Eberhard-Karls-Universität Tübingen

Fakultät für Informations- und Kognitionswissenschaften
Wilhelm-Schickard-Institut für Informatik
Arbeitsbereich für Theoretische Informatik / Formale Sprachen

Integration von eKaay Sesame in TYPO3

Bachelorarbeit

im Fachgebiet Informatik

Jonas Reichert
16. Februar 2012

Betreuer:

Dr. Bernd Borchert
Prof. Dr. Klaus Reinhardt

Zusammenfassung

Die Einbindung des eKaay-Verfahrens – welches eine Alternative zum herkömmlichen Login-Verfahren darstellt – in ein bestehendes Softwaresystem wie das Content-Management-System TYPO3, stellt Anwender aufgrund dessen Komplexität vor einige Schwierigkeiten. Diese Arbeit analysiert die Mechanismen der Benutzerauthentifizierung von TYPO3, untersucht die zur Erstellung einer Erweiterung nötigen Schritte und stellt eine Implementierung vor, die das eKaay-Verfahren einfach und unkompliziert in TYPO3 integriert.

Des Weiteren werden in Kapitel 5 die geltenden rechtlichen Rahmenbedingungen, die beim Upload einer Extension in das TYPO3 Extension Repository bestehen, dargelegt.

Eidesstattliche Erklärung

Ich, Jonas Reichert, versichere hiermit, dass ich meine Bachelorarbeit mit dem Thema

Integration von eKaay Sesame in TYPO3

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, wobei ich alle wörtlichen und sinngemäßen Zitate als solche gekennzeichnet habe. Die Arbeit wurde bisher keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

Tübingen, den 16. Februar 2012

Jonas Reichert

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Den Anstoß zur Ausschreibung dieses Themas gab die von den für eKaay verantwortlichen Personen beschlossene Strategie, die eKaay-Implementierung zunächst kostenlos und möglichst weit zu verbreiten. Obwohl sich der Aufwand für die Installation des eKaay-Verfahrens und die damit verbundene Integration in eine bereits bestehende Webapplikation im Laufe der Entwicklung immer weiter verringert hat, sind dazu immer noch tiefer gehende Kenntnisse über die Authentifizierungsmechanismen vonnöten. Auf der anderen Seite hat die fortschreitende Verbesserung der Benutzbarkeit und die u. a. daraus resultierende zunehmende Verbreitung moderner Content-Management-Systeme (CMS) dazu geführt, dass deren Betreiber – im weiteren Verlauf der Arbeit auch gleichbedeutend *Administratoren* genannt – nicht zwangsweise über solche Kenntnisse verfügen müssen.

Um die zwischen dem eKaay-Verfahren und dessen problemloser Einbindung in aktuelle CMS existierende Lücke zu schließen, wurde dessen Integration in TYPO3 – welches u. a. zusammen mit Joomla, Drupal und Wordpress als eines der bekanntesten und am weitesten verbreiteten CMS gilt[W3T12] – in Form einer Erweiterung beschlossen.

1.2 Ziel der Arbeit

Das Ziel dieser Arbeit ist es, eine lauffähige Erweiterung für TYPO3 zu erstellen, mittels derer eine TYPO3-Installation „mit einem Klick“ um das eKaay-Verfahren ergänzt werden kann, so dass sich die TYPO3-Benutzer ab diesem Zeitpunkt mit ihrem Smartphone einloggen können. Da bereits die Installation einer TYPO3-Erweiterung – unabhängig von deren Komplexität – mehr als einen Klick benötigt, ist dieses Zitat nicht wortwörtlich zu verstehen, viel eher sollte die Einbindung der Erweiterung so einfach wie möglich und ohne unnötiges Eingreifen des Administrators in den Installationsprozess ablaufen.

Um einen problemlosen Updatemechanismus zu gewährleisten, soll die bestehende eKaay-Implementierung in ihrer ursprünglichen Form als Kern der Erweiterung

dienen, die TYPO3-Erweiterung selbst wirkt als Schnittstelle zwischen eKaay und der TYPO3-Installation. Somit gilt für die Erweiterung die Einschränkung, dass der Quelltext der eKaay-Implementation nicht verändert werden darf. Im weiteren Verlauf der Arbeit wird die bereits bestehende, nicht zu verändernde eKaay-Implementierung *eKaay-Core* genannt.

1.3 Aufbau der Arbeit

Zunächst wird in Abschnitt 1.2 der Einleitung das Ziel dieser Arbeit genau definiert. Des Weiteren vermitteln die Abschnitte 1.4.1 sowie 1.4.2 grundlegendes Wissen zu den, die fundamentale Basis für diese Arbeit bildenden Technologien TYPO3 und eKaay.

Kapitel 2 „Schreiben einer TYPO3-Extension“ beschreibt allgemein die notwendigen Schritte zum Erstellen einer TYPO3-Extension und analysiert deren Verzeichnis- und Dateistruktur. Zum Ende des Kapitels wird noch kurz auf die TYPO3-Extension *Kickstarter* eingegangen.

Im darauffolgenden Kapitel wird die TYPO3-eigene Benutzerauthentisierung betrachtet. Neben der Darstellung verschiedenster Authentifizierungsmechanismen wird das Konzept der Authentication Services vorgestellt und beschrieben.

Kapitel 5 beleuchtet anschließend das TYPO3 Extension Repository und beschreibt beispielhaft den Upload einer Extension. Zusätzlich zeigt Abschnitt 5.2 dieses Kapitels die zu beachtenden rechtlichen Grundlagen beim Veröffentlichen von TYPO3-Erweiterungen im TYPO3 Extension Repository.

Zuletzt werden in Kapitel 6 die zuvor formulierten Ziele mit dem erreichten Ergebnis verglichen und ein Fazit gezogen.

1.4 Voraussetzungen zum Verständnis der Arbeit

1.4.1 TYPO3

TYPO3 ist ein auf PHP und MySQL basierendes Open Source Content-Management-Framework. Es erfreut sich u. a. aufgrund seiner Zuverlässigkeit, der umfassenden Benutzerverwaltung, besonders aber aufgrund seiner Modularität vor allem in größeren Unternehmen großer Beliebtheit.[Die09] Das gesamte System ist nach dem „Baukastenprinzip“ aufgebaut und lässt sich auf diese Weise mit weiteren Modulen

beliebig erweitern. Diese Module werden im TYPO3-Jargon üblicherweise *Extensions* genannt. Im weiteren Verlauf dieser Arbeit werden nunmehr beide Begriffe gleichbedeutend verwendet.

Frontend und Backend Eine TYPO3-Installation ist in zwei Bereiche aufgeteilt, das *Frontend* und das *Backend*. Das Frontend ist die öffentlich zugängliche Webseite, auf der die im Backend aufbereiteten Inhalte präsentiert werden. Im Backend, welches über das Frontend im Regelfall nicht direkt erreichbar ist, lässt sich die Seite beliebig konfigurieren, es können neue Inhalte bereitgestellt und Bestehende verändert oder wieder gelöscht werden.

Die TYPO3-Benutzerverwaltung unterscheidet generell zwei verschiedene Arten von Benutzern: Frontend- und Backend-Benutzer. Backend-Benutzer können sich ausschließlich beim Backend anmelden und haben – abhängig von der ihrem Benutzer zugeordneten Benutzergruppe – spezielle, meist eingeschränkte Berechtigungen. Im Gegensatz hierzu ist der Login im Backend für Frontend-Benutzer nicht erlaubt, diese dürfen sich nur auf der öffentlich zugänglichen Webseite anmelden.

Frontend-Plugins und Backend-Module Frontend-Plugins, Backend-Module und Services sind optionale Bestandteile von Extensions, eine TYPO3-Erweiterung kann beliebig viele dieser Elemente beinhalten. Frontend-Plugins werden im Backend – wie in Abbildung 1.1 zu sehen – als Seiteninhalt vom Typ *Plugin* auf einer Seite platziert und erzeugen im Frontend eine Ausgabe auf der entsprechenden Seite.

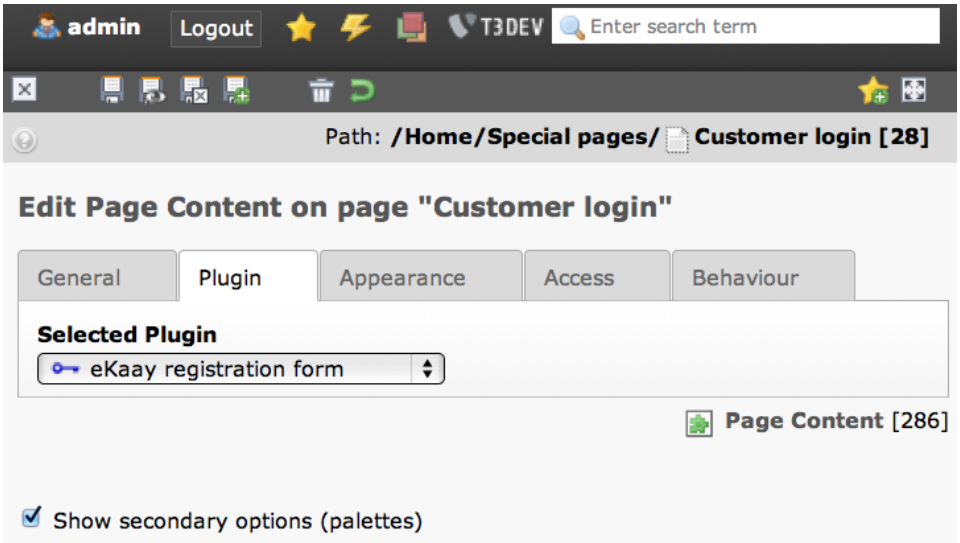

Abbildung 1.1: Anlegen eines neuen Seiteninhalts vom Typ Frontend-Plugin im Backend, Jonas Reichert

Backend-Module erweitern die Funktionalität des Backends, indem sie entweder eine der vier vorhandenen Modul-Kategorien – *Web*, *File*, *User Tools* oder *Help* erweitern, oder eine eigene Kategorie definieren. Im ersten Fall spricht man von *Sub-Moduls*, im Zweiten von *Main-Moduls*.

Extensionmanager Der Extensionmanager ist ein in die TYPO3-Grundinstallation integriertes Backend-Modul, über den die bereits installierten Extensions verwaltet, sowie Neue aus dem integrierten TYPO3 Extension Repository Browser importiert werden können. Zusätzlich können über die Schaltfläche *Upload extension file* Extensions wahlweise auch in Form von gepackten Dateien mit der Dateiendung **.t3x* hochgeladen werden.

1.4.2 eKaay

eKaay ist ein Verfahren zur Benutzerauthentifizierung auf einer Webseite, das seit 2009 von Mitarbeitern des Lehrstuhls für Formale Sprachen am Wilhelm-Schickard-Institut für Informatik der Universität Tübingen entwickelt wird. Die Bezeichnung *eKaay* wird in dieser Arbeit synonym verwendet für das eigentliche Verfahren selbst, sowie für dessen Implementierung.

Das eKaay-Verfahren ermöglicht dem Benutzer, im weiteren Verlauf der Arbeit auch gleichbedeutend *Client* genannt, den Zugang zum geschützten Bereich einer Webseite per Abfotografieren eines 2D- bzw. QR-Codes mit dem Smartphone. Der Webserver dieser Webseite wird im Folgenden *Account-Server* genannt, die eKaay-Implementation, welche als Server fungiert, wird mit *eKaay-Server* bezeichnet. Im Gegensatz zur herkömmlichen Benutzerauthentifizierung, bei der der Benutzer anhand von seinen per Formular übertragenen Login-Daten[1] identifiziert wird, erfolgt keine Eingabe über die Tastatur. Abbildung 1.2 zeigt den schematischen Aufbau der eKaay-Implementierung. Der eigentliche Vorgang der Authentifizierung, nämlich die Verifizierung der Behauptung der Authentizität des Benutzers, findet zwischen dem Handy und dem eKaay-Server statt. Lediglich deren Ergebnis – Erfolg oder Fehlschlag – wird über die Login-Schnittstelle dem Account-Server kommuniziert. Dieses Verfahren hat somit den Vorteil, sicher gegenüber auf dem Computer des Benutzers installierter Malware – wie Keylogger und Trojanische Pferde –[2] zu sein: Da keine Tastatureingabe erfolgt, kann diese auch nicht protokolliert werden. Der eKaay-Core verwendet die Template Engine *Smarty*, das Ajax-Framework *xajax* und benötigt eine MySQL Datenbank, der Code basiert größtenteils auf PHP.

[1] üblicherweise Benutzername und Passwort

[2] Schadprogramme, die durch das protokollieren der Benutzereingaben an einem Computer sensible Daten wie Benutzernamen und Passwörter ausspähen

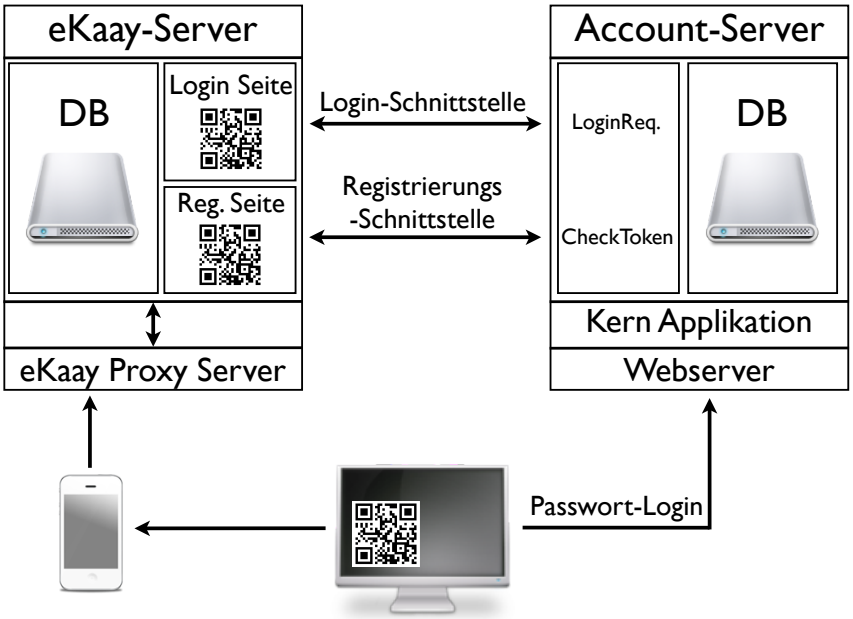

Abbildung 1.2: Übersicht über das eKaay Verfahren, vgl. [Bor12]

Auto-Registrierung Während der Arbeit an der TYPO3-Extension wurde der eKaay-Core um ein weiteres Feature erweitert, welches das automatische Hinzufügen neuer Benutzer zum Account-Server erlaubt – wieder durch Scannen eines QR-Codes mit dem Smartphone. Diese neue Funktionalität wird im weiteren Verlauf dieser Arbeit *Auto-Registrierung* genannt.

Module Die Interaktion zwischen dem Benutzer und eKaay findet ausschließlich über das Abfotografieren von 2D-Codes statt. Es existiert jeweils ein 2D-Code für den Login Vorgang, Einer für die Aktivierung des eKaay-Verfahrens für einen Account des Account-Servers und Einer für die Auto-Registrierung. Im weiteren Verlauf dieser Arbeit werden diese drei QR-Codes, von denen jeder einen abgeschlossenen Vorgang der Authentifizierung – Login, eKaay-Aktivierung und Auto-Registrierung – repräsentiert, auch als Module bezeichnet.

1.5 Typographische Konventionen

In dieser Bachelorarbeit werden die folgenden typografischen Konventionen verwendet:

Kursive Schrift:	Begriffsdefinitionen, Datei- und Verzeichnisnamen, (Button-)Beschriftungen und Namen von Extensions
`Nichtproportionale Schrift:`	Variablen, Konstanten und Werte

2 Schreiben einer TYPO3-Extension

2.1 Lokale und globale Extensions

Vor dem Beginn der Arbeit an der Erweiterung steht die Entscheidung an, ob die Extension lokal oder global in die TYPO3-Installation eingebunden werden soll. Prinzipiell kann eine Extension so programmiert werden, dass sie sowohl lokal als auch global eingebunden werden kann. Unter bestimmten Umständen ist dies aber nicht möglich, so dass die Vor- und Nachteile dieser beiden Varianten im Voraus abzuwägen sind, bevor eine Vereinbarung getroffen wird.

Globale Extensions Globale Extensions werden, ausgehend vom TYPO3 Hauptverzeichnis, im Unterordner *typo3/ext/* gespeichert. Wird eine Extension global eingebunden hat dies den Vorteil, dass beliebig viele TYPO3-Installationen, die den selben *typo3_src/*-Ordner verwenden, auf diese eine Extension Zugriff haben. Der Nutzen von global eingefügten Erweiterungen liegt also in der Einsparung redundanter Arbeitsschritte, welcher aber nur zum Tragen kommt, wenn der obige Fall mehrerer TYPO3-Installationen vorliegt.

Der Nachteil globaler Extensions beruht auf der Tatsache, dass bei einem TYPO3-Systemupdate der komplette Inhalt des *typo3/* Unterverzeichnisses überschrieben wird. Dies bedeutet, dass alle installierten globalen Extensions automatisch gelöscht werden und somit neu installiert werden müssen.[Typ12c]

Lokale Extensions Lokal eingebundene Erweiterungen werden im Unterordner *typo3conf/ext* gespeichert, befinden sich somit außerhalb des bei einem Systemupdate überspielten Verzeichnisses *typo3/* und werden daher nicht gelöscht.

Allerdings können lokal installierte Erweiterungen nicht – wie im vorherigen Absatz beschrieben – von mehreren TYPO3-Installationen genutzt werden. Somit muss die Extension für jede einzelne Installation separat eingebunden werden.[Typ12c]

2.2 Verzeichnisstruktur und Extension Keys

Jede Extension – egal ob lokal oder global eingebunden – besitzt eine einheitliche Verzeichnisstruktur, welche wie in Abbildung 2.1 zu sehen mit einer Baumstruktur veranschaulicht werden kann. Das oberste Verzeichnis der Extension – und somit

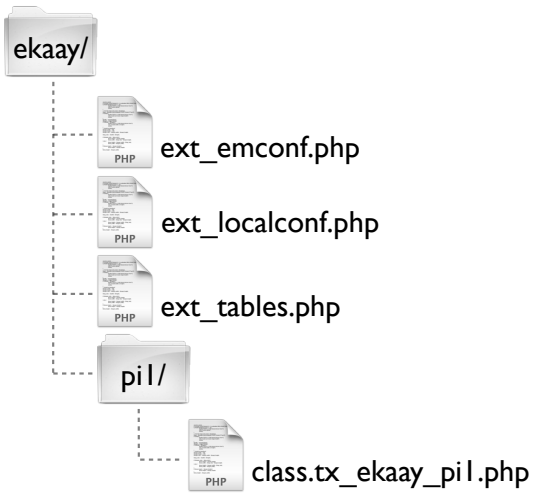

Abbildung 2.1: Vereinfachte Verzeichnisstruktur einer TYPO3-Extension, Jonas Reichert

das Wurzelelement der Baumstruktur – trägt den Namen des der Extension zugewiesenen *Extension Keys*. Bei einem Extension Key handelt es sich um einen Bezeichner, welcher die Extension eindeutig kennzeichnet. Die Bezeichnung des Extension Keys ist dabei nicht beliebig, sondern muss bestimmte Regeln befolgen.[Ext12]

ext_emconf.php Fester Bestandteil jeder TYPO3-Extension ist die Datei *ext_emconf.php*. Sie enthält zur Erweiterung zugehörige Metadaten wie z. B. Name, Beschreibung, Versionsnummer und Status der Extension in Form von Key-Value Paaren, die sich beliebig anpassen lassen. Tabelle 2.1 zeigt eine Auflistung der wichtigsten Konfigurationsmöglichkeiten.

Zusätzlich befinden sich am Dateiende md5-Hashwerte aller zur Extension zugehörigen Dateien. Auf diese Weise kann der Extensionmanager erkennen, ob Dateien der Extension verändert wurden.[typ12a]

ext_localconf.php In *ext_localconf.php* kann das TYPO3-interne Konfigurationsarray `$TYPO3_CONF_VARS`, welches in der Datei *localconf.php* initialisiert wird, um weitere Werte ergänzt werden.[Typ12b] Außerdem werden hier weitere Elemente

Key	Bedeutung
title	Name der Extension
description	Beschreibung der Extension
author	Name des Urhebers
author_email	Email Adresse des Urhebers
shy	Wenn aktiviert (Wert 1) wird die Extension im Extensionmanager nur angezeigt, wenn dort *Display shy extensions* aktiviert ist
dependencies	Abhängigkeiten von anderen Extensions
priority	(`top` | `bottom`) Regelt die Reihenfolge, in der die Extension relativ zu anderen Extensions geladen wird. Zuerst werden alle mit top, danach die mit bottom gekennzeichneten Extensions geladen. Die Abfolge des Einbindens von Extensions der selben priority wird nach dem Datum der Installation geregelt. (Älteste zuerst)
State	Status der Extension (`Experimental` | `Alpha` | `Beta` | `Stable`)
clearCacheOnLoad	(`1` | `0`) Wenn gesetzt, wird nach der Installation der Extension automatisch der Frontend-Cache geleert
version	Die Versionsnummer der Extension
constraints	Aufgeteilt in `depends`, `conflicts` und `suggests`. In jedem dieser 3 Felder lassen sich Bedingungen formulieren, die für den korrekten Betrieb der Extension notwendig sind, wie z. B. die minimal benötigte TYPO3/PHP Version oder andere Extensions, die die einwandfreie Funktionsweise dieser Erweiterung hindern.

Tabelle 2.1: Konfiguration *ext_emconf.php*, u. a. [Rip08]

Typ	Präfix
Frontend-Plugin	`pi*`
Service	`sv*`
Backend-Modul	`mod*`

Tabelle 2.2: Typen und deren Präfixe, [Rip08]

der Extension – wie Frontend-Plugins und Services, siehe Abschnitt 1.4.1 und 3.2– registriert.

`$TYPO3_CONF_VARS` wird in dieser Reihenfolge initialisiert: zuerst wird die Datei *localconf.php* eingebunden, dann folgt die Generierung der Konstanten für den Datenbankzugriff. Erst danach werden alle *ext_localconf.php* Dateien der Extensions ausgewertet, somit ist sichergestellt, dass weitestgehend alle Einstellungen der *localconf.php* Datei ändern kann, nicht aber die Datenbankparameter.[typ12a]

ext_tables.php Die optionale Datei *ext_tables.php* erweitert – ähnlich wie *ext_localconf.php* die Datei *localconf.php* ergänzt – die ausgehend vom TYPO3-Hauptverzeichnis im Verzeichnis *t3lib/stddb/* liegende TYPO3-Datei *tables.php* um Konfigurationen für z. B. Tabellen, Module und Backendstile.[Kno12]

pi/sv/mod-Ordner In die Extension integrierte Frontend-Plugins, Services und Backend-Module werden jeweils in einem eigenen Ordner unterhalb des obersten Verzeichnisses der Extension eingebunden. Der Verzeichnisname ergibt sich dabei aus dem dem jeweiligen Typ zugehörigen Präfix – siehe Tabelle 2.2 – sowie einer fortlaufenden Nummerierung beginnend bei 1 für die erste Instanz eines Typs.

Die wichtigste Datei innerhalb dieses Verzeichnisses ist die *class.tx_extKey_Präfix*.php*, wobei *extKey* mit dem Extension Key und *Präfix** mit dem Verzeichnisnamen nach obiger Regel zu ersetzen ist. Im in Abbildung 2.1 gezeigten Beispiel handelt es sich somit um ein in die Extension mit dem Extension Key *ekaay* integriertes Frontend-Plugin. In jedem Fall muss diese Klasse von der ihrem Typ zugehörigen Basisklasse erben.

2.3 Kickstarter

Das im vorigen Abschnitt beschriebene Grundgerüst[3] einer Extension lässt sich komfortabler und schneller mit der TYPO3-Erweiterung Kickstarter anlegen, die jedoch

[3] Verzeichnis- und Dateistruktur

nicht zum Grundumfang des TYPO3-Systems nach dessen Installation angehört und deshalb bei Bedarf manuell nachinstalliert werden muss. Nach der Installation des Kickstarters kann dieser ab sofort im Extensionmanager aufgerufen und mit dem Anlegen einer Erweiterung begonnen werden. Abbildung 2.2 demonstriert die Anwendung des Kickstarters beim Erstellen einer Extension. Das Ergebnis dieser

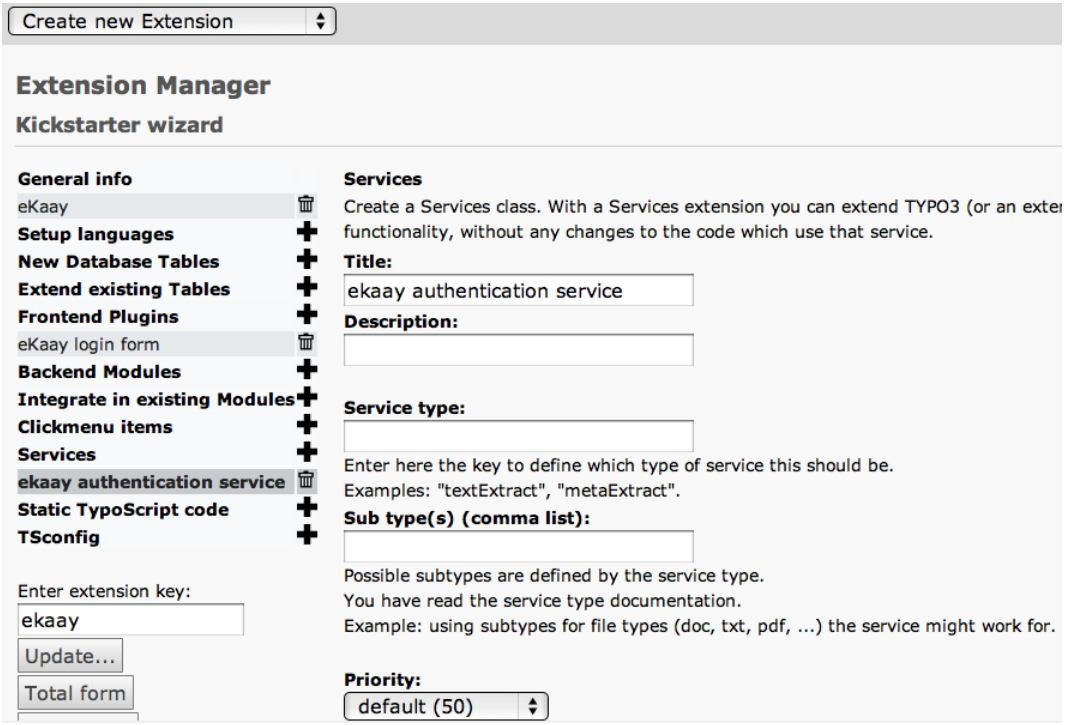

Abbildung 2.2: Die Extension Kickstarter, Jonas Reichert

Beispielkonfiguration wäre die aus Abbildung 2.1 bekannte Verzeichnisstruktur mit dem Frontend-Plugin *eKaay login form* im Ordner *pi1*, ebenso wie ein zusätzlicher Ordner *sv1*, welcher den Service *eKaay authentication service* enthalten würde.

Neben Frontend-Plugins und Services lassen sich u. a. auch bestehende Datenbanktabellen verändern oder Neue erstellen, mehrere Sprachen einbinden sowie statischer Typoscript Code einbinden. Es gilt zu beachten, dass der Kickstarter lediglich als Hilfe für das für den Programmierer langwierige Anlegen des Grundgerüsts der Erweiterung dient. Er ist auf keinen Fall für das Editieren von bestehenden Extensions zu gebrauchen, da dies alle nicht vom Kickstarter getätigten Änderungen rückgängig machen würde.

3 Benutzerauthentisierung in TYPO3

In Abschnitt 1.4.1 wurden die unterschiedlichen Benutzergruppen in TYPO3 – Frontend- und Backend-Benutzer – bereits vorgestellt. Diese beiden Benutzergruppen unterliegen sowohl in der reinen Datenhaltung – Frontend-Benutzer werden in der Datenbank-Tabelle *fe_users* gespeichert, Backend-Benutzer in der Tabelle *be_users* – als auch während des Authentifizierungsvorgangs einer strikten Trennung. So kann sich keine der genannten Benutzergruppen beim Login-Formular der Anderen authentisieren.

3.1 Formen der Authentisierung

Neben der klassischen, in der TYPO3-Grundinstallation bereits mitgelieferten Benutzerauthentifizierung mittels Benutzernamen und Passwort lässt sich TYPO3 durch die Installation von weiteren Extensions mit zusätzlichen Authentifizierungsmethoden erweitern. Tabelle 3.1 zeigt einen kleinen Ausschnitt der im TYPO3 Extension Repository (TER)[4] befindlichen Erweiterungen, die zusätzliche Authentisierungsmechanismen implementieren.

Diese Vielfalt an Authentisierungsmethoden wird ermöglicht durch das im Abschnitt 3.2 beschriebene Konzept der *(Authentication-)Services*.

3.2 Services

Services sind PHP-Klassen innerhalb von Extensions, welche die Funktionalität von TYPO3 erweitern, ohne den ursprünglichen TYPO3-Code bzw. den der Extension zu verändern. Zur Veranschaulichung des Zwecks der Services soll folgendes Beispiel dienen.

Angenommen einer Extension hätte zur Erfüllung ihrer Aufgabe mehrere, unabhängige Funktionen zur Verfügung, die diese Aufgabe unterschiedlich gut lösen. Die Funktion, die die besten Resultate liefert, stütze sich auf Module oder Erweiterungen, die nicht jeder Webserver zwangsweise installiert hat. Ein in die Extension

[4] Genauere Informationen über das TER liefert Abschnitt 5.1

Extension	Beschreibung
cc_ipauth	Erlaubt die Authentifizierung von Benutzern über deren IP-Adresse
rlmp_extdbauth	Authentifiziert Benutzer aus externen Datenbanken
sf_imap_login	Gestattet die Benutzerauthentifizierung über das IMAP Protokoll
ig_ldap_sso_auth	Importiert Benutzer aus einer LDAP-Verzeichnisstruktur
ml_fileauth	Authentifiziert in CSV-Dateien gespeicherte Benutzer
srijan_ldap_auth	Siehe ig_ldap_sso_auth
radius_auth	Authentifiziert Benutzer an einem RADIUS-Server
shibboleth_auth	Authentisiert Benutzer mit per Single Sign On Shibboleth

Tabelle 3.1: Überblick über verschiedene Authentisierungsmechanismen, entnommen dem TYPO3 Extension Repository[TER12]

integrierter Service kann den Webserver auf das Vorhandensein dieser optionalen Module überprüfen und je nachdem die geeignete Funktion aufrufen.

Des weiteren können mehrere Services implementiert werden, die die selbe Aufgabe haben, aber nacheinander in einer bestimmten Reihenfolge aufgerufen werden. Sobald ein Service erfolgreich ist, wird diese Kette unterbrochen. Die erwähnte Reihenfolge hängt dabei von zwei Parametern ab: *priority* und *quality*. Priority ist mit einem beliebigen, positiven numerischen Wert definiert – in der Praxis liegt dieser normalerweise zwischen 0 und 100. Die Services werden vom TYPO3-System anhand ihrer priority absteigend sortiert, der Service mit der höchsten priority wird als Erstes verwendet. Sollten mehrere Services den gleichen priority-Wert haben, entscheidet nun deren quality – auch ein Wert zwischen 0 und 100 – über die Reihenfolge. [Unb09]

3.3 Authentication Services

Aus der Sicht des Programmierers handelt es sich bei einem Authentication Service um einen Service des Typs *auth*. Anschaulich können Authentication Services als Dienste beschrieben werden, die auf einen *Authentication Request* warten und diesen entsprechend verarbeiten. Ein Authentication Request ist eine Anfrage zur Benutzeridentifikation und -autorisierung an das TYPO3-System.

3.3.1 Funktionsweise

Abbildung 3.1 zeigt beispielhaft die Funktionsweise von Authentication Services. Zunächst wird der Service mit der höchsten priority geladen, dieser prüft die Authen-

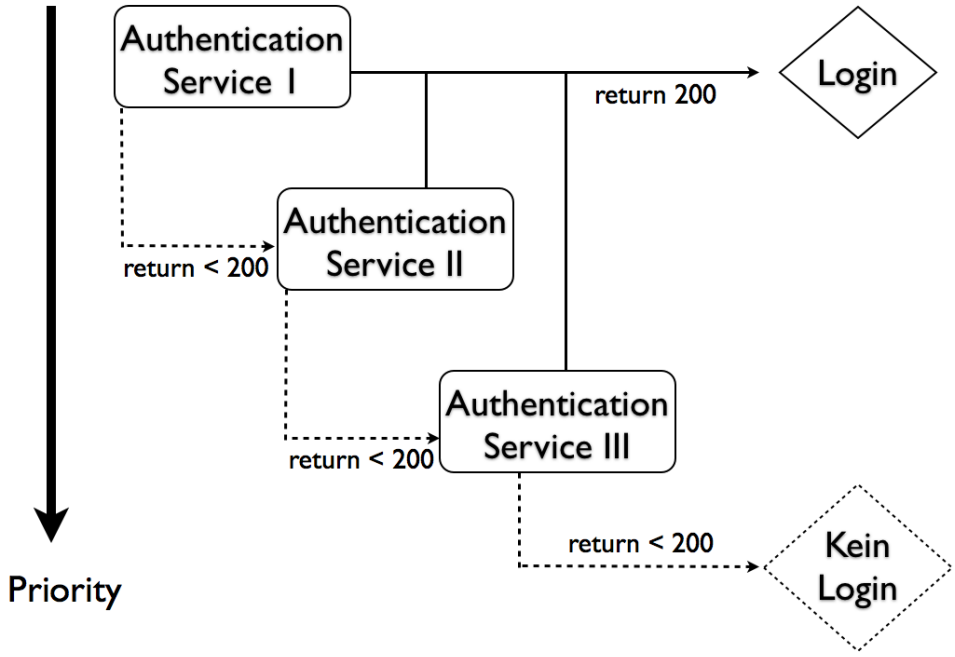

Abbildung 3.1: Flussdiagramm des TYPO3 Authentisierungsmechanismus, Jonas Reichert

tizität des Benutzers, der den Authentication Request gesendet hat. Im Erfolgsfall gibt der Authentication Service die Zahl 200 zurück und der Benutzer ist erfolgreich angemeldet. Schlägt die Prüfung der Authentizität fehl, wird eine Zahl, die kleiner als 200 ist, zurückgegeben. Nun werden so lange die nach der priority sortierten Authentication Services aufgerufen, bis einer erfolgreich ist und der Benutzer somit angemeldet wird, oder bis alle erfolglos abgearbeitet wurden. In dem Fall schlägt die Benutzerauthentisierung fehl.

3.3.2 Subtypes

Da es nicht für jeden Anwendungsfall nötig ist, den kompletten Service neu zu schreiben, bietet TYPO3 die Möglichkeit, durch das Konzept der *Subtypes* nur die Teile des Services anzugeben, die implementiert werden sollen.[Fri05] Der Service-Type *auth* bietet die in Tabelle 3.2 aufgeführten Subtypes:

Subtype	Beschreibung
getUserFE getUserBE	Findet einen (Frontend- / Backend-)Benutzer auf der Basis von übergebenen Informationen wie z. B. dem Benutzernamen
authUserFE authUserBE	Authentisiert den (Frontend- / Backend-)Benutzer
getGroupsFE getGroupsBE	Findet die Benutzergruppe des Benutzers
authGroupsFE authGroupsBE	Überprüft die Gültigkeit der Benutzergruppe des Benutzers

Tabelle 3.2: Überblick über die Subtypes des Service-Type auth

Für einen Authentification Service, der beispielsweise nur Frontend-Benutzer authentisiert, reicht es demzufolge aus, den Service nur die beiden Subtypes *getUserFE* und *authUserFE* implementieren zu lassen. In diesem Fall würden in der zum Service gehörenden Datei *class.tx_extKey_sv*.php* entsprechend nur die beiden Funktionen getUser() und authUser($user) überschrieben werden. In beiden Funktionen kann nun auf verschiedene Daten zugegriffen werden. Beispielsweise enthält das Array `login` die per Login-Formular übermittelten Daten, wie den Benutzernamen oder das Passwort in verschiedenen Variationen.[5]

[5] Als Plaintext, einfach oder mehrmals gehashten md5-Hashwert

4 Die eKaay TYPO3-Extension

Wie bereits in Abschnitt 1.2 geschildert, soll die Einbindung des eKaay-Verfahrens in eine TYPO3-Installation so einfach wie möglich vonstattengehen. Andererseits darf diese Minimierung des Arbeitsaufwandes keineswegs dazu führen, dass der Ort – respektive die Seiten – an dem die einzelnen Module des eKaay-Verfahrens platziert werden, fest vorgegeben wird, da dies zu Lasten der Anpassbarkeit der Extension gehen würde.

Hinsichtlich des im Abschnitt 1.2 gesteckten Ziels, muss eine Einschränkung zugestanden werden: Da sich die – für eine Einbindung des eKaay-Login Moduls notwendige – dynamische Anpassung der Backend-Login Seite des TYPO3-Systems als schwierig herausgestellt hat, unterstützt die eKaay-Extension nur den Login für Frontend-Benutzer.

4.1 Aufbau der eKaay-Extension

Aufgrund der zu Beginn dieses Kapitels beschriebenen Anforderungen hat die eKaay-Extension folgenden Aufbau, der größtmögliche Anpassungsmöglichkeiten für den Administrator bei gleichzeitiger Minimierung des für die Installation nötigen Aufwands bieten soll.

4.1.1 eKaay-Core

Die eKaay-Core Dateien werden in einem separaten Verzeichnis unterhalb des Wurzelverzeichnisses der Extension eingebunden. Wie bereits erwähnt, dürfen die in diesem Verzeichnis befindlichen Dateien, welche bei einem Update des ekaay-Cores überschrieben werden, nicht verändert werden.[6]

[6] siehe Abschnitt 1.2

settings.php Zentraler Bestandteil des eKaay-Cores ist die Datei *settings.php*, welche außerhalb der von einem Update betroffenen Verzeichnisse liegt. In ihr werden u. a. Angaben zum Server, wie z. B. die Server-Adresse, in Form von Konstanten gespeichert, die die ursprüngliche eKaay-Implementierung für die korrekte Ausführung ihrer Skripte benötigt. Hierzu wird zusätzlich die im Hauptverzeichnis der Extension liegende Datei *db_constants.php* inkludiert, welche weitere Konstanten, welche von den in diesem Abschnitt beschriebenen Frontend-Plugins aus TYPO3 ausgelesen wurden, definiert. Tabelle 4.1 zeigt einen Überblick über einige der in der *settings.php* definierten Konstanten. Die erneute Speicherung der Datenbank-Zugangsdaten erhöht dabei nicht das Gefährdungspotenzial: In der Datei *localconf.php* werden Diese bereits im Klartext gespeichert. In der Standardkonfiguration ohne die eKaay-Extension ist TYPO3 also sicher unter der Voraussetzung, dass dieses Verzeichnis vor fremdem Zugriff geschützt ist. Unter dieser Voraussetzung stellt die erneute Speicherung der MySQL-Zugangsdaten in einem Verzeichnis unterhalb der *localconf.php* – und somit ebenfalls in einem zu schützenden Verzeichnis – kein zusätzliches Sicherheitsrisiko dar.

4.1.2 Frontend-Plugin 1: eKaay registration form

Dieses Frontend-Plugin zeigt einem eingeloggten Frontend-Benutzer das für die Registrierung für das eKaay-Verfahren zuständige Modul an. Dieses Modul wird, ebenso wie in der ursprünglichen eKaay-Implementierung praktiziert, in einen Inline Frames eingebettet. Ist kein Frontend-Benutzer eingeloggt, wird eine leere Ausgabe erzeugt.

Zur Validierung der Authentizität des eingeloggten Benutzers berechnet dieses Plugin noch zusätzlich ein temporäres Token. Um Verwechslungen mit dem beim Login-Vorgang versendeten Token vorzubeugen, wird dieses Token fortan *Reverse-Token* genannt. Dieses Reverse-Token wird in einer eigens dafür vom eKaay-System angelegten Datenbanktabelle gespeichert. Zusätzlich wird das Reverse-Token an die vom Inline Frame aufgerufene URL angehängt, was den eKaay-Server dazu veranlasst, die Gültigkeit des Reverse-Tokens für den eingeloggten Benutzer beim Account-Server zu prüfen. Bei jedem Seitenaufruf dieses Plugins werden die veralteten, temporären Reverse-Tokens gelöscht.

4.1.3 Frontend-Plugin 2: eKaay login form

eKaay login form ist für die Anzeige des für den Login zuständigen QR-Codes verantwortlich. Im Gegensatz zum im vorigen Abschnitt 4.1.2 beschriebenen Frontend-

Konstante / Variable	Beschreibung
`OS_CONFIG_PATH`	absoluter Pfad zum Verzeichnis der *settings.php*
`OS_SYSTEM_PATH`	Der um das Unterverzeichnis *system/* erweiterte `OS_CONFIG_PATH`
`OS_WWWDIR_PATH`	Der um das Unterverzeichnis *wwwsrv/* erweiterte `OS_CONFIG_PATH`
`$path_to_dir`	relativer Pfad zum TYPO3-Hauptverzeichnis
`DB_USER`	Zugangsdaten zur Datenbank
`DB_PASS`	
`DB_NAME`	
`DB_SERVER`	
`DB_PREFIX`	Präfix der von eKaay erzeugten Datenbanktabellen
`OS_ACCOUNTSRV`	URL zum TYPO3 Hauptverzeichnis
`OS_CALL_SCRIPT`	Gibt an, an welches Skript Benutzername und Token gesendet werden sollen
`OS_POST_PARNAME_USERNAME`	Definiert den Namen der Variablen, welche den Benutzername speichert
`OS_POST_PARNAME_PASSWORD`	Definiert den Namen der Variablen, welche das Token speichert
`OS_POST_EXTRA_PARNAME1`	Definiert die zusätzlich für den Login-Vorgang nötige Variable `logintype`
`OS_POST_EXTRA_VALUE1`	
`OS_POST_EXTRA_PARNAME2`	Definiert die zusätzlich für den Login-Vorgang nötige Variable `pid`
`OS_POST_EXTRA_VALUE2`	
`OS_CHECK_REV_TOKEN`	(`true` | `false`) Gibt an, ob das Reverse-Token im für die Registrierung verantwortlichen Modul überprüft werden soll
`OS_SERVER_AUTREG_REQUESTS`	Definiert die für die Auto-Registrierung benötigten, per Handy zu übertragenden Parameter

Tabelle 4.1: Die wichtigsten Konstanten der *settings.php*-Datei

Plugin wird hier nur eine Ausgabe erzeugt, wenn der Seitenaufruf von keinem eingeloggten Benutzer erfolgt.

Außerdem werden aufgrund der in Abschnitt 1.2 angesprochenen Einschränkung, welche die Veränderung des Codes der eKaay-Implementierung untersagt, was u. a. zur Folge hat, dass innerhalb des eKaay Codes nicht auf TYPO3-spezifische Variablen zugegriffen werden kann, drei Parameter aus TYPO3 ausgelesen und in einer für das eKaay-System zugänglichen Datei gespeichert. Diese drei Parameter umfassen die TYPO3-Systemsprache, die TYPO3-spezifische `id` der Seite, auf der das *eKaay login form*-Plugin eingebunden ist, sowie die korrekte Server-Adresse, unter der die TYPO3-Webseite zu erreichen ist. Die Texte, die von den angesprochenen eKaay-Modulen an die Ausgabe gereicht werden, sind in eKaay momentan zweisprachig vorhanden: In Deutsch und in Englisch. Abhängig von der TYPO3-Systemsprache werden die eKaay-Texte entweder in Deutsch, wenn die TYPO3-Systemsprache deutsch ist, oder in Englisch für alle anderen Sprachen, ausgegeben. Die weiteren Parameter werden vom eKaay-Core benötigt.

4.1.4 Frontend-Plugin 3: eKaay Auto registration form

Das dritte Frontend-Plugin der eKaay-Extension bettet den für die Auto-Registrierung nötigen QR-Code in die Ausgabe ein. Dieses Plugin wird ebenfalls nur angezeigt, wenn der Client beim Seitenaufruf der Seite, auf der dieses Plugin eingebunden ist, nicht beim Frontend angemeldet ist.

Ebenso wie beim – im vorigen Abschnitt 4.1.3 beschriebenen – Frontend-Plugin *eKaay login form* liest dieses Plugin zwei Parameter aus dem TYPO3-System und schreibt deren Werte in Form von Konstanten in die angesprochene, vom eKaay-Core lesbare Datei. Bei diesen Parametern handelt es sich zum einen um eine TYPO3-spezifische `id` einer gültigen Benutzergruppe und zum anderen um die gültige `id` einer TYPO3-Seite, auf der die Frontend-Benutzer eingetragen werden. Beide Parameter dienen der korrekten Eintragung des neuen Benutzers in die Datenbank.

4.1.5 Service: eKaay authentication service

Um eine korrekte Authentifizierung am TYPO3-System zu gewährleisten, implementiert die eKaay-Extension einen – in Abschnitt 3.3 vorgestellten – Authentication Service. Dieser implementiert die Subtypes *getUserFE* und *authUserFE*, demzufolge werden in der Service-Datei *class.tx_ekaay_sv1.php* die Methoden getUser() sowie authUser($user) überschrieben.

4.2 Der Authentifizierungsvorgang

Die eKaay-Extension ist so konfiguriert, dass beim Scannen des vom *eKaay login form*-Frontend-Plugin generierten QR-Codes der auf dem Smartphone gespeicherte Benutzername sowie ein Token, zusammen mit dem für die Auslösung des Login Vorgangs nötigen Parametern an TYPO3 gesendet wird. Nun versucht das TYPO3-System wie in Abbildung 3.1 veranschaulicht, den Benutzer mittels der mitgeschickten Daten – Benutzername und Token – gegen alle bei TYPO3 registrierten Authentication Services zu authentifizieren. Dies wird bei allen Services mit Ausnahme des *eKaay authentication service* mit ziemlicher Sicherheit fehlschlagen, da die Wahrscheinlichkeit, dass das mitgelieferte Token auch auf das von anderen Authentifizierungsmethoden abgefragte Passwort passt, äußerst gering ist.

getUser und Chain of Services Die im *eKaay authentication service* überschriebene Methode getUser ähnelt dabei stark der von der Basisklasse tx_sv_authbase geerbten Variante, allerdings mit folgender Erweiterung: Während die Version aus der Basisklasse nur statisch Funktionen verwendet, die die TYPO3-interne Datenbanktabelle fe_users nach dem Benutzer durchsucht, verwendet der Authentication Service der eKaay-Extension eine so genannte *chain of services*:[Unb09] Dabei werden die getUser-Methoden aller vorhandenen Authentication Services verwendet, um den gewünschten Benutzer aufzufinden. Diese Methode ist flexibler als die statische Benutzersuche in der TYPO3-internen Datenbank, da auch die Möglichkeit, dass das TYPO3-System eine, die Benutzerverwaltung betreffend, abweichende Datenhaltung nutzt,[7] abgedeckt und unterstützt wird.

authUser Die Funktion *authUser* leitet Benutzername und Token per CURL[8] an den eKaay-Server weiter, der die Validität des zum Benutzernamen zugehörigen Tokens überprüft. Ist der Rückgabewert des CURL-Skripts `ok`, ist das Token valide und die Authentisierung des Benutzers wird durch die Rückgabe des Werts `200` erfolgreich abgeschlossen.

[7] Unter Umständen bedingt durch andere installierte Authentifizierungs-Methoden

[8] Bibliothek; erlaubt die Verbindung zu Webservern durch Verwendung unterschiedlichster Protokolle[CUR12]

5 TER Upload und Rechtliches

5.1 TYPO3 Extension Repository

Das *TYPO3 Extension Repository* (TER) ist ein – zum einen über die offizielle Webseite[TER12], zum anderen über den Extension-Manager – öffentlich zugängliches Verzeichnis, in dem Extensions gespeichert sind. Die im TER gespeicherten Erweiterungen lassen sich über den Extension-Manager schnell zu TYPO3 hinzufügen und entfernen. Um eine selbst entwickelte Extension ins TER hochzuladen, muss dazu zuerst ein Account registriert und mittels diesem der gewünschte Extension Key für die Erweiterung reserviert werden. Anschließend kann, nach der Auswahl der Erweiterung mit dem reservierten Extension Key im Extension-Manager, die Seite *Upload to TER* – siehe Abbildung 5.1 – geladen werden. Nach der Eingabe der

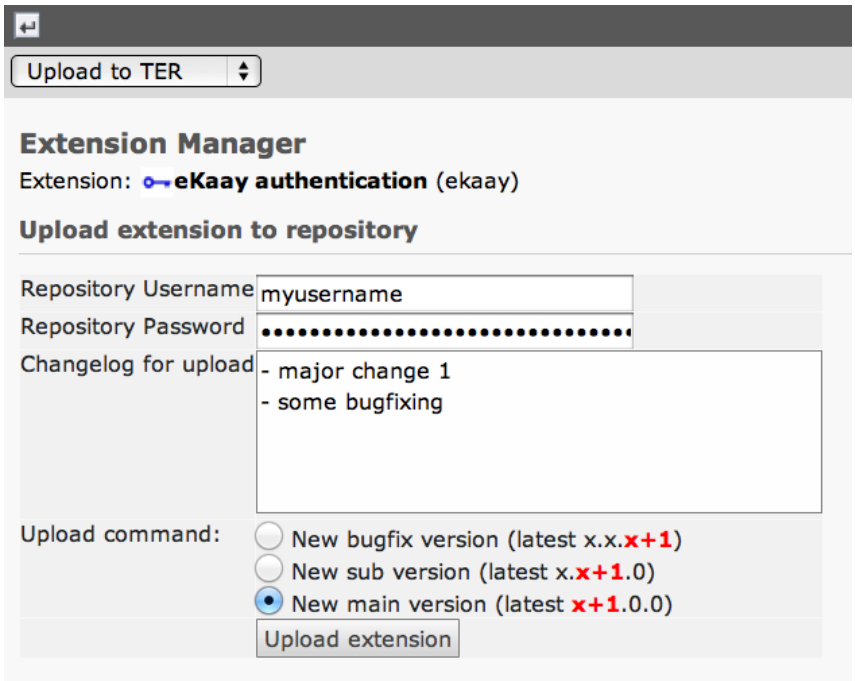

Abbildung 5.1: Extension Upload im Extension-Manager

Login-Daten des zuvor erstellten Accounts wird durch Klick auf den – mit *Upload Extension* beschrifteten – Button der Upload eingeleitet.

5.2 Lizenzierung

Durch die Registrierung eines Extension Keys akzeptiert man automatisch die von der TYPO3-Association formulierten Nutzungsbedingungen für den TER-Upload, welche besagen, dass jeder ins TYPO3 Extension Repository hochgeladene Inhalt unter der GPL Lizenz veröffentlicht wird.[Ext12] Unter Abschnitt „2. Basic Permissions“ der *GNU General Public License* befindet sich u. a. folgender Abschnitt:

> „You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.“

Demnach ist es zulässig, unter GPL-Lizenz veröffentlichte Software unverändert zu vervielfältigen sowie weiterzuverbreiten, jedoch nur unter der Voraussetzung, dass die Kopie ebenfalls unter GPL-Lizenz genommen wird.[GPL07]

6 Fazit

Das in Abschnitt 1.2 formulierte Ziel, die Einbindung der eKaay-Extension – und damit die Integration des eKaay-Verfahrens – in TYPO3 möglichst einfach und flexibel zu gestalten, wurde mit der Imlementierung der Extension erreicht. Der Administrator ist mit der minimal nötigen Anzahl an Klicks imstande, die Extension zu installieren und zu aktivieren. Anschließend können die drei eKaay-Module, welche für Login, Aktivierung des eKaay-Verfahrens für einen Frontend-Benutzer, sowie für die Auto-Registrierung – ebenfalls für Frontend-Benutzer – zuständig sind, beliebig auf der Seite platziert werden.

Allerdings stellt die in Kapitel 4 erklärte Problematik, nach der sich nur Frontend-Benutzer mit dem eKaay-Verfahren bei TYPO3 einloggen können, durchaus eine Einschränkung der Funktionalität und eine Verfehlung des im Abschnitt 1.2 „Ziel der Arbeit“ allgemein formulierten Anspruchs, dass alle TYPO3-Benutzer[9] sich mit dem eKaay-Verfahren anmelden können müssen, dar. Die Tatsache, dass die Zahl der Backend-Benutzer im Vergleich zu der Zahl der registrierten Frontend-Benutzer in der Regel deutlich geringer sein dürften, relativiert diese Einschränkung ein wenig.

Die Extension funktioniert auf den zwei zur Verfügung stehenden Testsystemen wie gewünscht, das eKaay-Verfahren wurde erfolgreich in TYPO3 integriert. Aufgrund der zum eKaay-Core bestehenden Schnittstelle, welche über nicht in TYPO3 eingebettete Skripte kommuniziert, kann in Diesen nicht auf TYPO3-Funktionen wie beispielsweise die Datenbank Wrapper-Klasse, die Schutz gegenüber SQL Injections bietet, zugegriffen werden. Aus diesem Grund mussten die Skripte, welche SQL-Abfragen an der Datenbank durchführen, durch Optimierung der SQL-Queries gegen die Bedrohung durch SQL Injections gesichert werden.

Zusätzlich zu den im Abschnitt Ziel 1.2 formulierten Zielen wurde noch das Feature Auto-Registrierung implementiert, welches es ermöglicht, einen Benutzer per Scannen eines QR-Codes automatisch bei TYPO3 anzulegen. Zu beachten gilt, dass diese Form der Registrierung aufgrund der im obigen Absatz angesprochenen, fehlenden Möglichkeit, TYPO3-eigene Funktionen innerhalb von externen Skripten aufzurufen, den korrekten Ablauf über TYPO3-Funktionen umgeht und den Benutzer durch

[9] Frontend- und Backend-Benutzer

Einfügen in die TYPO3-Datenbank hinzufügt. Dies kann für den Fall, dass die Benutzer in einer externen Datenbank gespeichert werden, vereinzelt zu Fehlverhalten führen.

Vom ursprünglich geplanten Upload der eKaay-Extension in das TYPO3 Extension Repository wurde nach Prüfung der Rechtslage Abstand genommen. Da – wie in Abschnitt 5.2 dargelegt – die Extension durch den Upload unter GPL Lizenz gesetzt würde, würde dies die Verbreitung und Veränderung des Quellcodes ohne unsere Genehmigung zulassen. Aufgrund bestehender Patentrechte für das eKaay-Verfahren wäre dies nicht akzeptabel.

Literaturverzeichnis

[Bor12] Borchert, Bernd: *eKaay Schnittstellen*. `http://www.ekaay.com/implement`, 11.02.2012

[CUR12] The PHP Group: *CURL - Einführung*. `http://www.php.net/manual/de/intro.curl.php`, 14.02.2012

[Die09] Diedrich, Dr. O.: *Trendstudie Open Source - Wie Open-Source-Software in Deutschland eingesetzt wird*. `http://www.heise.de/open/artikel/Eingesetzte-Produkte-224518.html`, 2009

[Ext12] TYPO3 Association: *Extension Keys*. `http://typo3.org/extensions/extension-keys/`, 2012

[Fri05] Fritz, Ren: *Authentication Services*, 2005

[GPL07] Free Software Foundation: *GNU General Public License (GPL) v3.0*. `http://www.gnu.org/licenses/gpl.html`, 2007

[Kno12] Knoll, Dipl. Ing. (FH) E.: *knolledge - TYPO3 Wissensbasis*. `http://www.typo3-nürnberg.de/entwickler/extension-struktur/ext-tablesphp/`, 13.02.2012

[Rip08] Ripfel, Franz: TYPO3-Extensions entwickeln, Teil 2. In: *t3n Magazin* (2008), Nr. 11

[TER12] *TYPO3 Extension Repository*. `http://typo3.org/extensions/repository/`, 14.02.2012

[typ12a] *Dateistruktur einer TYPO3-Extension - Teil I*. `http://www.codeandgraphic.de/webentwickler-blog/typo3/dateistruktur-einer-typo3-extension-teil-i`, 12.02.2012

[Typ12b] Typo3.org: *Core Documentation - Configuration*. `http://typo3.org/documentation/document-library/core-documentation/doc_core_inside/4.1.0/view/3/3/`, 12.02.2012

[Typ12c] Typo3Wiki: *Extension Developers Guide*. `http://wiki.typo3.org/Extension_Developers_Guide#Extension_Maintenance`, 12.02.2012

[Unb09] Unbekannt ; TYPO3 Core Development Team (Hrsg.): *TYPO3-Services*. TYPO3 Core Development Team, 2009

[W3T12] W3Techs: *Usage of content management systems for websites*. `http://w3techs.com/technologies/overview/content_management/all`, 11.02.2012