (siehe I) kann an dieser Stelle als String persistiert werden, da der Benutzer den Namen selbst festlegt. Damit ist ein Label für die Mehrsprachigkeit nicht notwendig. In dem Bereich „filter“ (siehe II) sind alle gesetzten Einschränkungen in der Backend App mit Ausdrücken der intelligenten Filter beschrieben. Es fällt auf, dass sowohl der String- (siehe III) als auch der Seitenfilter (siehe IV) in diesem Format definiert sind. Die einzelnen Kriterien in diesem Teil sind jeweils mit dem UND-Operator verknüpft. Im Gegensatz dazu werden die Elemente innerhalb eines Kriteriums mit der ODER-Verknüpfung verbunden. Der „view“-Sektor (siehe V) beinhaltet alle ansichtsspezifischen Einstellungen.

Analog dazu ist die YAML-Struktur der Backend-App-Konfiguration aufgebaut. Diese ist im Anhang 7 mit den gleichen Bereichsmarkierungen dargestellt. Ein Unterschied ist in der Bezeichnung (siehe I) zu erkennen, die für eine Mehrsprachigkeitsunterstützung vorgesehen ist. Da die Presets aus der App-Konfiguration für alle Benutzer im TYPO3 Backend zur Verfügung stehen, muss die Übersetzung für weitere Sprachen sichergestellt sein.

Grundlegend werden die Definitionen als „buttons“ (siehe Anhang 7 oben) bezeichnet, da visuell neue Schaltflächen für Ansichtsvorlagen zur Backend App hinzugefügt werden. Die nachfolgende Abbildung veranschaulicht den Beginn der Preset-Konfiguration bis zur Button-Definition.

```
component: 'presets'
settings:
  globalGroups:
    categories:
      label: 'EXT:event_manager/Resources/Private/Language/locallang_apps_eventapp.xlf:presets.categories'
      autoGenerate:
        - tx_eventmanager_domain_model_event.categories

      buttons:
```

Abbildung 57: Ausschnitt der Preset-Konfiguration einer TYPO3 Backend App; Autor: A. Steiger, 2015

Über die Konfiguration der „globalGroups“ können Bereiche in der Benutzeroberfläche für Schaltflächen erzeugt werden. „categories“ stellt bei diesem Beispiel eine solche Sektion dar. Mit dem Array „autoGenerate“ können automatisiert globale Vorlagen aus bestimmten Spalten erstellt werden. Diese Datenbankattribute müssen dabei vom Basistyp Auswahl (siehe Anhang 5) sein.

Das UI der Preset-Komponente besteht daher aus folgenden Elementen:

I. Bereich für die eigenen Presets

II. Bereich für die globalen Presets

III. Label zur Anzeige des gewählten Presets

IV. Schaltfläche zum Speichern und Erstellen eines Presets

V. Schaltfläche zum Aktivieren eines erweiterten Modus, um eigene Presets zu entfernen

Analog zu dieser Nummerierung sind im folgenden Wireframe die benannten Elemente dargestellt.

Abbildung 58: Wireframe der Preset-Komponente einer TYPO3 Backend App; Autor: A. Steiger, 2015

Die Bereiche I und II sind erst sichtbar, sobald die linke obere Schaltfläche „Presets“ betätigt wird. Beim Klick auf IV wird eine Dialogbox geöffnet. Nachfolgend wird das Wireframe dafür gezeigt.

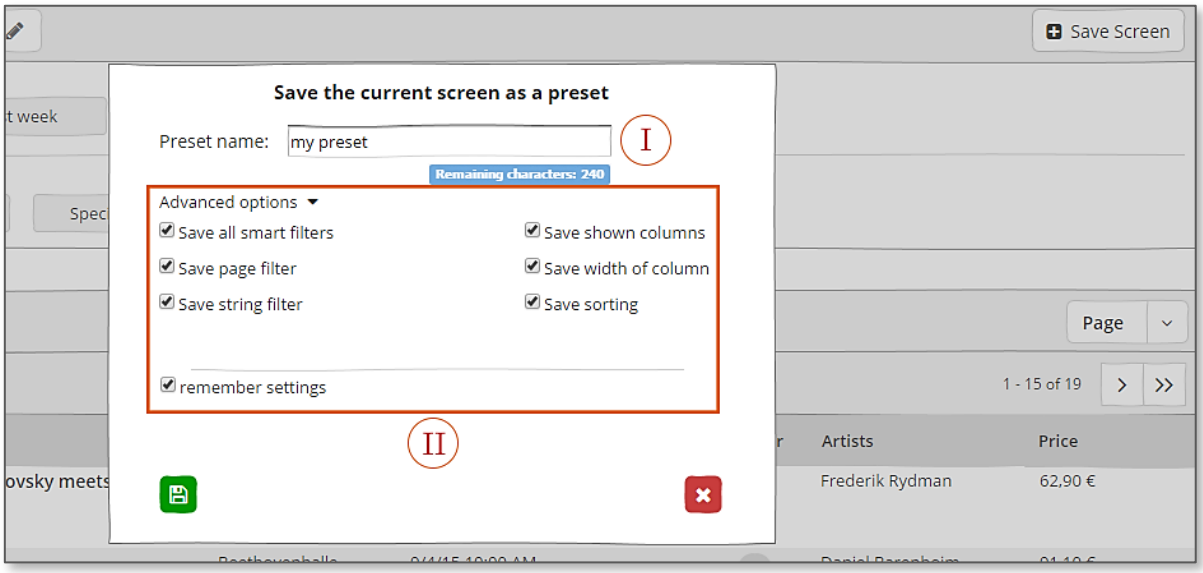


Abbildung 59: Wireframe des Fensters zum Sichern eines Presets in einer TYPO3 Backend App; Autor: A. Steiger, 2015

Das Fenster in Abbildung 59 fordert eine Namensvergabe der neuen Ansichtsvorlage (siehe I) sowie die Bestätigung der zu sichernden Eigenschaften. Über eine erweiterte Ansicht können detaillierte

Festlegungen der Preset-Einstellungen getroffen werden (siehe II). Bei der Vergabe einer bereits existierenden Vorlagenbezeichnung wird der Benutzer vom System darauf hingewiesen. Dabei können vorhandene Presets mit neuen Eigenschaften überschrieben werden.

Darüber hinaus wird das Entfernen von bereits gesicherten Vorlagen in der Preset-Komponente erschwert, um ein versehentliches Löschen zu vermeiden. Ferner wird ein weiterer Modus über das Stift-Symbol (siehe V, Abbildung 58) erreicht. Nach dem Betätigen dieser Schaltfläche können selbst erstellte Presets über ein Kreuz-Icon entfernt werden. Nachfolgend wird dieser Modus im Wireframe verdeutlicht.

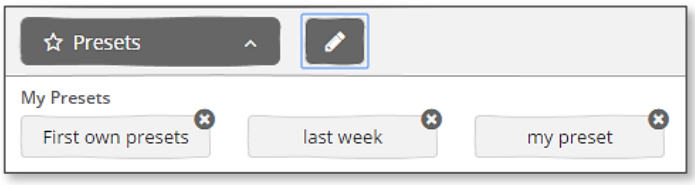


Abbildung 60: Wireframe des Entfernen-Modus für eigene Presets in einer TYPO3 Backend App; Autor: A. Steiger, 2015

7 BEWERTUNG UND AUSBLICK

7.1 Empirische Bewertung der Usability

7.1.1 Testverfahren

Mittels der entwickelten Konzepte kann ein Prototyp der Benutzeroberfläche mit den jeweiligen Funktionen umgesetzt werden (siehe Anhang 13: „Prototyp“). Anhand dieses Click-Dummies wird ein A/B-Testverfahren durchgeführt, um die Usability in den Ergebnissen der vorliegenden Arbeit messbar auszuwerten. Dieses Testverfahren analysiert die Benutzung zweier Softwarevarianten an einer Testperson. Version A stellt dabei den IST-Zustand des TYPO3 CMS in der Version 7.4 dar. Version B ist der entwickelte Prototyp, welcher mit dem Konzept der TYPO3 Backend Apps arbeitet. Beide Varianten beinhalten die gleichen Daten und basieren auf dem Anwendungsfall des Veranstaltungsmanagements (siehe Kapitel 3.6). Für das A/B-Testverfahren werden User Stories[24] verwendet, um die einzelnen Features an Redakteuren testen zu können. Die Probanden haben die Aufgabe, die Anwendungsszenarien zuerst mit Variante A und danach mit Variante B zu erledigen. Das Verfahren wird von einem Testleiter geführt, der die User Stories als Aufgaben stellt. Sobald die Testperson für eine User Story länger als drei Minuten benötigt, wird der Testschritt abgebrochen. Probanden sollen ihre Gedanken aussprechen, damit der Testleiter die Wege der Aufgabenerfüllung nachvollziehen und aussagekräftig dokumentieren kann.

Die Erarbeitung der User Stories wird anhand der neuen Funktionen hinsichtlich der alltäglichen Redakteursarbeit mit dem CMS entwickelt. Im Anhang 8 sind die erarbeiteten User Stories aufgeführt. Da in der Version A keine intelligenten Filter, Backend-App-Einstellungen oder eine Ansichtswahl für verschiedene Datenvisualisierungen verfügbar sind, werden im ersten Testverfahren lediglich die User Stories US1 bis US7 bearbeitet. Die nachfolgende Tabelle zeigt welche Features in den User Stories getestet werden.

[24] Englisch für: Anwendererzählungen

Features	User Story
String-Filter	US1
Eintrag editieren	US1
Seitenfilter/Seitenwahl	US2
Sortierung	US3
Preset erstellen/speichern	US4
Preset laden	US5
Spaltenwahl	US6
Computed Values	US6
Neuen Datensatz erstellen	US7
Auswahl verschiedener Ansichten	US8
Backend-App-Einstellungen	US9
Intelligente Filter	US10

Tabelle 5: Ausgewählte Features, welche mit bestimmten User Stories getestet werden; Autor: A. Steiger, 2015

Für die vorliegende Ausarbeitung wurden vier Testpersonen ausgewählt, die einen unterschiedlichen Erfahrungsstand mit dem TYPO3 CMS vorweisen. Die einzelnen Profile der Probanden werden im Anhang 9 beschrieben. Da es ebenso wichtig ist Benutzer außerhalb der Zielgruppe zu testen, wurde Testperson 3 ebenfalls für das Verfahren herangezogen. [Kru14, S.120 f.]

Für das A/B-Testverfahren werden Bewertungskriterien gewählt, welche auf den Eigenschaften der Nutzungsqualität bzw. Usability basieren (siehe Kapitel 2.3.2). Damit kann eine Usability-Messung aus den Testergebnissen ermittelt werden. In der nachfolgenden Tabelle sind die betrachteten Bewertungskriterien hinsichtlich der Nutzungsqualitätseigenschaften aufgeführt.

	Bewertungskriterium	Eigenschaft der Software-Nutzungsqualität
a)	Erfolgreiches Erledigen der Aufgabe	Nützlich, erlernbar, effektiv, begehrenswert
b)	Benötigte Zeit	Erlernbar, einprägsam, effizient, begehrenswert
c)	Anzahl der Fehlschritte	Erlernbar, einprägsam, effizient
d)	Benötigte Hilfestellung durch den Testleiter	Erlernbar, einprägsam
e)	Verständnisprobleme bei der Bearbeitung	Erlernbar, einprägsam

Tabelle 6: Gegenüberstellung der Bewertungskriterien und den zu betrachtenden Nutzungsqualitätseigenschaften im A/B-Testverfahren; Autor: A. Steiger, 2015

7.1.2 Auswertung

Nach der Durchführung der Testverfahren werden die Ergebnisse ausgewertet. Hierbei wird eine Bewertung nach dem Schulnotensystem vorgenommen. Der Bewertungsmaßstab für die verschiedenen Kriterien wird nachfolgend in einer Übersicht verdeutlicht.

Kriterium	Note					
	1	2	3	4	5	6
a)	erledigt	-	-	-	-	abgebrochen
b)	00:00 - 00:29	00:30 - 00:59	01:00 - 01:29	01:30 - 01:59	02:00 - 02:29	02:30 - 3:00
c)	0	1	2	3	4	5
d)	nein	-	-	-	-	ja
e)	nein	-	-	-	-	ja

Tabelle 7: Bewertungsmaßstab für die Ergebnisse des A/B-Testverfahrens; Autor: A. Steiger, 2015

Damit ergeben sich die Ergebnistabellen aller vier Testpersonen im Anhang 10. Da das Bewertungskriterium a) eine bedeutende Aussage über die Usability der Software gibt, wird diese Benotung doppelt gewichtet. Somit ergibt sich eine abschließende Durchschnittsberechnung der einzelnen User Stories in den Bewertungstabellen:

$$\frac{2 * Note\ a) + Note\ b) + Note\ c) + Note\ d) + Note\ e)}{6} = Durchschnittsnote\ je\ User\ Story$$

Die Ergebnisse der Testperson 3 fallen hierbei besonders auf. Diese besitzt keine Erfahrungswerte in der Arbeit mit TYPO3 CMS und konnte daher aufgrund diverser Verständnisprobleme das Backend nicht bedienen. Die Testschritte für die Version A wurden deshalb abgebrochen. Jedoch bringen die Resultate der Version B interessante Zahlen hervor. Die gemessenen Werte unterscheiden sich nicht erheblich von denen der erfahrenen TYPO3-Redakteure. Daher wurde durch die TYPO3 Backend Apps ebenso ein besserer Zugang für Neueinsteiger geschaffen sowie der Lernaufwand verringert.

Anhang 11 zeigt die visuellen Auswertungen der benötigten Zeiten zum Bearbeiten der User Stories. Durch die Lage der linearen Regressionsgeraden wird der durchschnittliche Zeitbereich deutlich. Dabei liegt Version A in allen Fällen über der Zeit von Variante B. Zusammenfassend werden die folgenden gemittelten Bearbeitungszeiten je User Story grafisch dargestellt. Es sind ebenso die beiden unterschiedlichen Regressionsergebnisse verzeichnet.

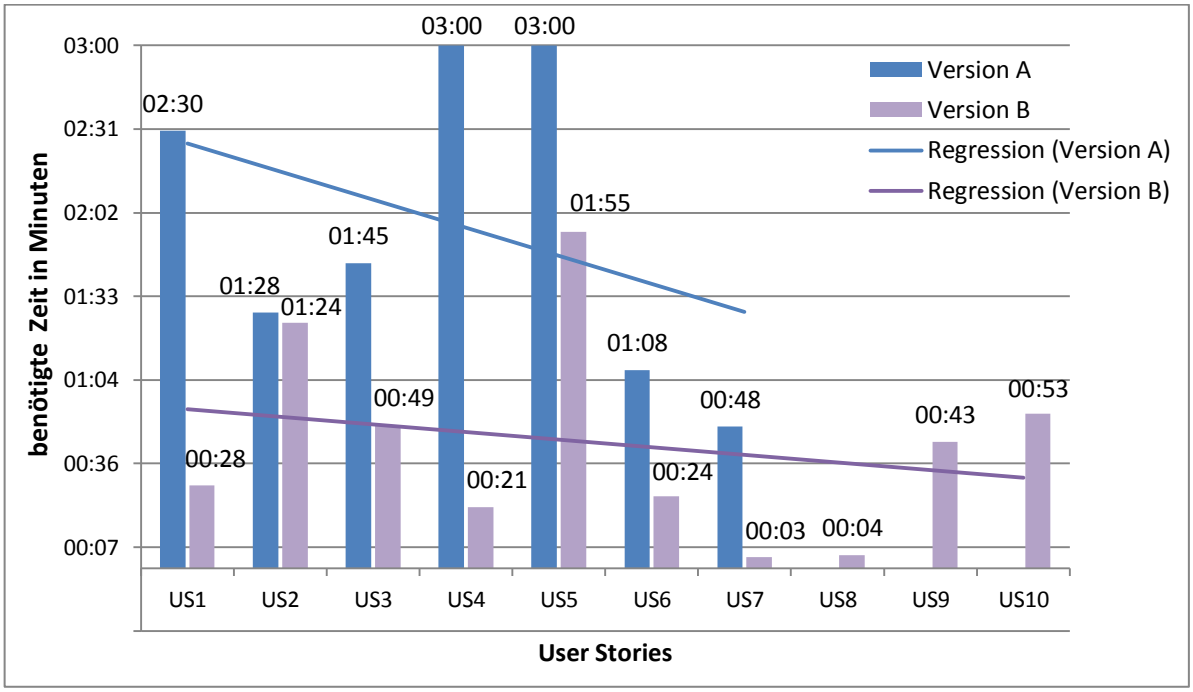


Abbildung 61: Mittelwerte der benötigten Zeiten über alle Testpersonen; Autor: A. Steiger, 2015

Des Weiteren ergibt sich die folgende Resultatstabelle aus den Mittelwerten der Notenbewertung.

	Testperson 1	Testperson 2	Testperson 3	Testperson 4	**Abschlussbewertung**
A	3,3	2,7	6,0	3,2	**3,8**
B	1,7	1,8	2,3	1,8	**1,9**

Tabelle 8: Mittelwerte der Benotung mit Abschlussbewertung für die Testversionen A und B; Autor: A. Steiger, 2015

Anhang 12 stellt diese Zahlen in einem abschließenden Diagramm dar und macht damit visuell den Unterschied in der Benutzerfreundlichkeit deutlich. Die Übersicht sowie das Diagramm zeigen somit eine messbare Verbesserung in der Abschlussbewertung der Usability bzw. Nutzungsqualität durch das neue Konzept der TYPO3 Backend Apps.

7.2 Zusammenfassung

Mit dieser abschließenden Bewertung lässt sich insgesamt ein positives Fazit über den Einsatz von TYPO3 Backend Apps ziehen. Mit den entwickelten Usability- und Software-Konzepten konnte ein Gesamtüberblick über die Komponenten sowie eine skalierbare Integrationsmöglichkeit geschaffen werden. Eine Verbesserung der Usability für TYPO3-Endwanwender wurde messbar bestätigt, wodurch die Ziele Z1, Z6 sowie Z7 (siehe Kapitel 1.3) der vorliegenden Arbeit erreicht werden. Die

UI-Komplexität (Z3) wurde durch zusätzliche erweiterte Ansichten, die nicht sofort sichtbar sind, in der normalen Benutzeroberfläche minimal gehalten. Eine rollenspezifische Komplexität (Z2) konnte aufgrund der umfangreichen Rechte-Thematik lediglich ansatzweise erlangt werden und muss in den weiterführenden Schritten außerhalb dieser Masterarbeit ausführlich herausgearbeitet werden. Voreinstellungen für die Benutzer sind über die Backend-App-Konfiguration definierbar und bestätigen damit das Ziel Z5.

Das aufgabenfokussierte Arbeiten (Z8) stellt sich als eine benutzerfreundliche Herangehensweise heraus. Das Listenmodul bzw. ein TYPO3 Backend Modul kann zukünftig durch mehrere Backend Apps ersetzt werden. Durch solch eine Kapselung von Aufgaben, wird dem Anwender eine übersichtliche Benutzeroberfläche mit kontextbezogenen Funktionen geboten. Im Kapitel 5 konnten verschiedene Aufgabenbereiche und Visualisierungsmöglichkeiten für das Listenmodul erarbeitet werden.

Das Integrations- und Konfigurationskonzept bietet richtungsweisende Anregungen für eine Umsetzung im TYPO3 CMS. Jedoch konnten dabei nicht alle Aspekte Beachtung finden, da diese erst in einer praktischen Implementierung deutlich werden. Dabei wird das TYPO3-Backend-App-Konzept mit den technischen Eigenheiten von TYPO3 CMS konfrontiert und zeigt dementsprechend ein bestimmtes Verhalten auf das speziell eingegangen werden muss. Dies konnte in der theoretischen Abhandlung der vorliegenden Arbeit nicht simuliert werden.

Grundsätzlich wird deutlich, dass sich hinter der Backend-App-Thematik ein vielschichtiger Themenkomplex verbirgt, welcher aufgrund des Umfangs nicht in jedem Detail erarbeitet werden konnte. Ebenso kann bei der Datenvisualisierung nicht von einer richtigen oder falschen Umsetzung gesprochen werden. Es sind zahlreiche Möglichkeiten und Freiheiten für eine individuelle Ansicht denkbar, welche auf den jeweiligen Aufgabenbereich zugeschnitten sind. Die vorliegende Ausarbeitung gibt daher Ideen sowie Anregungen für eine aufgabenoptimierte Visualisierungsmöglichkeit von Tabellen bzw. Listen. Die Resultate der Usability-Testverfahren zeigen, dass die entwickelten Datendarstellungen durchaus positiv auf den Benutzer wirken und damit erfolgreich konzeptioniert wurden. Zusammenfassend liefert das Konzept von TYPO3 Backend Apps einen Produktivitätsgewinn für die Arbeit mit dem CMS und sollte zukünftig in das System integriert werden.

7.3 Ausblick

Im Hinblick auf den getesteten Prototypen haben sich einige UI-Probleme gezeigt, die in den weiterführenden Schritten und vor der Implementierung überarbeitet werden sollten. Beim Testverfahren

sind Verständnisschwierigkeiten in der Bedeutung von Presets und Seitenfilter aufgetreten. Die Seitenauswahl könnte mit Hilfe des Labels „Veranstaltungen auf Seite:“ in der Verständlichkeit verbessert werden. Zusätzlich sind in dem Seitenfilter-Dropdown die Identifikationsnummern der Seiten („uid“) zur besseren Unterscheidung zu integrieren. Der Begriff „Presets“ sollte in diesem Zusammenhang mit „Screen Presets“ oder „Load Screen“ (analog zu „Save Screen“) ersetzt werden. Des Weiteren sind die beiden rechtsplatzierten Schaltflächen zum Sichern eines Presets und zum Wählen eines Seitenfilters schlecht sichtbar, da der Lesefluss des Benutzers von links nach rechts verläuft. Bei größeren Abständen zwischen den Elementen werden daher weitere Komponenten am rechten Rand übersehen. Die benannten Schaltflächen sollten im linken Seitenbereich platziert werden, um somit im Fokus des Benutzers zu bleiben. Darüber hinaus vermissten einige Testpersonen diverse Funktionen für eine bessere Bedienbarkeit. Darunter zählen unter anderem ein Tooltip bei der Paginierung, welcher über einen Hinweis auf die Konfigurationsmöglichkeiten in den App-Einstellungen verweist sowie ein zusätzlicher Button im Filterbereich, der alle gesetzten Filter mit einem Klick entfernt.

Weiterhin könnte die Benutzerfreundlichkeit der Filterkomponente verbessert werden, wenn im Eingabefeld automatische Vorschläge zu den meistgenutzten Filtern angeboten würden. Ebenso könnte dem Benutzer bei einer leeren Ergebnismenge über Empfehlungen wie „Meinten Sie den Begriff“ oder „Ähnliche Begriffe“ alternative Suchbegriffe angezeigt werden. Besonders wichtig wären deutliche Hinweise, die bei allen Filteraktionen eine Systemreaktion zeigen. Hierbei könnten dem Benutzer beispielsweise Hinweismeldungen im oberen Bereich helfen. Ein Ladebalken beim Suchen nach Datensätzen würde zudem Informationen über den Zustand des Systems liefern können.

Ein weiterer Aspekt zur Verbesserung der Funktionsverständlichkeit einer TYPO3 Backend App wären interaktive Bedienungsanleitungen, welche beim ersten Öffnen angeboten würden. Über animierte Anleitungen könnte dem Anwender gezeigt werden, welche Aktionen über die einzelnen Komponenten möglich sind und an welcher Stelle die wichtigsten Informationen aufgeführt werden. Diese Hilfestellungen könnten entsprechend der eingesetzten Komponenten automatisiert angeboten werden, da beispielsweise der Filter oder der Kopfbereich der Ergebniskomponente in den meisten Fällen die gleichen Funktionen aufzeigen.

Hinsichtlich der Export-Funktion könnten zusätzlich zur Speicherung einer CSV-Datei, welche im aktuellen Listenmodul von TYPO3 CMS bereits zur Verfügung steht, weitere Möglichkeiten angeboten werden. Ansichten sollten druckbar sowie exportierbar in die Dateiformate PDF, JPEG oder XML sein. Außerdem könnten benutzerdefinierte Presets als Datei gespeichert werden, um diese in andere TYPO3 Backend Apps zu importieren.

Ein zukünftiges Problem bei der Arbeit mit Backend Apps wird voraussichtlich die steigende Anzahl an Elementen im seitlichen Modulbereich des TYPO3 CMS sein. Mit einer Vielzahl an Backend Apps im Menü geht die Übersichtlichkeit verloren und der Benutzer ist gezwungen zu scrollen und seine relevante App zu suchen. Lösungsansätze zur Wahrung der Übersichtlichkeit wären beispielsweise die Einführung eines Dashboards in TYPO3 CMS sowie ein App-in-App-Konzept, um die Elemente im System gezielt zu verteilen.

Nach einer Implementierung der App Engine in das TYPO3 CMS und einer funktionsfähigen Backend-App-Integration kann das Konzept des App Builders (siehe Kapitel 4.4) ausgearbeitet werden. Hierbei sind Themen wie die Erstellung einer Backend App mit nur einem Klick, ein intelligenter Konfigurationsassistent sowie eine zusätzliche Benutzeroberfläche mit umfangreichen Interaktionsmöglichkeiten zum Kreieren einer individuellen Backend App zu betrachten. In allen genannten Fällen soll damit das Schreiben von Programm- bzw. Konfigurations-Code vermieden werden, um Integratoren eine Hilfestellung anzubieten.

Da in der vorliegenden Ausarbeitung die Schwerpunkte auf der Funktion und Benutzeroberfläche einer TYPO3 Backend App liegen, sind viele Aspekte der technischen Seite offen geblieben. Aufgrund des begrenzten Umfangs der vorliegenden Abhandlung und der limitierten Zeit dafür sind weitere Konzepte für die Mehrsprachigkeit und die Dimensionen von Datensätzen zu erarbeiten. Weiterhin sollte die Integration der Arbeitsbereiche (Workspaces) und der Versionierung sowie das Rechtemanagement für Backend-Benutzer konzipiert werden. Zusätzlich sind Grundlagen zu einem Datenaustausch über AJAX und dem eventuellen Einsatz eines Javascript-Frameworks mit Model-View-Controller-Unterstützung zu entwickeln. Darüber hinaus wäre ein Konzept zum Offline-Editieren zu entwerfen, wobei die Gefahren beim gleichzeitigen Zugriff auf Entitäten sowie bei der Parallelität der Bearbeitung (Concurrency) beachtet werden sollten.

8 QUELLENVERZEICHNIS

[Alt02] W. Altmann, *„Doc: Handbuch für Redakteure“*, webconcepts & solutions GmbH, Veröffentlichung: 2002. PDF [Online]. URL: https://www.typo3.tu-berlin.de/uploads/media/070909_handbuch_redak_02.pdf. [Abgerufen: 07. September 2015].

[Bas14] E. Baschny, *„TYPO3 CMS Active Contributors Meeting, 2014“*, TYPO3, Veröffentlichung: 03. April 2014. [Online].
URL: http://typo3.org/news/article/typo3-cms-active-contributors-meeting-2014/. [Abgerufen: 08. Juni 2015].

[Bed15] A. Bedford u. a., *„4 iOS Regeln, die Sie brechen sollten“*, Institut für Software-Ergonomie und Usability, Veröffentlichung: 19. Juli 2015. [Online].
URL: http://www.usability.ch/news/4-ios-regeln-verletzen.html. [Abgerufen: 11. August 2015].

[Bli05] L. Blickhan, *„Ao. Univ.-Prof. Ing. Dr. Michael Trimmel Human Factors im Kontext der ICT Inhalt Theorie: Ergonomische Bedingungen und individuelle Bedingungen die Leistungen.“*, UsabilityGeek, Veröffentlichung: 2005. [Online].
URL: http://slideplayer.org/slide/1338130/. [Abgerufen: 17. Juni 2015].

[Bre13] C. Breitenstrom u. a., *„Sicherheitsstudie Content Management Systeme (CMS)“*, Bundesamt für Sicherheit und Informationstechnik, Veröffentlichung: 2013. PDF [Online]. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/CMS/Studie_CMS.pdf?__blob=publicationFile. [Abgerufen: 30. Juli 2015].

[Ecm99] *„JSON“, Ecma International*. Veröffentlichung: 1999 [Online].
URL: http://json.org/json-de.html. [Abgerufen: 12. August 2015].

[Eva09] C. Evans, *„The Official YAML Web Site“,* letzte Änderung: 1. Oktober 2009. [Online].
URL: http://yaml.org/. [Abgerufen: 12. August 2015].

[Kru14] S. Krug, „Don’t Make Me Think!“, 3. Ausgabe, Mönchengladbach: mitp, 1. Aufl. 2014

[Lob14] P. Lobacher, „Certified TYPO3 CMS Integrator “, München: Open Source Press GmbH, 3. Aufl. 2014

[Lob14a] P. Lobacher, „TYPO3 Extbase“, München: Open Source Press GmbH, 1. Aufl. 2014

[Mac14] B. Mack, *„Insights from the TYPO3 CMS Active Contributor Meetup in Munich“*, TYPO3, Veröffentlichung: 16. September 2014. [Online].
URL: http://typo3.org/news/article/insights-from-the-typo3-cms-active-contributor-meetup-in-munich/. [Abgerufen: 08. Juni 2015].

[Mac14a] B. Mack, *„Embrace and Innovate - TYPO3 CMS 7“*, TYPO3, Veröffentlichung: 02. November 2014. [Online].
URL: http://typo3.org/news/article/embrace-and-innovate-typo3-cms-7/. [Abgerufen: 08. Juni 2015].

[Mac15] B. Mack, *„Announcing TYPO3 7.4 – Nothing is impossible“*, TYPO3, Veröffentlichung: 04. August 2015. [Online].
URL: http://typo3.org/news/article/announcing-typo3-74-nothing-is-impossible/. [Abgerufen: 25. August 2015].

[New12] J. Newmann, *„How Mobile Apps Are Changing Desktop Software | PCWorld“*, PCWorld, Veröffentlichung: 11. Juli 2012, 18:30 Uhr. [Online]. URL:
http://www.pcworld.com/article/259110/app_invasion_coming_soon_to_your_pc.html. [Abgerufen: 25. Mai 2015].

[Nor02] D. Norman, „*The Design of Everyday Things“* , 2002 [E-Book]

[Pry14] W. Pryjda, *„OS-Marktanteile: Windows 8.x im August mit überraschendem Plus - WinFuture.de“*, WinFuture, Veröffentlichung: 01. September 2014. [Online]. URL: http://winfuture.de/news,83407.html. [Abgerufen: 25. Juni 2015].

[Sco09] B. Scott und T. Neil, *„Designing Web Interfaces“*, Sebastopol, CA: O‘Reilly, 2009

[She15] K. Sherwin, *„Kontrastarme Texte sind keine Lösung : Institut für Software-Ergonomie und Usability“*, Institut für Software-Ergonomie und Usability AG, Veröffentlichung: 07. Juni 2015. [Online]. URL: http://www.usability.ch/news/schwacher-kontrast.html. [Abgerufen: 17. Juni 2015].

[Sie14] D. Siepmann, *„TYPO3 - CMS“*, Bitbucket, Veröffentlichung: 07. Dezember 2014. [Online]. URL: https://bitbucket.org/DanielSiepmann/typo3-cms/src. [Abgerufen: 18. Juni 2015].

[Typ13] TYPO3 Association, *„Wiki – TYPO3 6.2 Projects (+) – TYPO3 Forge”*, TYPO3, Veröffentlichung: 2013. [Online].
URL: https://forge.typo3.org/projects/typo3cms-v62/wiki. [Abgerufen: 06. August 2015].

[Typ13a] TYPO3 Association, *„Long Term Support – TYPO3 6.2 Projects (+) – TYPO3 Forge”,* TYPO3, Veröffentlichung: 2013. [Online].
URL: https://forge.typo3.org/projects/typo3cms-v62/wiki/Long_Term_Support. [Abgerufen: 06. August 2015].

[Typ14] TYPO3 Association, *„De:Fluid – TYPO3Wiki”,* TYPO3, letzte Aktualisierung: 28. Mai 2014. [Online]. URL: https://wiki.typo3.org/De:Fluid. [Abgerufen: 21. August 2015].

[Typ15] TYPO3 Association, *„Configuring The Plugin”*, TYPO3, letzte Aktualisierung: 26. August 2015. [Online]. URL: https://docs.typo3.org/typo3cms/ExtbaseFluidBook/latest/4-FirstExtension/7-configuring-the-plugin.html. [Abgerufen: 28. August 2015].

[Typ15a] TYPO3 Association, *„Coding Guidelines”*, TYPO3, letzte Aktualisierung: 26. August 2015. [Online]. URL: https://docs.typo3.org/typo3cms/ExtbaseFluidBook/a-CodingGuidelines/Index.html#file-and-class-names. [Abgerufen: 28. August 2015].

[Typ15b] TYPO3 Association, *„What is $TCA”*, TYPO3, letzte Aktualisierung: 26. August 2015. [Online]. URL: https://docs.typo3.org/typo3cms/TCAReference/WhatIsTca/Index.html. [Abgerufen: 28. August 2015].

[Typ15c] TYPO3 Association, *„Backend User Object”*, TYPO3, letzte Aktualisierung: 26. August 2015. [Online]. URL:
https://docs.typo3.org/typo3cms/CoreApiReference/ApiOverview/BackendUserObject/Index.html. [Abgerufen: 28. August 2015].

[Typ15d] TYPO3 Association, *„Individual Database Query”*, TYPO3, letzte Aktualisierung: 26. August 2015. [Online]. URL: https://docs.typo3.org/typo3cms/ExtbaseFluidBook/6-Persistence/3-implement-individual-database-queries.html. [Abgerufen: 28. August 2015].

[Veu15] A. Veuve, *„Agency Meetup Days 2014“*, TYPO3, Veröffentlichung: 28. Februar 2015. PDF [Online]. URL: https://typo3.org/fileadmin/t3org/images/FM-news/2015/Marketing/AgencyMeetupSummary_2014_final.pdf. [Abgerufen: 23. Juni 2015].

[Whi14] K. Whitenton, *„Filter vs. Facettennavigation: Definitionen - Institut für Software-Ergonomie und Usability“*, Institut für Software-Ergonomie und Usability AG, Veröffentlichung: 16. März 2014. [Online]. URL: http://www.usability.ch/news/filter-vs-facettennavigation-definitionen.html. [Abgerufen: 18. August 2015].

[Wik13] *„Backport - Wikipedia”*, Wikipedia, letzte Aktualisierung: 30. März 2013. [Online]. URL: https://de.wikipedia.org/wiki/Backport. [Abgerufen: 07. August 2015].

[Wik13a] *„Selektivität (Informatik) - Wikipedia”*, Wikipedia, letzte Aktualisierung: 09. Februar 2013. [Online]. URL: https://de.wikipedia.org/wiki/Selektivit%C3%A4t_(Informatik). [Abgerufen: 14. August 2015].

[Wik15a] *„TYPO3 - Wikipedia”*, Wikipedia, letzte Aktualisierung: 07. August 2015. [Online]. URL: https://de.wikipedia.org/wiki/TYPO3#TypoScript. [Abgerufen: 11. August 2015].

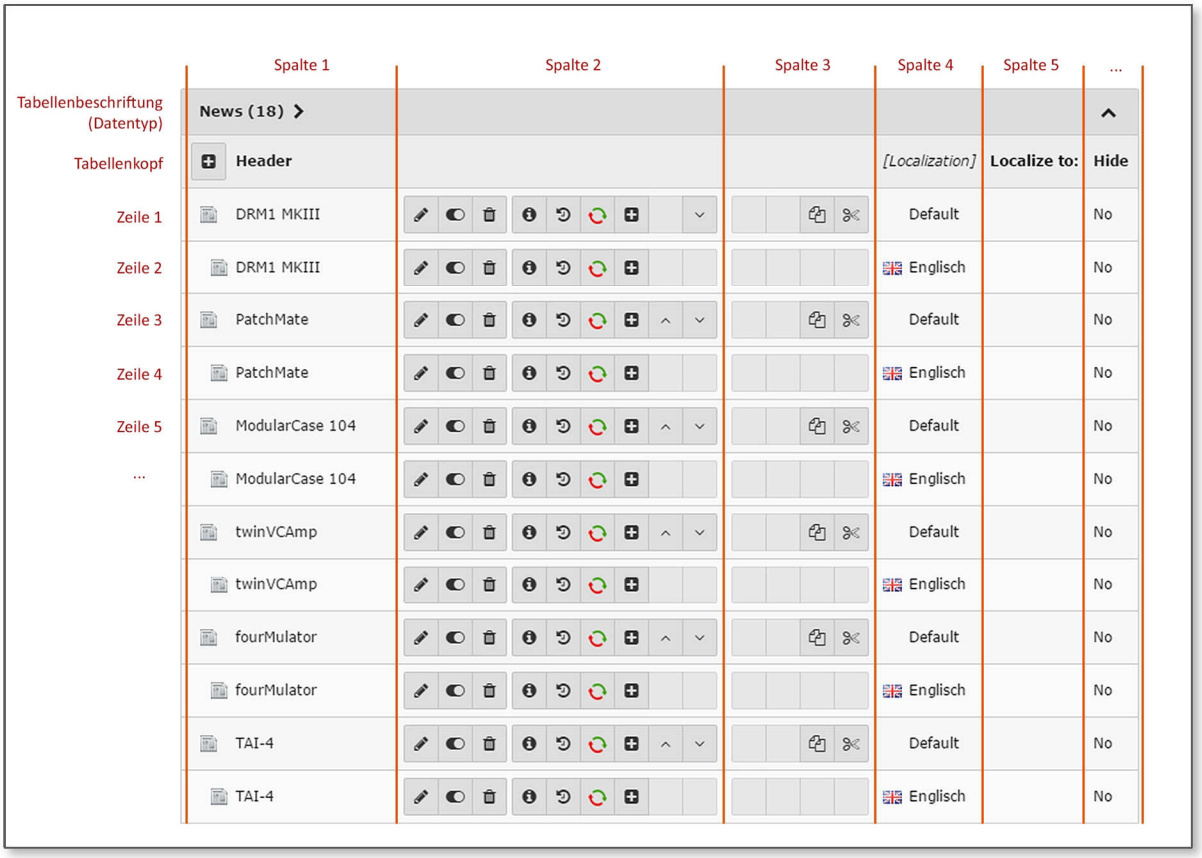
[Wik15b] *„TYPO3 - Wikipedia"*, Wikipedia, letzte Aktualisierung: 05. August 2015. [Online]. URL: https://de.wikipedia.org/wiki/Datensatz. [Abgerufen: 14. August 2015].

[Wik15c] *„Representational State Transfer - Wikipedia"*, Wikipedia, letzte Aktualisierung: 13. Juli 2015. [Online]. URL: https://de.wikipedia.org/wiki/Representational_State_Transfer. [Abgerufen: 09. September 2015].

9 ANHANG

Anhang 1	Screenshot zur Beschreibung von Zeilen und Spalten im TYPO3 CMS
Anhang 2	Auswertungstabellen der Konkurrenzanalyse
Anhang 3	ERM des Anwendungsbeispiels: Veranstaltungsmanagement
Anhang 4	ERM des Anwendungsbeispiels: Dateiverwaltung
Anhang 5	Tabelle der Basistypen intelligenter Filter
Anhang 6	JSON-String der persistierten Preset-Einstellungen am Beispiel des Veranstaltungsmanagements
Anhang 7	Preset-Konfiguration einer TYPO3 Backend App für globale Vorlagen am Beispiel des Veranstaltungsmanagements
Anhang 8	User Stories
Anhang 9	Profile der Testpersonen
Anhang 10	Bewertungstabellen des A/B-Testverfahrens
Anhang 11	Zeitauswertungen des A/B-Testverfahrens
Anhang 12	Abschlussbewertung des A/B-Testverfahrens
Anhang 13	Link-Verzeichnis der Dateien

Anhang 1 Screenshot zur Beschreibung von Zeilen und Spalten im TYPO3 CMS



Autor: A. Steiger, 2015

Anhang 2 Auswertungstabellen der Konkurrenzanalyse

Kriterium		Verfügbarkeit und Beschreibung		Verfügbarkeit und Beschreibung	
		Wordpress (4.2.2)		Drupal (7.35)	Excel (Microsoft Office 2010)
1.	**Aufbau des Layouts**	○	Vertikal von oben nach unten aufgebaut; wenig seitliche Elemente	○ Vertikal von oben nach unten aufgebaut; wenig seitliche Elemente	○ Vertikal von oben nach unten aufgebaut
2.	**Erweiterte Ansichten**	✓	Schmale Listen; hohe Listen; Kacheln	✓ Weitere Spaltenanzeige möglich; Kachelansicht	✓ Diagramme und Grafiken; verschiedene Bearbeitungs-/Ansichtsmodi
3.	**Navigation/Orientierung**	✗	Eine Ebene mit allen Elementen; Kann gefiltert werden nach Voreinstellungen; gesetzte Filter nicht immer erkennbar	✓ Breadcrumb oben links; über Reiter oben; eine Ebene mit allen Elementen; Kann gefiltert werden; kein Seitenbaum	✓ Über Reiter unten
4.	**Speicherung**	✓	Über „Update"-Button; nicht konsistent positioniert	✓ Live-Änderungen werden erkannt, jedoch nicht automatisch gespeichert; Hinweis auf manuelles Speichern über Button unten links	✓ Über "Datei" oder Disketten-Icon oben links
5.	**Tabellenmanipulationen**				
5.1	**Management-Ebene**				
5.1.1	***Sortierung***	✗	Über Kicken des Spaltenkopfes; eingeschränkt möglich	✗ Über Kicken des Spaltenkopfes; eingeschränkt möglich	✓ Über Schaltfläche "Sort and Filter"
5.1.2	***Filterung***	✗	Über vordefinierte, intelligente Filterbereiche oder Link in Liste; gesetzte Filter nicht immer erkennbar; keine eigenen Filter definierbar	✗ Einfacher Filter; vordefinierte Spalten nach einzelnen Begriffen filterbar; keine eigenen Filter definierbar	✓ Über Schaltfläche "Sort and Filter"
5.1.3	***Größenanpassung***	✗		✗	✓ Rechts und links am Rand des Tabellenkopfes per Drag-and-Drop
5.1.4	***Verschiebung von Spalten***	✗		✗	✓ Flexibel per Drag-and-Drop
5.1.5	***Ein- und Ausblendung von Spalten***	✓	Über extra Menü; wenig verfügbare Spalten	✗	✓ Über Kontextmenü
5.2	**Persistenz-Ebene**				
5.2.1	***Hinzufügung von Zeilen***	✓	Über „Add New"-Button oben links	✓ Über „Add content"-Button links oben	✓ Über Kontextmenü
5.2.2	*Hinzufügung von Spalten*	✗		✗	✓ Über Kontextmenü
5.2.3	***Ein- und Ausblendung von Zeilen***	✓	Über In-Page-Editing oder extra Formular	✓ Über „edit"-Button und extra Formularseite	✓ Über Doppelklick
5.2.4	***Bearbeitung***	✓	Nach Auswahl über "Bulk Actions"-Dropdown	✓ Nach Auswahl über "Update Options"-Dropdown	✓ Über Kontextmenü
5.2.5	***Entfernung von Zeilen***				
5.2.6	*Entfernung von Spalten*				

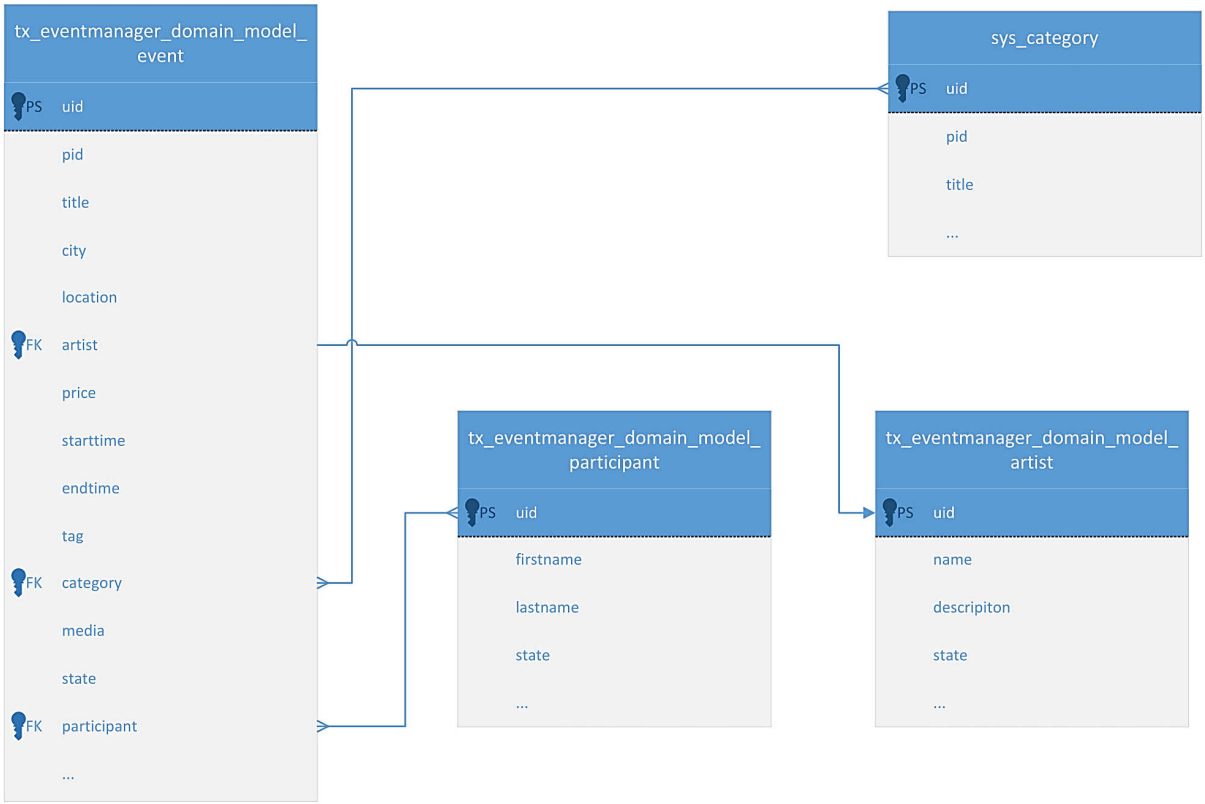
Legende: ✓ = ja; ✗ = nein; ○ = teilweise

Autor: A. Steiger, 2015

	Kriterium	Verfügbarkeit und Beschreibung	Verfügbarkeit und Beschreibung	Verfügbarkeit und Beschreibung
		Tabellen (Google Docs)	**Explorer (Windows 7)**	**Finder (Mac OSX)**
1.	**Aufbau des Layouts**	Vertikal von oben nach unten aufgebaut	Vertikal und Horizontal aufgebaut	Vertikal und Horizontal aufgebaut
2.	**Erweiterte Ansichten**	Diagramme und Grafiken; verschiedene Bearbeitungs-/Ansichtsmodi	Umschaltung der Ansichten rechts oben; erweiterte Ansichten können über die Ordneroptionen definiert werden	Umschaltung der Ansichten im oberen Bereich
3.	**Navigation/Orientierung**	Über Reiter unten	Über Breadcrumb oben; über Seitenbaum links;	Über Breadcrumb oben
4.	**Speicherung**	Automatische Speicherung	Automatische Speicherung	Automatische Speicherung
5.	**Tabellenmanipulationen**			
5.1	**Management-Ebene**			
5.1.1	***Sortierung***	Über Menüpunkt "Daten"	Über Klicken des Spaltenkopfes	Über Klicken des Spaltenkopfes
5.1.2	***Filterung***	Über Schaltfläche "Filter" oder Menüpunkt "Daten"	Über voreingestellte Filterbereiche (intelligent) oder Texteingabe im Schnellfitlerfeld	Eigene Filter speichern
5.1.3	***Größenanpassung***	Rechts und links am Rand des Tabellenkopfes per Drag-and-Drop	Rechts und links am Rand des Tabellenkopfes per Drag-and-Drop	Rechts und links am Rand des Tabellenkopfes per Drag-and-Drop
5.1.4	***Verschiebung von Spalten***	Flexibel per Drag-and-Drop	Flexibel per Drag-and-Drop	Flexibel per Drag-and-Drop
5.1.5	***Ein- und Ausblendung von Spalten***	Über Kontextmenü	Über Kontextmenü am Spaltenkopf; große Auswahl	Über Kontextmenü am Spaltenkopf
5.2	**Persistenz-Ebene**			
5.2.1	***Hinzufügung von Zeilen***	Über Kontextmenü	Über Kontextmenü	Nur über externe Programme
5.2.2	*Hinzufügung von Spalten*			
5.2.3	***Ein- und Ausblendung von Zeilen***	Über Kontextmenü	Über Anzeigeeinstellungen; nur begrenzt möglich	Über Terminal; nur begrenzt möglich
5.2.4	***Bearbeitung***	Über Doppelklick	Nur Dateiname/Ordnername	Nur Dateiname/Ordnername
5.2.5	***Entfernung von Zeilen***	Über Kontextmenü	Über Kontextmenü oder "Entf"-Taste	Über Kontextmenü, Drag-and-Drop oder die Tasten "Cmd" + "Return"
5.2.6	*Entfernung von Spalten*			
	Legende: = ja; = nein; = teilweise			

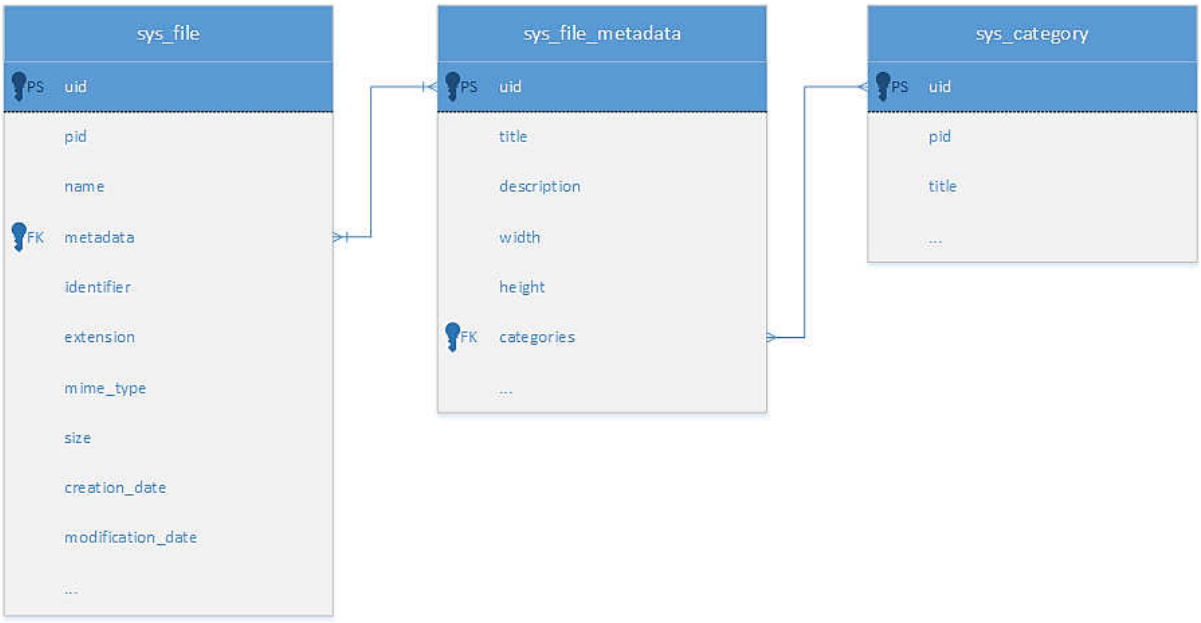
Autor: A. Steiger, 2015

Anhang 3 ERM des Anwendungsbeispiels: Veranstaltungsmanagement



Autor: A. Steiger, 2015

Anhang 4 ERM des Anwendungsbeispiels: Dateiverwaltung



Autor: A. Steiger, 2015

Anhang 5 Tabelle der Basistypen intelligenter Filter

EBENE 1	EBENE 2		EBENE 3		EBENE 4		
Typ	Label	Typ	Inhalte	Typ	Inhalte	Label	Typ	Inhalte
Einzeiliger Text (String)	-	Auswahl	*enthält*	Eingabe	-	-	-	-
			ist gleich	Eingabe	-	-	-	-
			beginnt mit	Eingabe	-	-	-	-
			endet mit	Eingabe	-	-	-	-
Mehrzeiliger Text / Inhalt	enthält Text	Eingabe	-	-	-	-	-	-
Boolescher Wert	-	Auswahl	*Ja*	-	-	-	-	-
			Nein	-	-	-	-	-
Numerischer Wert	ist	Auswahl	*gleich*	Eingabe	-	-	(Auswahl)	*(vorgegebene Einheiten)*
			nicht gleich	Eingabe	-	-	(Auswahl)	*(vorgegebene Einheiten)*
			größer als	Eingabe	-	-	(Auswahl)	*(vorgegebene Einheiten)*
			kleiner als	Eingabe	-	-	(Auswahl)	*(vorgegebene Einheiten)*
			innerhalb	Eingabe	-	bis	Eingabe	-
Datum / Zeit	ist	Auswahl	*innerhalb der letzten*	Eingabe	-	-	Auswahl	*Stunde(n)* *Tag(e)* *Woche(n)* *Monat(e)* *Jahr(e)*
			innerhalb der nächsten	Eingabe	-	-	-	-
			letzte(r)	Auswahl	*Stunde* *Tag* *Woche* *Monat* *Jahr*	-	-	-
			diese(r)	Auswahl	*Stunde* *Tag* *Woche* *Monat* *Jahr*	-	-	-
			vor	Eingabe	-	-	-	-
			nach	Eingabe	-	-	-	-
			innerhalb	Eingabe	-	bis	Eingabe	-
Auswahl	ist	Auswahl mit mehrfacher Selektion	*vorgegebene Werte*	-	-	-	-	-

Autor: A. Steiger, 2015

Anhang 6 JSON-String der persistierten Preset-Einstellungen am Beispiel des Veranstaltungsmanagements

Autor: A. Steiger, 2015

Anhang 7 Preset-Konfiguration einer TYPO3 Backend App für globale Vorlagen am Beispiel des Veranstaltungsmanagements

```
buttons:
  specialGlobalPreset:
    label: 'EXT:event_manager/Resources/Private/Language/locallang_apps_eventapp.xlf:presets.categories.s
    filter:
      -
        criteria:
          -
            column: tx_eventmanager_domain_model_event.crdate
            constraints:
              - 'within last'
              - 2
              - 'weeks'
      -
        criteria:
          -
            column: tx_eventmanager_domain_model_event.eventstart
            constraints:
              - 'this'
              - 'month'
      -
        criteria:
          -
            column: tx_eventmanager_domain_model_event.title
            constraints:
              - 'contains'
              - 'dance'
          -
            column: tx_eventmanager_domain_model_event.location
            constraints:
              - 'contains'
              - 'dance'
      -
        criteria:
          -
            column: tx_eventmanager_domain_model_event.pid
            constraints:
              - 'equals'
              - 5
    view:
      list:
        columns:
          -
            column: tx_eventmanager_domain_model_event.title
            width: 395px
            sorting: true
          -
            column: tx_eventmanager_domain_model_event.categories
            width: 200px
          -
            column: tx_eventmanager_domain_model_event.city
            width: 150px
```

I II III IV V

Autor: A. Steiger, 2015

Anhang 8 User Stories

US1: Als Redakteur möchte ich im TYPO3 Backend eine bestimmte Veranstaltung finden, um diese zu bearbeiten.

Beispiel: Als Redakteur möchte ich im TYPO3 Backend die Veranstaltung Faust in Berlin finden, um diese zu bearbeiten.

US2: Als Redakteur möchte ich im TYPO3 Backend alle Veranstaltungen einer bestimmten Seite sehen, um eine Übersicht zu erhalten.

Beispiel: Als Redakteur möchte ich im TYPO3 Backend alle Veranstaltungen der Seite „Website one/What's up?/Events" sehen, um eine Übersicht zu erhalten.

US3: Als Redakteur möchte ich im TYPO3 Backend eine Tabelle nach einer Spalte sortieren, um Elemente besser zu finden.

US4: Als Redakteur möchte ich im TYPO3 Backend eine sortierte Tabellenansicht speichern, um später diese erneut anzeigen zu lassen.

US5: Als Redakteur möchte ich im TYPO3 Backend die gespeicherten Ansichten aufrufen.

US6: Als Redakteur möchte ich im TYPO3 Backend den angezeigten Veranstaltungen zusätzliche Informationen entnehmen, um diese weiter zu verarbeiten.

Beispiel: Als Redakteur möchte ich im TYPO3 Backend die Dauer (in Tagen) der angezeigten Veranstaltungen herausfinden, um diese weiter zu verarbeiten.

US7: Als Redakteur möchte ich im TYPO3 Backend eine neue Veranstaltung anlegen.

US8: Als Redakteur möchte ich im TYPO3 Backend Veranstaltungen in verschiedenen Ansichten verwalten, damit ich für unterschiedliche Aufgaben die passende Visualisierung zur Verfügung habe.

US9: Als Redakteur möchte ich im TYPO3 Backend Einstellungen zentral anpassen.

Als Redakteur möchte ich im TYPO3 Backend die Anzahl der Veranstaltungen je Paginierung anpassen.

US10: Als Redakteur möchte ich im TYPO3 Backend eine Veranstaltungsliste filtern, um ausschließlich die für mich relevanten Einträge zu sehen.

Beispiel: Als Redakteur möchte ich im TYPO3 Backend eine Veranstaltungsliste filtern, um ausschließlich die Klassik-Veranstaltungen zu sehen, die innerhalb der nächsten drei Monate beginnen.

Anhang 9 Profile der Testpersonen

Testperson 1

Beruf:	Frontend-Entwickler, Web-Designer
Computererfahrung:	tägliche Arbeit mit CMS; tägliche Web-Programmierung
TYPO3-Erfahrung:	häufige Redakteursarbeit; geschult

Testperson 2

Beruf:	Geschäftsführer, Web-Designer
Computererfahrung:	tägliche Arbeit mit CMS; Design mit Photoshop; Office-Programme
TYPO3-Erfahrung:	tägliche Redakteursarbeit; geschult

Testperson 3

Beruf:	Auszubildende für Büromanagement
Computererfahrung:	Arbeit mit Photoshop und Gimp; Office-Programme; Internet
TYPO3-Erfahrung:	keine

Testperson 4

Beruf:	Fachberater und Dozent im Gesundheitswesen
Computererfahrung:	tägliche Arbeit am Computer; Office-Programme; Internet
TYPO3-Erfahrung:	seltene Redakteursarbeit; selbst angeeignet

Anhang 10 Bewertungstabellen des A/B-Testverfahrens

Testperson 1													
User Stories	a) Aufgabe	Note	b) Benötigte Zeit in Minuten	Note	c) Anzahl der Fehlanläufe zum Lösungsweg	Note	d) Hilfestellung durch Test-leiter notwendig	Note	e) Verständnisprobleme	Note	∅*
Testversion A											
US1	erledigt	1	01:53	3	1	2	nein	1	nein	1	1,5
US2	erledigt	1	01:12	3	1	2	ja	6	nein	1	2,3
US3	abgebrochen	6	03:00	6	5	6	Ja	6	nein	1	5,2
US4	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US5	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US6	erledigt	1	00:51	2	0	1	nein	1	nein	1	1,2
US7	erledigt	1	00:04	1	0	1	nein	1	nein	1	1,0
									gemittelte Bewertungen:		**3,3**
Testversion B											
US1	erledigt	1	00:34	2	1	2	nein	1	nein	1	1,3
US2	erledigt	1	00:39	2	1	2	nein	1	ja	6	2,2
US3	erledigt	1	00:05	1	0	1	nein	1	nein	1	1,0
US4	erledigt	1	00:18	1	0	1	nein	1	nein	1	1,0
US5	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US6	erledigt	1	00:10	1	0	1	nein	1	nein	1	1,0
US7	erledigt	1	00:03	1	0	1	nein	1	nein	1	1,0
US8	erledigt	1	00:04	1	0	1	nein	1	nein	1	1,0
US9	erledigt	1	00:13	1	1	2	nein	1	nein	1	1,2
US10	erledigt	1	00:37	2	0	1	nein	1	nein	1	1,2
									gemittelte Bewertungen:		**1,7**
* = der Durchschnitt wird hierbei mit einer doppelten Gewichtung von a) berechnet.											

Autor: A. Steiger, 2015

Testperson 2													
User Stories	a) Aufgabe	Note	b) Benötigte Zeit in Minuten	Note	c) Anzahl der Fehlanläufe zum Lösungsweg	Note	d) Hilfestellung durch Test-leiter notwendig	Note	e) Verständnisprobleme	Note	∅*
Testversion A											
US1	erledigt	1	02:10	4	5	6	ja	6	nein	1	3,2
US2	erledigt	1	01:07	3	2	3	ja	6	nein	1	2,5
US3	erledigt	1	00:18	1	1	2	nein	1	nein	1	1,2
US4	abgebrochen	6	03:00	6	5	6	ja	6	nein	1	5,2
US5	abgebrochen	6	03:00	6	5	6	ja	6	nein	1	5,2
US6	erledigt	1	00:20	1	0	1	nein	1	nein	1	1,0
US7	erledigt	1	00:06	1	0	1	nein	1	nein	1	1,0
									gemittelte Bewertungen:		**2,7**
Testversion B											
US1	erledigt	1	00:29	1	0	1	nein	1	nein	1	1,0
US2	erledigt	1	00:56	2	2	3	ja	6	nein	1	2,3
US3	erledigt	1	00:04	1	0	1	nein	1	nein	1	1,0
US4	erledigt	1	00:14	1	0	1	nein	1	nein	1	1,0
US5	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US6	erledigt	1	00:29	1	0	1	ja	6	ja	6	2,7
US7	erledigt	1	00:06	1	0	1	nein	1	nein	1	1,0
US8	erledigt	1	00:05	1	0	1	nein	1	nein	1	1,0
US9	erledigt	1	00:19	1	1	2	nein	1	nein	1	1,2
US10	erledigt	1	00:52	2	0	1	nein	1	nein	1	1,2
									gemittelte Bewertungen:		**1,8**
* = der Durchschnitt wird hierbei mit einer doppelten Gewichtung von a) berechnet.											

Autor: A. Steiger, 2015

Testperson 3												
User Stories	a) Aufgabe	Note	b) Benötigte Zeit in Minuten	Note	c) Anzahl der Fehlanläufe zum Lösungsweg	Note	d) Hilfestellung durch Test-leiter notwendig	Note	e) Verständnisprobleme	Note	∅*
Testversion A											
US1	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US2	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US3	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US4	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US5	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US6	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US7	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
									gemittelte Bewertungen:		**6,0**
Testversion B											
US1	erledigt	1	00:25	1	0	1	nein	1	nein	1	1,0
US2	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US3	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US4	erledigt	1	00:21	1	0	1	nein	1	nein	1	1,0
US5	erledigt	1	00:52	2	2	3	nein	1	nein	1	1,5
US6	erledigt	1	00:42	2	0	1	ja	6	ja	6	2,8
US7	erledigt	1	00:03	1	0	1	nein	1	nein	1	1,0
US8	erledigt	1	00:06	1	0	1	nein	1	nein	1	1,0
US9	erledigt	1	00:31	2	2	3	nein	1	nein	1	1,5
US10	erledigt	1	01:01	3	0	1	nein	1	nein	1	1,3
									gemittelte Bewertungen:		**2,3**

* = der Durchschnitt wird hierbei mit einer doppelten Gewichtung von a) berechnet.

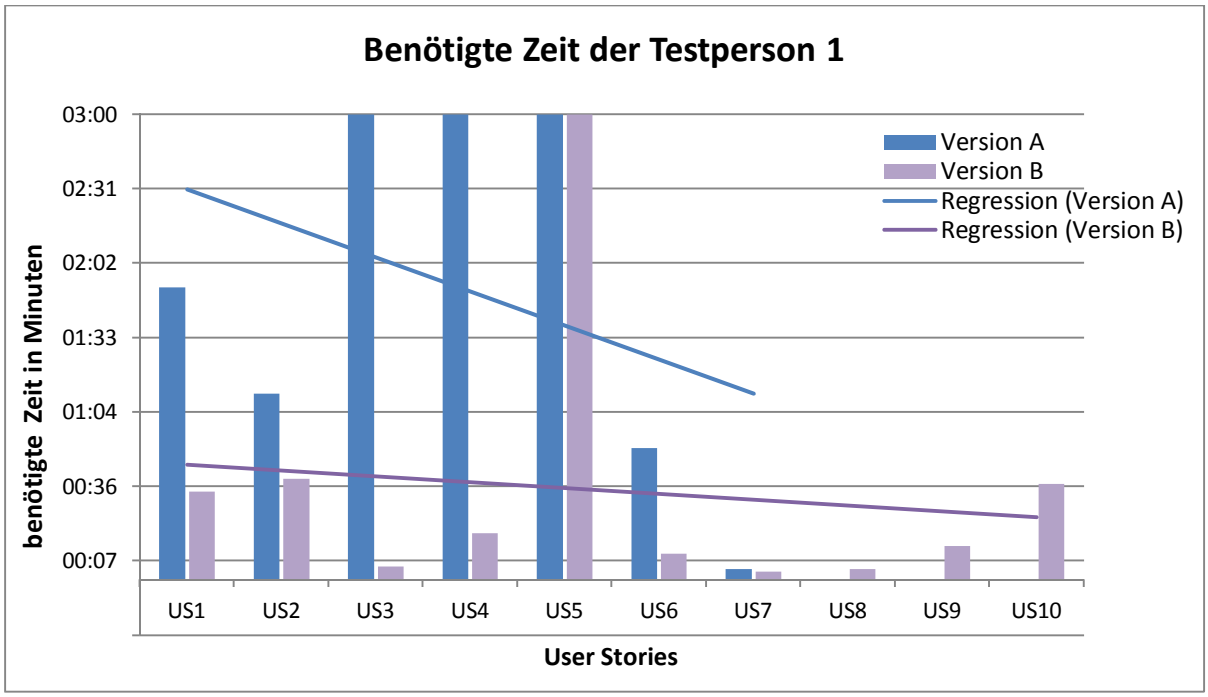
Autor: A. Steiger, 2015

Testperson 4												
User Stories	a) Aufgabe	Note	b) Benötigte Zeit in Minuten	Note	c) Anzahl der Fehlanläufe zum Lösungsweg	Note	d) Hilfestellung durch Test-leiter notwendig	Note	e) Verständnisprobleme	Note	∅*
Testversion A											
US1	abgebrochen	6	03:00	6	5	6	ja	6	nein	1	5,2
US2	erledigt	1	00:33	2	0	1	nein	1	nein	1	1,2
US3	erledig	1	00:42	2	1	2	nein	1	ja	6	2,2
US4	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US5	abgebrochen	6	03:00	6	5	6	ja	6	ja	6	6,0
US6	erledigt	1	00:22	1	1	2	nein	1	nein	1	1,2
US7	erledigt	1	00:05	1	0	1	nein	1	nein	1	1,0
									gemittelte Bewertungen:		**3,2**
Testversion B											
US1	erledigt	1	00:26	1	0	1	nein	1	nein	1	1,0
US2	erledigt	1	01:03	3	1	2	ja	6	ja	6	3,2
US3	erledigt	1	00:08	1	0	1	nein	1	nein	1	1,0
US4	erledigt	1	00:31	2	1	2	ja	6	nein	1	2,2
US5	erledigt	1	00:51	2	1	2	ja	6	ja	6	3,0
US6	erledigt	1	00:18	1	0	1	nein	1	nein	1	1,0
US7	erledigt	1	00:03	1	0	1	nein	1	nein	1	1,0
US8	erledigt	1	00:03	1	0	1	nein	1	nein	1	1,0
US9	erledigt	1	01:51	4	5	6	ja	6	nein	1	3,2
US10	erledigt	1	01:03	3	0	1	nein	1	nein	1	1,3
									gemittelte Bewertungen:		**1,8**

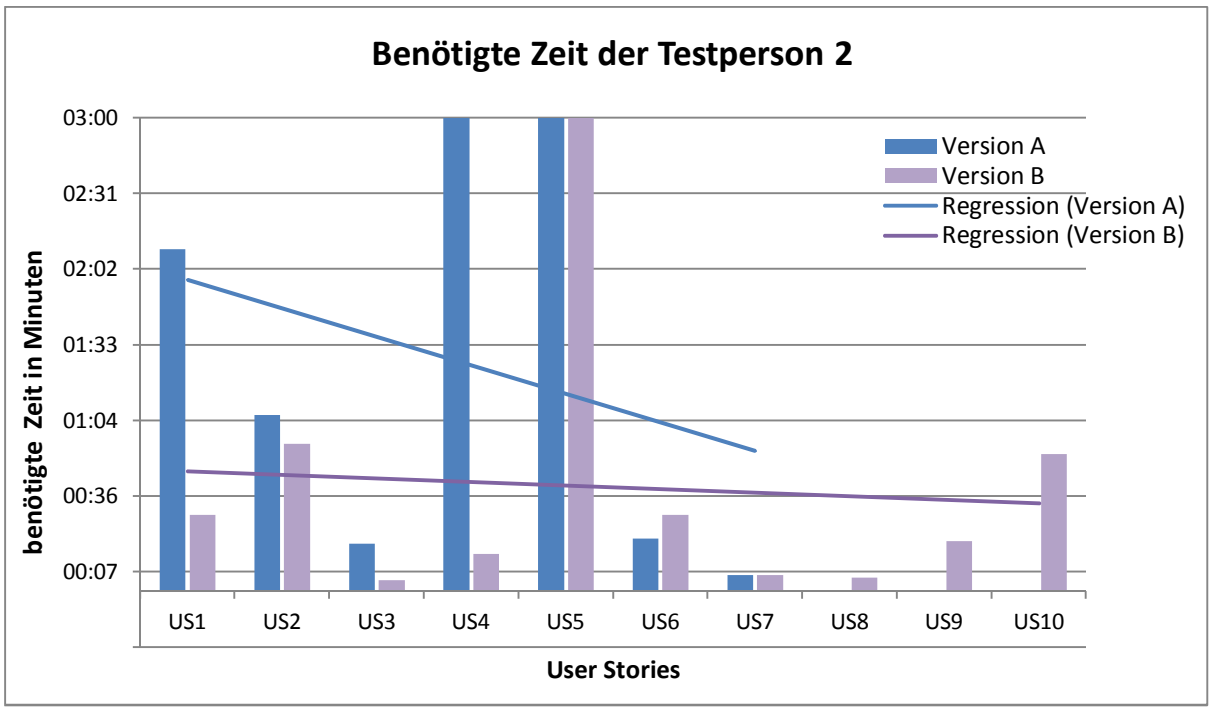
* = der Durchschnitt wird hierbei mit einer doppelten Gewichtung von a) berechnet.

Autor: A. Steiger, 2015

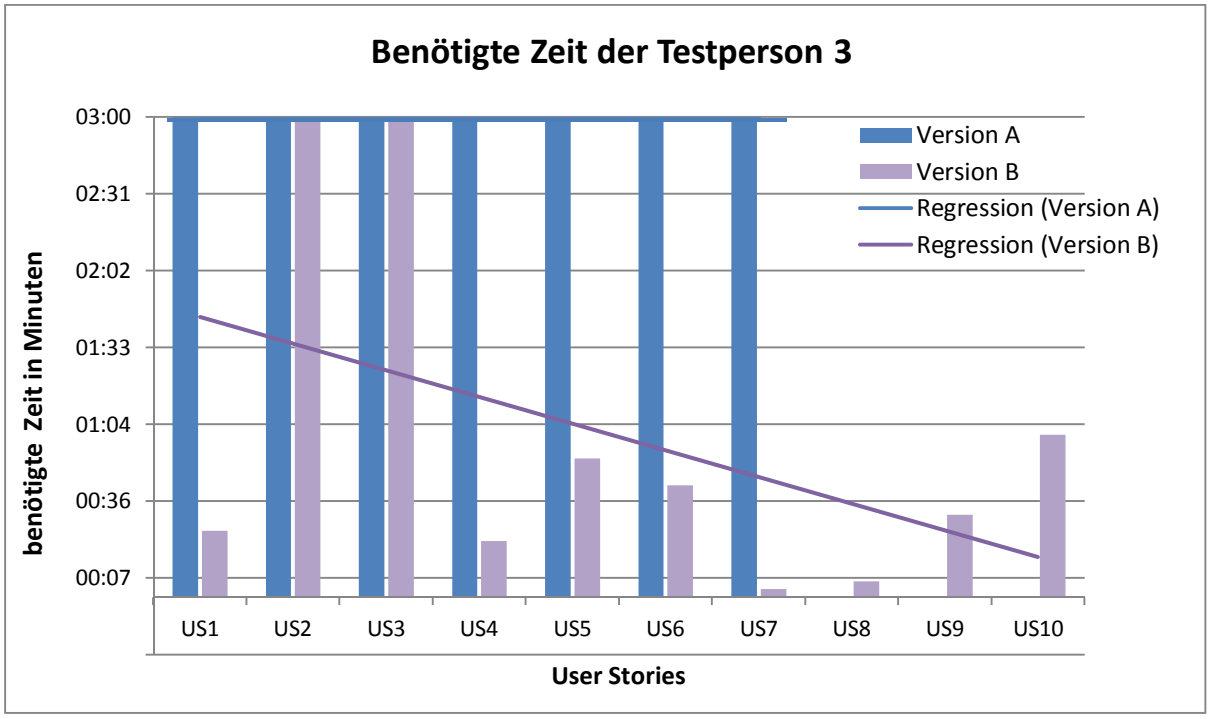
Anhang 11 Zeitauswertungen des A/B-Testverfahrens



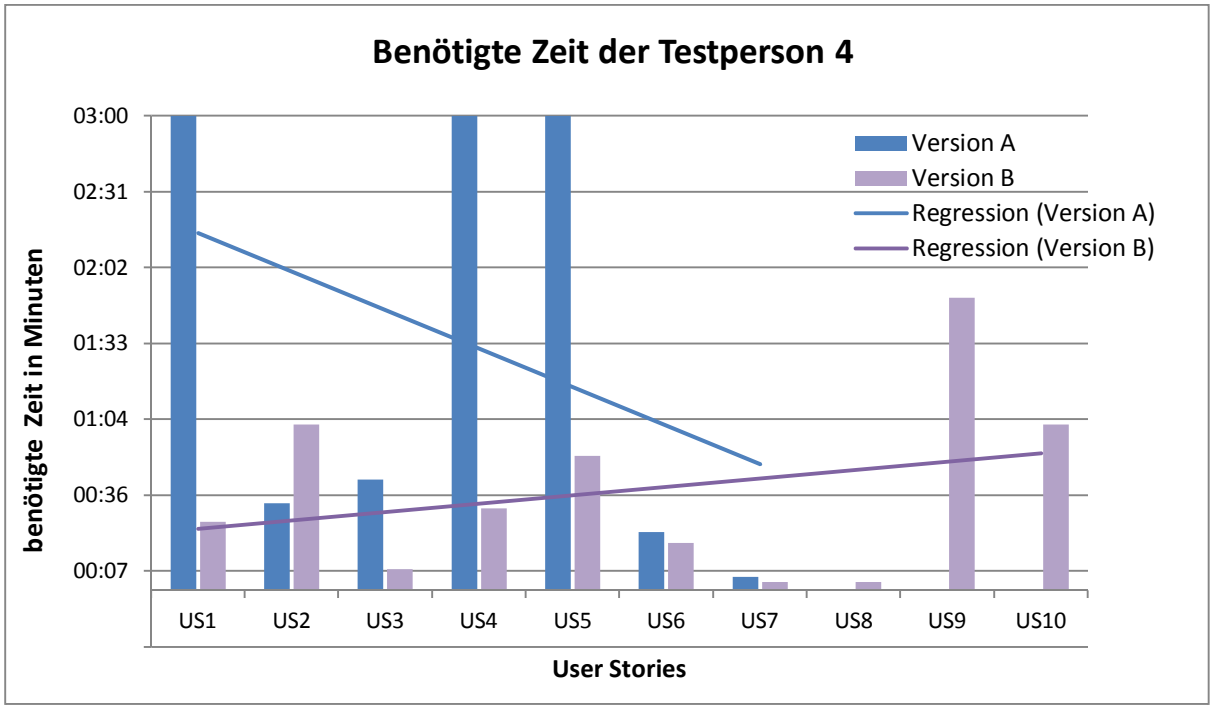
Autor: A. Steiger, 2015



Autor: A. Steiger, 2015

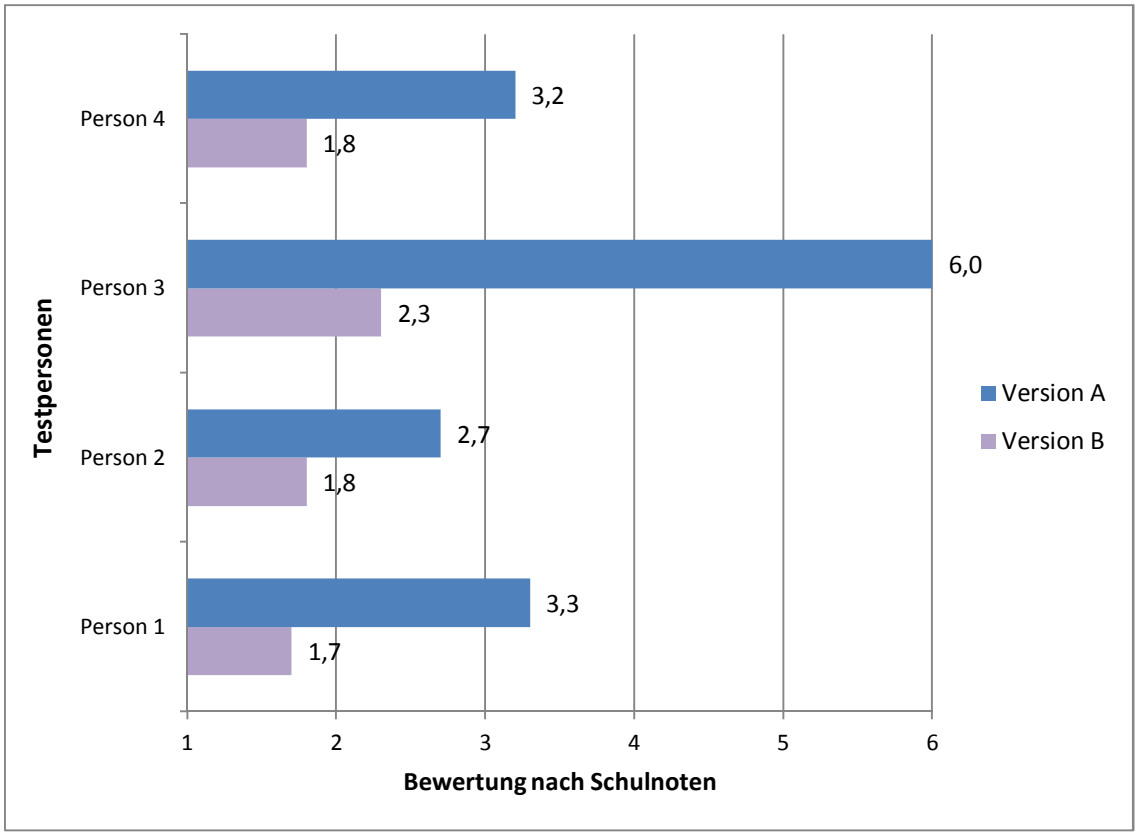


Autor: A. Steiger, 2015



Autor: A. Steiger, 2015

Anhang 12 Abschlussbewertung des A/B-Testverfahrens



Autor: A. Steiger, 2015

Anhang 13 Link-Verzeichnis der Dateien

1 Präsentation der User Experience Week 2014 zum Thema: Focus Apps:
http://t3app.steiger.webseiten.cc/Backend_Module_Toolbox.pdf

2 Prototyp: http://t3app.steiger.webseiten.cc/start_app.html

3 YAML-Datei der List App als Basis-Konfiguration:
http://t3app.steiger.webseiten.cc/ListApp.yaml

4 YAML-Datei der Event-App-Konfiguration:
http://t3app.steiger.webseiten.cc/EventApp.yaml

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde nach meiner besten Kenntnis bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Hof, 10.09.2015 Andreas Steiger