Europagymnasium Linz

AUHOF

Online-Individualfeedback von SchülerInnen an LehrerInnen in einer Schule

VORWISSENSCHAFTLICHE ARBEIT

aus dem Fachbereich

INFORMATIK

Verfasst von:

Maximilian Levi Heisinger, 8B

Betreuer:

Dipl.-Ing. Stephan Saalberg, bakk.techn.

Angefertigt an der Bildungseinrichtung:

Europagymnasium Auhof, Aubrunnerweg 4, 4040 Linz

This document is set in Palatino, compiled with pdfLaTeX2e and `Biber`.

The LaTeX template from Karl Voit is based on KOMA script and can be found online: https://github.com/novoid/LaTeX-KOMA-template

Eidesstattliche Erklärung

Ich, Maximilian Levi Heisinger, erkläre an Eides statt, dass ich die vorliegende vorwissenschaftliche Arbeit selbständig und ohne fremde Hilfe Dritter verfasst, andere als die angegebenen Quellen und Hilfsmittels nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe. Des Weiteren versichere ich, dass ich diese vorwissenschaftliche Arbeit weder im In- noch im Ausland in irgendeiner Form als Prüfungsarbeit vorgelegt habe.

Linz, am _____ _____

Datum Maximilian Levi Heisinger

Abstract

Diese Arbeit beschreibt die derzeitige Situation in der Landschaft des SchülerInnen zu LehrerInnen Individualfeedbacks an österreichischen Schulen und stellt eine mögliche Lösung vor: Webbasiertes LehrerInnenfeedback, welches am Server der jeweiligen Schule installiert ist und nicht von Außen bereitgestellt wird.

Nach einer umfangreichen Definition des benötigten Funktionsumfangs und der verwendeten Begriffe, wird die technische Implementierung eines derartigen Systems behandelt und die einzelnen Programmkomponenten mitsamt den möglichen Konfigurationsoptionen beschrieben. Die Implementierung wird detaillierter erläutert, weiters werden die benötigten Softwarebibliotheken vorgestellt. Außerdem werden die Grundzüge der Sicherheit für eine Webanwendung dargestellt und die Gründe hinter den einzelnen Entscheidungen während der Entwicklung des hier dargestellten Feedbacksystems angeführt.

Um dem Aufwand der Erstellung gerecht zu werden und das entwickelte System auch produktiv einsetzen zu können, wird außerdem beschrieben, wie der Sprung in die schulinterne IT-Landschaft und in den Alltag der SchülerInnen, sowie der LehrerInnen geglückt ist. Abschließend wird die Vorstellung der neuen Funktionen des Systems im Schülerparlament, bis zur endgültigen Festsetzung als offizielle Feedbackmethode im Europagymnasium Auhof dargestellt.

Inhaltsverzeichnis

Kapitel 1

Einleitung und Motivation

1.1 Derzeitige Situation

Bevor diese Arbeit begonnen wurde, war die Situation um die Frage der richtigen Umsetzung eines LehrerInnenfeedbacks in unserer Schule, nach eigenen Beobachtungen während Schülervertretungskonferenzen und im Schulalltag, noch sehr stark umstritten. Verschiedene Einrichtungen haben eigene Richtlinien zur Erhebung von Rückmeldungen seitens der SchülerInnen an den Lehrkörper und viele LehrerInnen verwenden, oft aufgrund mangelnder Alternativen, vorgefertigte Fragebögen von KollegInnen, oder auch von ganz anderen Einrichtungen.

Diese großen Unterschiede zwischen den eingesetzten Fragebögen, welche selbst innerhalb einzelner Bildungseinrichtungen auftreten können, verursachen sowohl dem Lehrkörper, als auch der Direktion, leider einige Probleme:

1. Der Vergleich eigener Ergebnisse mit den Ergebnissen von KollegInnen wird durch inkonsistente Fragen und verschiedene Bewertungsschemata unnötig verkompliziert und ist auch durch die verschiedenen Kriterien nicht mehr aussagekräftig.
2. KollegInnen legen ihr Augenmerk eventuell auf andere Kriterien, die im eigenen Unterricht keine große Relevanz haben könnten.

3. Fragebögen, welche von Dritten ausgearbeitet und aus Zeitmangel nicht eigens angepasst wurden, passen nicht gut zu den jeweiligen Unterrichtsmethoden.
4. Bestimmte Meinungen von SchülerInnen können von manchen Frageformaten und -formulierungen nicht erfasst werden, weshalb die Ergebnisse nochmals stark an Aussagekraft verlieren können.

Aufgrund dieser und anderer Schwierigkeiten wird eine neue Lösung für ein LehrerInnenfeedback an Schulen gebraucht, welche leichter vergleichbar, einheitlicher auswertbar und schneller durchführbar ist.

1.2 Bedeutung von LehrerInnenfeedback im Schulalltag

Im Schulalltag stehen drei Gruppen in ständiger Wechselwirkung: Das Lehrerkollegium, die Eltern und die Gruppe der SchülerInnen. Die meisten Konflikte entstehen leider zwischen SchülerInnen und ihren ProfessorInnen, da sich diese beiden Gruppen täglich gegenüber stehen und sich in direkter Abhängigkeit zueinander befinden. Durch den sich aufstauenden Stress zwischen diesen beiden Parteien entstehen Probleme, welche, falls zu lange mit einem Lösungsansatz gezögert wird, nur noch durch Interventionen seitens der Eltern, der Direktion und auch viele Gespräche geklärt werden können. Viele dieser Probleme sind hierbei niemandem direkt zuzuschreiben, da im Prüfungsstress und unter ständiger Belastung fast niemand mehr einen klaren Kopf behalten kann und SchülerInnen, so wie auch LehrerInnen, sich schnell gegenseitig falsch verstehen können.

Dieser Druck während der Prüfungszeit kann in dieser Arbeit nicht behandelt werden, da er eher systemtechnischer Natur ist und an einer anderen Stelle angegangen werden müsste. Allerdings können durch besseres gegenseitiges Verständnis die Konfliktherde etwas vermindert und die Situation zumindest etwas entschärft werden. Wenn sich SchülerInnen durch ein anonymisiertes

System direkt mit ihren LehrerInnen und deren Unterrichtsmethoden befassen können, ohne persönliche Konsequenzen fürchten zu müssen, lassen sich viele Unstimmigkeiten bereits im Unterricht behandeln, bevor sie später eventuell noch zu Krisenherden heranwachsen könnten. Falls eine Lehrperson manche Themenbereiche leicht vernachlässigen sollte, oder andere sehr intensiv behandeln würde, könnten die SchülerInnen ihr durch ein solches System ihre kollektive Meinung mitteilen, ohne um ihr individuelles Ansehen beziehungsweise andere, persönliche Konsequenzen beim Lehrer fürchten zu müssen.

Um die verwendeten Begriffe genauer zu definieren, folgt eine persönliche Definitionsliste der einzelnen Fachausdrücke.

LehrerInnenfeedback
: Rückmeldungen von SchülerInnen an ihre ProfessorInnen.

Kollektive Meinung
: Die Meinungen der einzelnen SchülerInnen anonym, als gesamte Meinung einer Klasse oder einer Gruppe von SchülerInnen, dargestellt.

Persönliche Konsequenzen für SchülerInnen
: Subjektive, meist unbewusste Entscheidungen von Lehrpersonen, welche von ihrer Haltung gegenüber der jeweiligen SchülerIn abhängen und deren Jahresnote beeinflussen können.

Um ein gegenseitiges Verständnis zu fördern, müssen Möglichkeiten zur anonymisierten Kommunikation zwischen SchülerInnen und deren Lehrpersonen geschaffen werden. LehrerInnen können dann Fragen, welche bereits zuvor von erfahrenen Feedbackbeauftragten erstellt wurden, an die gesamte Klasse stellen, welche anonym von jedem Schüler beantwortet werden können. Diese Form der Kommunikation kann auf mehrere Arten durchgeführt werden, welche im folgenden Text näher erläutert werden.

1.3 Verschiedene Feedbackmethoden

1.3.1 Zettel mit ankreuzbaren Antworten

Ausgedruckte Zettel, welche von der KlassenlehrerIn in einer Schulstunde ausgegeben werden, sind die traditionelle Herangehensweise. Diese Zettel können schnell und ohne großen Aufwand von den SchülerInnen, entweder während der Unterrichtseinheit, oder auch außerhalb des Unterrichts, ausgefüllt werden. Die LehrerInnen können diese schriftlichen Fragebögen daraufhin ohne große Statistikkenntnisse auswerten.

Diese Methode ist zwar auf den ersten Blick die Einfachste, bei näherer Betrachtung fallen allerdings vermehrt die Nachteile ins Auge:

- Die Auszählung der Ergebnisse ist mühsam und fehleranfällig,
- Fragebögen können schwerer verändert werden wenn sie bereits gedruckt wurden,
- und der Papierverbrauch ist bei umfangreicheren Fragebögen sehr hoch.

Aufgrund der schweren Veränderbarkeit werden diese Bögen meistens nicht genauer angepasst und stattdessen nur kopiert, womit aktuellere Themen ausgeblendet beziehungsweise unzureichend oder gar nicht erst behandelt werden.

1.3.2 Zettel mit Fragen und ausfüllbaren Antwortfeldern

Bögen mit abgedruckten Textboxen, in denen man eigene Antworten schreiben kann, werden oft bei umfangreicheren Fragen und für Vorschläge verwendet. Diese Art der Gestaltung hat zwar auf der einen Seite den Vorteil, dass Schüler ihre *eigenen* Gedanken ausdrücken können, ohne sich an vorgegebene Antwortmöglichkeiten halten zu müssen, gefährdet die Anonymität der Ergebnisse

allerdings stark. Die Fragebögen könnten zwar mit einer Schreibmaschine ausgefüllt werden, da diese Vorgehensweise allerdings zu aufwändig ist, ist sie im Schulalltag nur sehr selten anzutreffen.

Diese Bögen werden gerne mit ankreuzbaren Antworten kombiniert, um eine umfassende Meinungsäußerung zu ermöglichen. Durch die verlorene Anonymität aufgrund von handgeschriebenen Antworten, verlieren die Kreuze der vorhergehenden Fragen allerdings schnell ihre verschleiernde Wirkung. Außerdem ist diese Methode, genauso wie die Erste, sehr zeitaufwändig, da auch hier jeder Bogen einzeln abgearbeitet werden muss.

1.3.3 Online Feedback mit gemischten Typen von Fragen

Eine webbasierte Feedbacklösung löst einige Probleme der ersten 2 Methoden: Die Auswertung geschieht automatisch, Fragen können leicht geändert werden und es wird kein Papier verbraucht. Zusätzlich kommt noch der entscheidende Vorteil der Anonymität ins Spiel, welcher bei dieser Lösung stärker gegeben ist als bei den manuellen Verfahren, während gleichzeitig die Möglichkeit der individuellen Meinungsäußerung gegeben ist. Die hier beschriebene Form des Feedbacks richtet sich an die Kommunikation einer Gruppe von SchülerInnen an eine einzelne LehrerIn. Obwohl diese Form des Feedbacks zwar das angestrebte Ziel ist, da die Gruppe der SchülerInnen bis jetzt diejenige mit der leisesten Stimme war, können die hier vorgestellten Werte und Lösungen natürlich auch für Rückmeldungen von Eltern an den Lehrkörper herangezogen werden.

Ein Nachteil einer webbasierten Lösung des Problems ist die längere Zeit, welche benötigt wird, um ein solches System zu kreieren. Diese VWA umfasst allerdings genau dieses Problem der Erstellung und löst es im gleichen Atemzug, weshalb der Aufwand zur Programmierung eines vollautomatischen Systems für eine Schule, welche es lediglich verwenden will, vernachlässigbar wird. Die Möglichkeiten in der statistischen Auswertung erhobener Ergebnisse

überwiegen den Nachteil des Erstellungsaufwands ebenfalls um ein Vielfaches.

Durch die vielen Vorteile, welche ein webbasiertes Feedback-System bietet, ist ein solches am Besten geeignet, um die Bedürfnisse an den meisten Schulen gut zu erfüllen. Es ermöglicht eine einfache, schnelle und umfassende Auswertung der Ergebnisse, die Fragen können zentral verwaltet werden und es ist anonym benutzbar.

Um eventuelle Unklarheiten der verwendeten Begriffe zu vermeiden, folgt eine Auflistung der wichtigsten Ausdrücke zusammen mit den jeweiligen Bedeutungen.

Webbasiertes (Online-) Feedback
: Ein Fragebogen, welcher am Computer ausfüllbar ist und sich automatisch generiert.

Typen von Fragen
: Zum Beispiel ankreuzbare Antworten, Ja/Nein Fragen oder in einem kurzen Text beantwortbare Fragen.

1.4 Eigene Motivation für diese Arbeit

Als Schüler des Europagymnasium Auhof in Linz war ich in meiner fast achtjährigen Laufbahn vielen verschiedenen Situationen ausgesetzt, in denen ich mir etwas mehr Gewicht meiner Meinung und auch unserer Gesamtmeinung als Klasse gewünscht hätte. SchülerInnen werden, vor allem in niedrigeren Klassen, leider manchmal zu wenig ernst genommen und stehen am Ende immer unter ihrer jeweiligen Lehrperson. Wenn man als SchülerIn dann eine gute Idee hat oder einem auffällt, dass in der Klasse oder im Unterricht Probleme entstehen, hat man nur geringe Chancen, sich zu helfen. Jeder, der eine Schule besucht hat, wird vielleicht ähnliche Erlebnisse gemacht haben.

Da es mir die Informatik immer schon sehr angetan hat, lernte ich bereits früh, mit Computern Probleme zu lösen und meinen eigenen Weg umzusetzen, so wie ich es für das Beste hielt. Während ich in der 10. Schulstufe war, wurden die Stimmen nach einer eigenen Feedbacklösung für unsere Schule immer lauter. Da ich zu dieser Zeit Klassensprecher war, konnte ich mich direkt in den Prozess einbinden und die benötigten Grundkriterien festhalten. Während der 6. Klasse programmierte ich die Grundzüge eines Feedbacksystems und stellte es, Version für Version, unserer Schule zur Verfügung. Die Rückmeldungen fielen gut aus und bald wurde es zur offiziellen Methode an unserer Schule erklärt und verwendet. Nach dem Einsatz fuhr ich mit der Entwicklung des Systems bis in die 8. Klasse fort und stellte die Funktionen an vielen Stellen den verantwortlichen LehrerInnen vor, um ihr Verständnis zu stärken.

In dieser Arbeit will ich genauer auf die Entscheidungen zur inneren Funktionsweise des Systems eingehen und diese detaillierter erläutern, damit interessierte BenutzerInnen mehr Informationen erhalten und ProgrammiererInnen von eventuellen Nachfolgesystemen in diesem Bereich einen tieferen Einblick in meine Erkenntnisse während der Erstellungsphase bekommen können. Ein LehrerInnenfeedbacksystem kann die Qualität des Unterrichts an Schulen stark steigern, weshalb ich mich mit dieser Arbeit für derartige Systeme einsetzen möchte.

Kapitel 2

Planung eines neuen Feedback-Systems

Ein neues System zur Einholung von Feedback für viele verschiedene Unterrichtsgegenstände in Schulen bedarf einer längeren Planungsphase, um die benötigten Funktionen möglichst vollständig und zur größten Zufriedenheit aller Beteiligten zu implementieren. Dieses Kapitel beschreibt die Schritte der Planungsphase und die Überlegungen, welche in das System eingeflossen sind.

2.1 Evaluierung bereits existierender Systeme

Da das Thema des LehrerInnenfeedbacks in vielen Schulen aufgegriffen wird und sich viele LehrerInnen Rückmeldungen zu ihren Unterrichtspraktiken wünschen, gab es vor dieser Arbeit bereits einige funktionierende Methoden.

Die folgenden Informationen wurden von den jeweils angegebenen Websites eruiert und durch eigene Erfahrungen mit den jeweiligen Systemen ergänzt.

2.1.1 123Lehrer Feedback

123Lehrer Feedback[1] ist ein externes, webbasiertes Programm, welches speziell zum Einholen von Feedback zu LehrerInnen erstellt wurde. Die ErstellerIn einer Umfrage muss zuerst alle E-Mail Adressen der SchülerInnen eintragen, welche

[1] staffadvance GmbH. *123lehrerfeedback.de*. 2015. URL: https://www.123lehrerfeedback.de.

dann einen Link auf ihren jeweiligen Fragebogen bekommen. Die Seite gibt an, die Ergebnisse immer nur in anonymisierter Form anzuzeigen. Hier müsste allerdings entweder jede LehrerIn die E-Mail Adressen aller SchülerInnen jährlich einsammeln und aktuell halten, beziehungsweise diese Aufgabe an die KlassensprecherIn oder diese Verantwortung an Dritte übergeben. Dies bedeutet im Schulalltag zu viel Arbeit, weshalb sich dieses System nicht sehr gut für ein schulübergreifendes Feedback eignet.

2.1.2 Umfrage Online

Umfrage Online[2] ist ein webbasiertes Programm zum allgemeinen Durchführen und Analysieren von Umfragen. Dieses System kann durch etwas mehr Aufwand von ProfessorInnen dazu umfunktioniert werden, ihren eigenen Unterricht zu bewerten. Hier treten allerdings noch mehr Probleme als bei *123Lehrer Feedback* auf, da der Zeitaufwand zur Erstellung noch höher ist. Eine ganze Schule kann das eigene Feedback nicht komplett auf ein derartiges Angebot umstellen, da die benötigte Zeit zum Erstellen von sich wiederholenden Umfragen in den Augen Vieler in keinem Verhältnis zum Nutzen stehen würde.

2.1.3 schoolfeedback.at

Die Idee eines neuen Systems entstand bei der Verwendung des alten *Schoolfeedback.at*[3] Systems von Karl Hagenbuchner. Hier gab es die Möglichkeit, sich als LehrerIn zu registrieren und für die jeweilige Klasse einen vorgefertigten Fragebogen erstellen zu lassen. SchülerInnen mochten dieses System allerdings weniger, da man hier den Namen der ProfessorIn, der Schule und des Gegenstands eintragen musste, um zur Umfrage zu gelangen, und das System manchmal bei Überlastung nicht mehr erreichbar war. Leider ist die damalige

[2] enuvo GmbH. *Umfrage Online*. 2015. URL: https://www.umfrageonline.com.
[3] Karl Hagenbuchner. *Schoolfeedback.at*. 2014. URL: http://schoolfeedback.at/.

Website nicht mehr aktiv und wurde vom Netz genommen, daher kann man hier nicht mehr aktiv auf sie verweisen.

2.2 Geeignete Technologien für ein neues System

Eine Webanwendung besteht aus sehr vielen Komponenten aus verschiedenen Schichten. Solche Ebenen der Softwarearchitektur sind zum Beispiel die Anbindung an Datenbanken, das Absichern von Eingaben oder Softwarebibliotheken zum Zeitsparen während der Programmierung. Einige Schichten und Programmteile bereits als kostenlose und freie Software-Bibliotheken im Internet verfügbar, wodurch der Gesamtaufwand drastisch reduziert wird. Im folgenden Abschnitt werden die verwendeten Technologien aufgelistet und genauer beschrieben.

2.2.1 Speicherort und Datensicherheit

Da ein derartiges System teilweise sensible Daten speichert und Schulen häufig einen großen Wert auf Datensicherheit legen, ist es ratsam, das gesamte Programm auf dem jeweiligen Schulserver zu installieren. Mit diesem System bekommt jede Schule, welche sich eine Feedbacklösung wünscht, eine eigene Installation auf dem Schulserver (oder auf ihrem eduhi-Webspace, falls in der Schule kein eigener Server vorhanden ist), um Drittpersonen keinen Zugang zu den sensiblen Daten zu gewähren.

Die Ergebnisse der Umfragen werden somit nur schulintern verarbeitet und verlassen die Kontrolle der Schule zu keinem Zeitpunkt. Die einzigen Personen, die noch Zugriff auf die Ergebnisse haben, sind die AdministratorIn und BenutzerInnen mit Leseberechtigungen. Hierdurch wird verhindert, dass externe Personen auf die sensiblen Daten zugreifen können.

2.2.2 Serverseitige Programmiersprache

Die serverseitige Programmiersprache eines solchen Systems ist ebenfalls ein maßgeblicher Faktor. Sie muss weit verbreitet sein, bei vielen SchuladministratorInnen bekannt sein (damit kein Misstrauen um die Datensicherheit entsteht, siehe oben) und Programme, welche in dieser Sprache verfasst wurden, müssen möglichst schnell und einfach installierbar sein.

Aufgrund dieser Anforderungen wurde für dieses Projekt die Programmiersprache *PHP*[4] gewählt, da sie sehr weit verbreitet ist und (durch sehr viele beliebte Projekte wie zum Beispiel *WordPress*[5]) auch auf den meisten Webservern bereits installiert ist. Es ist für AdministratorInnen dadurch weniger Arbeit das Feedbacksystem in der eigenen Schule zu installieren. Dies führt gleichzeitig zu weniger Widerstand seitens des Lehrerkollegiums, was die Einführung des Systems in mehreren Schulen seitens der SchülerInnenvertretung stark vereinfacht.

Paketmanagement

Um die vielen benötigten Programmbibliotheken effektiv in einem einzelnen Projekt zu vereinen, haben sich in den letzten Jahren eigene Tools gebildet, die diese Aufgabe vereinfachen. Sie laden die benötigten Bibliotheken automatisch, aktualisieren diese auf Wunsch und binden sie in den eigenen Quelltext ohne viel unnötige Arbeit automatisch ein. Für PHP wurde hier der *Composer*[6] verwendet, da er der größte Paketmanager für *PHP* ist. Die folgende Datei ist die verwendete `composer.json` Datei des Systems.

```
1 {
2     "require": {
3         "anahkiasen/underscore-php" : "dev-master",
```

[4] The PHP Group. *PHP: Hypertext Preprocessor*. 2015. URL: https://secure.php.net/.
[5] WordPress Organisation. *WordPress: Blog Tool, Publishing Platform, and CMS*. 2015. URL: https://wordpress.org/.
[6] Composer Project. *Composer*. 2015. URL: https://getcomposer.org/.

```
4         "j4mie/idiorm": "dev-master",
5         "ezyang/htmlpurifier": "dev-master",
6         "Respect/Validation": "dev-master",
7         "dapphp/securimage": "dev-master",
8         "filp/whoops": "1.*",
9         "ircmaxell/password-compat": "dev-master",
10        "robmorgan/phinx": "0.4.*"
11    },
12    "autoload": {
13        "psr-4": {"App\\": "app/includes/"}
14    }
15 }
```

Listing 1: `composer.json` des Feedback Systems

2.2.3 Serverseitige Software-Bibliotheken

underscore.php

Underscore.js[7] ist eine Sammlung von nützlichen Funktionen, um die Formulierung komplexer Abläufe zu vereinfachen. Die Bibliothek bietet zum Beispiel Sortierfunktionen, Filterfunktionen und Templates.

Idiorm

Idiorm[8] stellt einen objektorientierten Zugang für verschiedene, relationale Datenbanksysteme zu Verfügung. Durch diesen Wrapper um die einzelnen Datenbanken, kann der benötigte Code zum Ansprechen der verschiedenen Systeme auf ein Minimum reduziert werden und es muss nichts doppelt geschrieben werden. Außerdem bringt *Idiorm* noch weitere Funktionen, wie zum Beispiel

7 Underscore Contributers. *Underscore.js*. 2015. URL: `http://underscorejs.org/`.
8 Jamie Mathews. *Idiorm & Paris*. 2015. URL: `https://j4mie.github.io/idiormandparis/`.

eine einfache Manipulation von größeren Datenmengen, schnelles Iterieren einzelner Datensätze und einen Schutz vor SQL-Injections, bereit. (Informationen und Definitionen über Datenbankabstraktionen aus[9])

HTML Purifier

HTML Purifier[10] ist eine Bibliothek zum Entfernen unerwünschter HTML-Tags aus eingegebenen Texten. Da das Feedback-System es den Nutzern ermöglicht, eigene Texte einzugeben, ist die Gefahr unerwünschter Elemente sehr groß.[11] Diese können von eingebundenen Bildern bis hin zu nachgeladenem Programmcode reichen. *HTML Purifier* entfernt alle schädlichen Tags auf der Seite des Servers und verhindert dadurch das Speichern von schädlichen Daten.

Validation

Validation[12] liefert eine umfassende Umgebung zur Validierung von Nutzereingaben. In dieser Anwendung wird es verwendet, um genauere Validierungen der Nutzereingaben (und der Ergebnisse) zu ermöglichen.

Securimage

Securimage[13] implementiert einen CAPTCHA, um eventuellen Bots und Hacker-Innen die Anmeldung zu erschweren. Dieses CAPTCHA bietet zwar keinen absoluten Schutz gegen einen gezielten Angriff, gegen allgemeine Angriffe und unerfahrene Jugendliche ist dieses Schutz aber mehr als ausreichend. (CAPTCHA Erklärung von[14])

---

9 Tobias Wassermann. *Sichere Webanwendungen mit PHP*. Erste Ausgabe. mitp, 2007. ISBN: 9783826617546, S. 90.
10 HTML Purifier Community. *HTML Purifier*. 2015. URL: http://htmlpurifier.org/.
11 Wassermann, *Sichere Webanwendungen mit PHP*, S. 86.
12 Alexandre Gomes Gaigalas. *Respect Validation*. 2015. URL: https://respect.github.io/Validation/.
13 Drew Phillips. *Securimage PHP Captcha*. 2015. URL: https://www.phpcaptcha.org/.
14 Wassermann, *Sichere Webanwendungen mit PHP*, S. 192.

Whoops

Whoops[15] ist eine PHP-Bibliothek für verbesserte Fehlermeldungen und erleichterte Fehlersuche. Da die normalen Fehlerausgaben in PHP nur wenige Informationen beinhalten und schwer lesbar sind, stellen die schöneren und informativeren Ausgaben von *whoops* eine große Verbesserung dar. Es wird genau ausgegeben, welcher Fehler in welchem Programmteil hervorgerufen wurde, und wie der Verlauf des Scripts war, bevor der Fehler passierte.

Um eventuelle Leistungseinbußen zu vermeiden, ist *whoops* nur im Entwicklungsmodus aktiviert und während des normalen Betriebs ausgeschaltet. Dieses Verhalten lässt sich in einer Konfigurationsdatei genauer einstellen.

Password-Compat

Password-Compat[16] stellt die *password_**[17]-Funktionen, welche normalerweise erst ab PHP Version 5.5 oder höher verfügbar sind, schon ab Version 5.3.7 bereit. Diese Funktionen dienen dazu, die Passwörter der BenutzerInnen in nicht lesbare, aufwändige Hashes zu verwandeln, anstatt sie im Klartext abzuspeichern. Sobald sich die NutzerIn danach anmeldet, wird sein eingegebenes Passwort gegen den gespeicherten Hash gerechnet und falls das eingegebene Passwort den gleichen Hash ergibt, ist es korrekt. Ein Beispiel für so einen Hash wäre:

```
$2y$10$.vGA1O9wmRjrwAVXD98HNOgsNpDczlqm3Jq7KnEd1rVAGv3Fykk1a
```

Durch die eigenen Algorithmen dieser Funktionen wird sichergestellt, dass die gespeicherten Passwörter nicht mehr ausgelesen werden können, da ein Angriff auf die starken Hashes zu viel Zeit beanspruchen würde. Dadurch ist die Sicherheit der Passwörter der Nutzer garantiert, sogar falls die AdministratorIn versuchen würde, sie zu „knacken“.

---

15 Filp. *whoops!* 2015. URL: https://filp.github.io/whoops/.

16 ircmaxwell. *password_compat*. 2015. URL: https://github.com/ircmaxell/password_compat.

17 The PHP Group. *PHP: Password Hashing - Manual*. 2015. URL: https://secure.php.net/password.

Diese Bibliothek ist vor allem für ältere Schulserver sehr wichtig, da für dieses System alleine kein Update des Servers benötigt werden sollte.

Phinx

Phinx[18] ist ein Werkzeug zum Verwalten von Datenbanken. Mithilfe dieses Programms kann man Datenbankschemas sehr schnell updaten und neue Versionen einer Anwendung automatisch installieren lassen. Außerdem kann hiermit auf eine deterministische Weise immer der gleiche Datenbankzustand mit derselben Struktur hergestellt werden, mit der das System auch entwickelt wurde. So können Fehler durch falsche Datenbankinstallationen oder abgebrochene Aktualisierungen behoben werden.

Einbinden der einzelnen Bibliotheken

Die oben genannten Bibliotheken werden in der `composer.json` Datei definiert, von *Composer* heruntergeladen und direkt in den Namespace der Applikation eingebunden. Die selbstgeschriebenen Programmteile werden danach ebenfalls von *Composer* verwaltet, um das Einbinden jedes Teils in anderen Dateien zu ermöglichen. Dies verringert den benötigten Code enorm, da man sich nicht mehr um die physische Strukturierung der einzelnen Dateien kümmern muss.

2.2.4 Frontend Framework / Seitendesign

Ein Frontend-Framework ist eine Sammlung von oft verwendeten Befehlen zur Gestaltung des Aussehens von Seiten[19]. Mit solchen Frameworks kann man die Seitenpräsentation (Schriften, allgemeines Layout, Buttons, Tabellen, ...) um einiges schneller erstellen, wodurch man mehr Zeit für das Backend, also die Datenverarbeitung im Hintergrund hat. Sie eignen sich daher sehr gut

[18] Rob Morgan. *Phinx*. 2015. URL: `https://phinx.org/`.

[19] Manuela Hoffmann. *Modernes Webdesign: Gestaltungsprinzipien, Webstandards, Praxis*. Zweite Ausgabe, 1. Überarbeitung. Galileo Design, 2010. ISBN: 9783836215022, S. 114.

für Systeme, deren Hauptzweck die Datenverarbeitung ist und allgemein eher mehr angewandt, als nur angesehen werden.

Da das Feedbacksystem sich eher auf die Datenverarbeitung als auf eine extravagante Präsentation fokussiert, wird in diesem System *Bootstrap*[20] in der Version 3 verwendet. Dieses Framework bietet viele vorgefertigte Elemente, von Buttons, über eigene Meldungsboxen, bis zu eigenen Fenstern und sich öffnenden Dialogen. Eine genaue Liste findet sich auf der Website des Projekts.

Dieses Framework bietet außerdem gute Möglichkeiten für die Verwendung in Fragebögen und anderen, auf Daten bezogenen Elementen. Ein gutes Beispiel für die Vorteile, die *Bootstrap* mit sich bringt, ist das Design des Feedbackbogens, welches in Abbildung 1 betrachtet werden kann.

2.2.5 Grafische Präsentation der Daten

Die grafische Präsentation der erhobenen Feedbackdaten kann entweder bereits am Server, oder im Webbrowser der NutzerIn selbst erfolgen. In dieser Arbeit wird die Methode der Erstellung auf der Seite der NutzerIn bevorzugt, da hier weniger Rechenkraft am Server benötigt wird und dadurch mehr Zugriffe mit weniger Rechenkraft abgearbeitet werden können.

Um eine browserübergreifend gute Darstellung sicherzustellen, wird in dieser VWA auf die *Chart.js*[21] Bibliothek zurückgegriffen. Diese Software läuft im Browser der NutzerIn und berechnet aus den eingefügten Daten Diagramme, welche sich genau an die jeweiligen Gegebenheiten (Fenstergröße, Schriften, ...) anpassen.

[20] Twitter & Contributors. *Bootstrap*. 2015. URL: http://getbootstrap.com/.
[21] Nick Downie. *Chart.js*. 2015. URL: http://www.chartjs.org/.

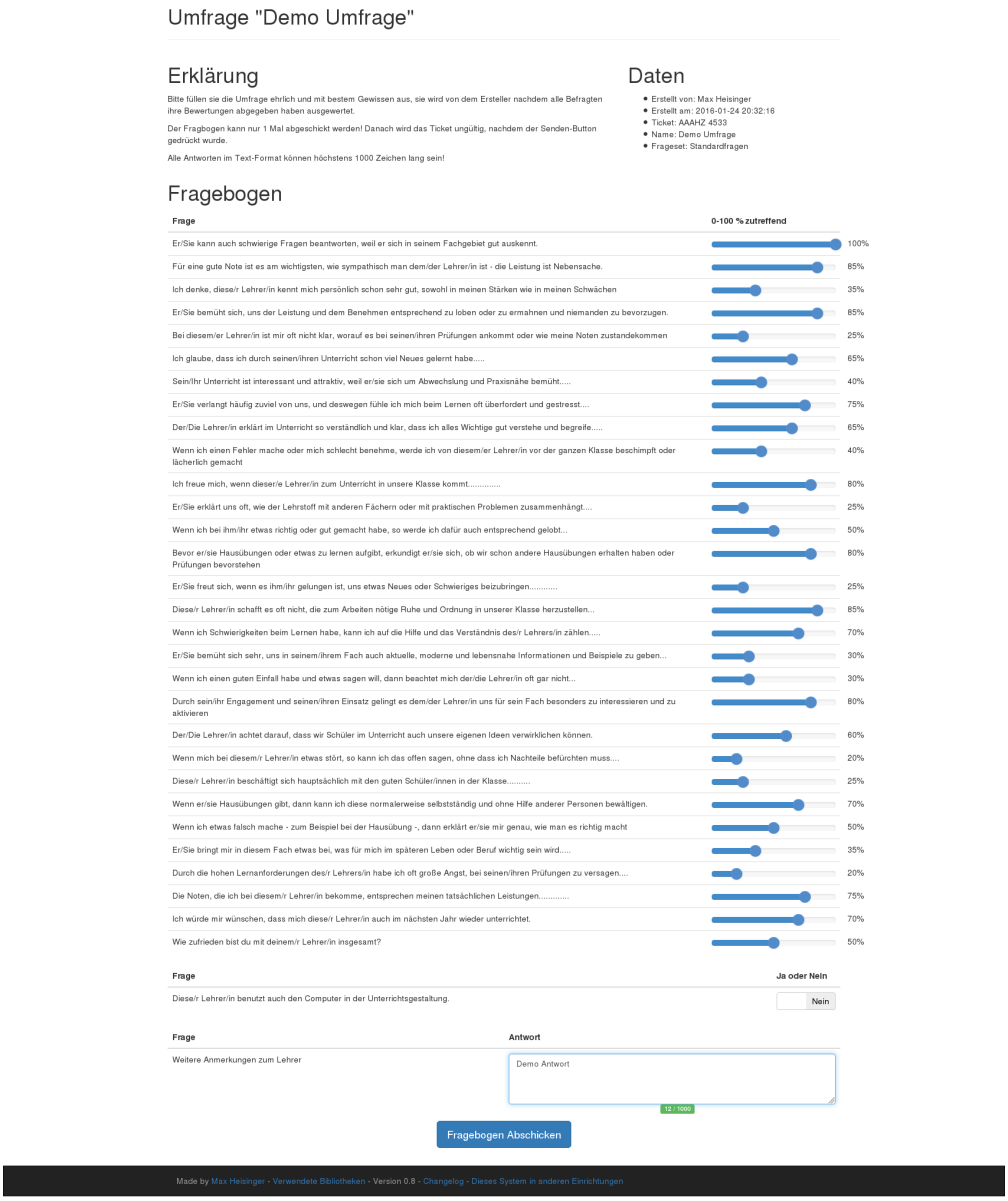

Abbildung 1: Ausfüllen eines Fragebogens im fertigen System

2.2.6 Zusammenfassung der technischen Entscheidungen

Um einen besseren Überblick zu ermöglichen, werden nachtragend nochmal alle technischen Entscheidungen in einer Liste zusammengefasst:

1. Hosting: In jeder Schule einzeln (Datensicherheit)
2. Programmiersprache: *PHP*
3. Paketmanagement: *Composer*
4. Frontend-Framework: *Bootstrap 3*
5. Routing: Kein Framework, PHP Files geben direkt die Seiten aus, ohne weitere Konfigurationen zu benötigen.
6. Datenbank: ORM zur Datenbankabstrahierung mit *Idiorm* auf *PDO*-Basis

2.3 Definition des erforderlichen Funktionsumfangs

Nach Gesprächen mit einigen ProfessorInnen und dem damaligen Schulsprecher, ließ sich der gewünschte Funktionsumfang gut definieren. Dieser ist in mehreren Stufen aufgebaut, damit zwischen den einzelnen Entwicklungsschritten jeweils neue Meinungen eingeholt werden konnten.

2.3.1 BenutzerInnenverwaltung

Das System soll ein eigenes Verzeichnis der registrierten BenutzerInnen in der verwendeten Datenbank mitführen, damit es unabhängig von Änderungen in der restlichen IT-Landschaft der Schule ist. Hierzu kommt noch, dass eine eigene BenutzerInnenverwaltung die Installation und Wartung stark vereinfacht, da man als AdministratorIn keine Anbindung an die in der Schule verwendete Software erstellen muss.

Ein Nachteil dieser eigenen BenutzerInnenverwaltung ist zwar, dass ProfessorInnen sich einen separaten Account anlegen müssen, um Feedbackbögen zu

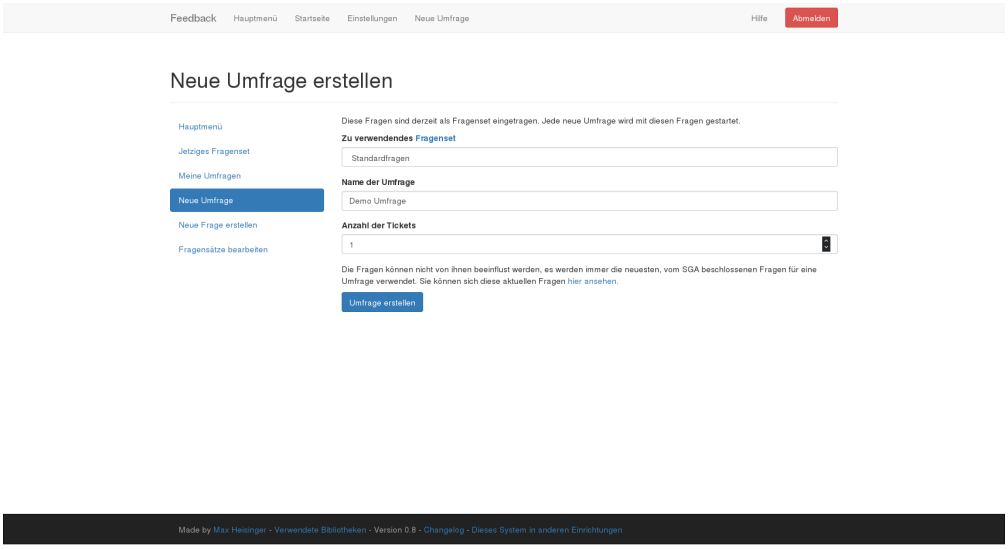

Abbildung 2: Erstellen einer neuen Umfrage im fertigen System

erstellen, statt ihren bereits vorhandenen Account im Schulsystem zu verwenden. Da allerdings nicht alle Schulen ein eigenes System zum Verwalten von BenutzerInnenaccounts haben und manche ProfessorInnen sich lieber aussuchen, in welchen Systemen ihre persönlichen Daten landen, wiegt dieser Nachteil nicht so schwer. Außerdem ermöglicht diese strikte Separierung des Systeme, dass Daten von SchülerInnen in keiner Weise mit dem gegebenen Feedback in Verbindung gebracht werden können, wodurch eine gute Anonymität für die SchülerInnen gewährleistet wird.

Die technische Implementierung der BenutzerInnenverwaltung kann im Kapitel 3.1.1 nachgeschlagen werden.

2.3.2 Umfragen

Es sollen sich Umfragen erstellen lassen, welche automatisch mit einem von der zuständigen Instanz erstellten Fragebogen ausgestattet werden. Durch die Generierung von *Tickets* (kurze alphanumerische Zeichenketten als Referenz auf ein bestimmtes Objekt oder einen bestimmten Zugang, in diesem Fall wurden 5-stellige Buchstabenkombinationen gewählt), kann man die Umfrage

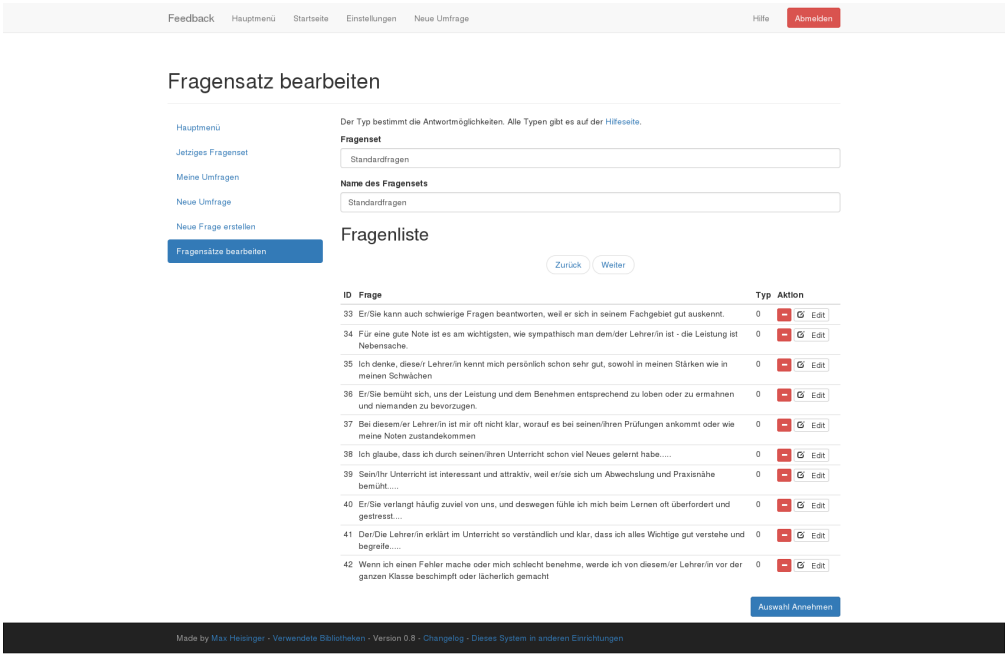

Abbildung 3: Bearbeiten eines Fragebogens im fertigen System

an TeilnehmerInnen weitergeben. Diese können mithilfe dieser Codes ohne weitere Anforderungen auf die Umfrage zugreifen und die ausgefüllten Daten an die ErstellerIn zurückschicken, ohne sich selber noch einen Account anlegen zu müssen.

2.3.3 Variable Fragebögen

Die Fragebögen, welche bei der Erstellung von Umfragen automatisch verwendet werden, sollen für mehrere Unterrichtsgegenstände und Fächergruppen anpassbar sein. Eine UmfragenerstellerIn soll bei der Erstellung dann die Möglichkeit haben, den für sein Fach am besten geeignetsten Fragebogen auszuwählen.

Diese verschiedenen Bögen müssen für die Verantwortlichen möglichst leicht änderbar sein, damit auch BenutzerInnen ohne tieferes technisches Wissen Änderungen durchführen können. Einige Vorschläge für die idealen Verantwortlichen wären zum Beispiel:

- der Schulgemeinschaftsausschuss (SGA),
- Fachgruppen (wie beispielsweise MathematikprofessorInnen, PhysikprofessorInnen, ...),
- oder interessierte ProfessorInnen.

2.3.4 Auswertung der Ergebnisse

Die Ergebnisse der einzelnen Umfragen müssen sich leicht verständlich auswerten lassen. Dafür bietet sich eine grafische Auswertungsmethode sehr gut an.

Grafische Auswertung

Eine grafische Auswertung basiert auf Diagrammen, welche automatisch aus den Ergebnissen generiert werden und auf einen Blick die Aufteilung der Meinungen anzeigen.

Die erhaltenen Ergebnisse werden in diesem Verfahren pro Frage in jeweils angepassten Graphen angezeigt, die sich in ihrer Präsentation zwischen den jeweiligen Fragetypen unterscheiden.

Verschiedene Fragetypen

- Typ-0 (0-100% Fragen): Der Bewertungsgrad liegt auf der X-Achse, während die Anzahl der Bewertungen auf der Y-Achse liegt. Die einzelnen Datenpunkte werden durch eine geschwungene Linie verbunden.
- Typ-1 (Ja/Nein Fragen): Hier wird ein Tortendiagramm verwendet. Die negativen Antworten liegen hier im roten Bereich, die positiven im grünen.
- Typ-2 (Volltext-Fragen): Die Antworten werden untereinander aufgelistet.

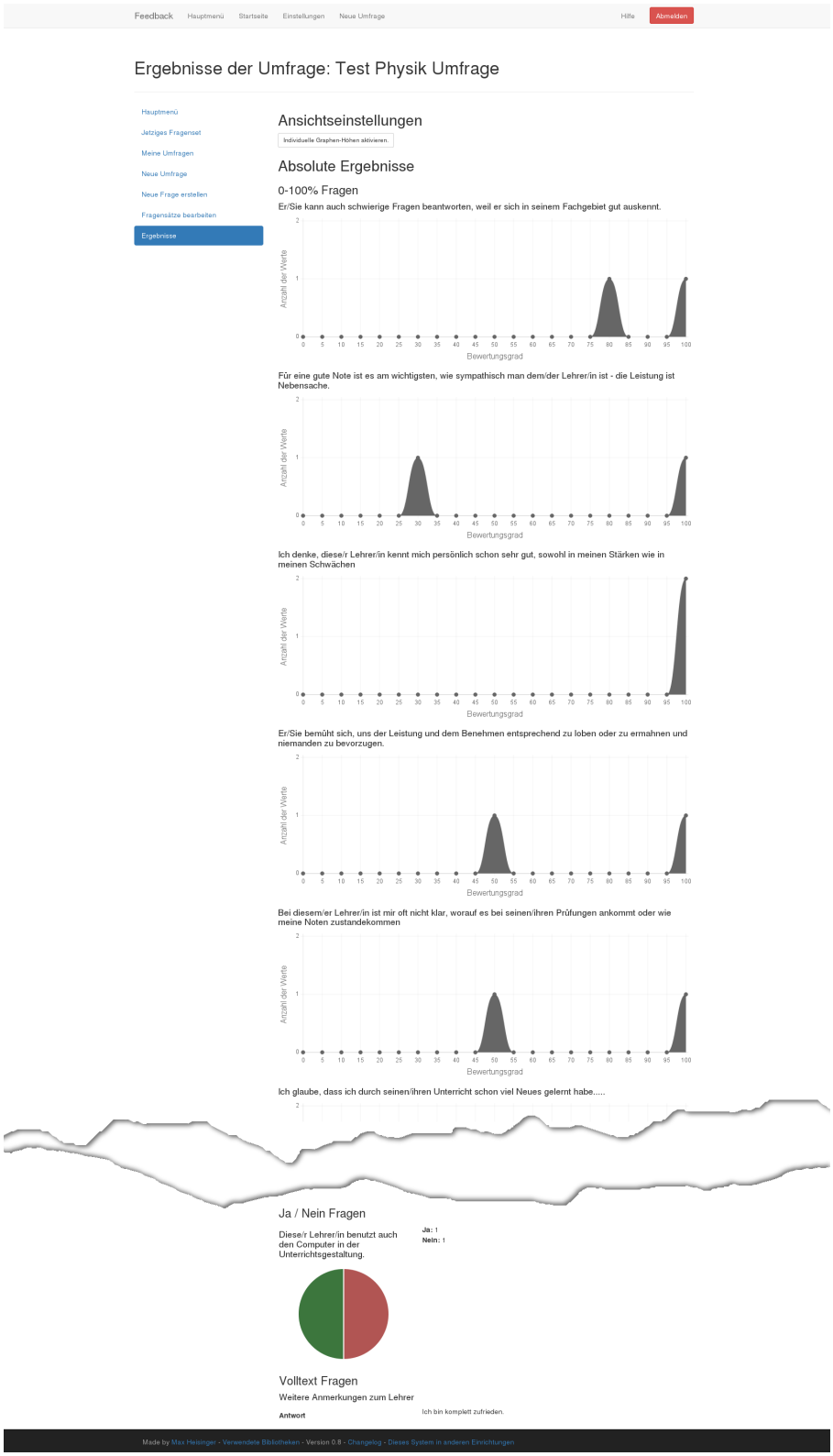

Abbildung 4: Auswerten eines abgeschlossenen Fragebogens im fertigen System (simulierte Ergebnisse, Bildschirmfoto wurde etwas verkürzt)

Die verwendeten Fragetypen, die einfache Gestaltung und die allgemeine, systematische Methode des verwendeten Feedbackbogens, beziehen sich auf[22].

2.3.5 Zusammenfassung der benötigten technischen Merkmale

Um einen schnellen Überblick zu ermöglichen, folgt eine Liste aller geforderten technischen Merkmale.

- Einfache Benutzeroberfläche, die sehr leicht zugänglich ist und die NutzerIn unterstützt, anstatt gegen sie zu arbeiten.
- Schnelle Verarbeitung der Anfragen, um ein möglichst reibungsloses Arbeitserlebnis zu liefern.
- Eine eigens eingebaute BenutzerInnenverwaltung, um den Installationsaufwand für AdministratorInnen zu reduzieren.
- Ein breites Angebot an Konfigurationsmöglichkeiten und verschiedenen Fragetypen.
- Ein schneller Workflow während des Eintragens von Feedback in Feedbackbögen.
- Ein einfacher Installations- und Aktualisierungsvorgang für SchuladministratorInnen, um das Einsetzen des Systems in möglichst vielen Bildungseinrichtungen zu ermöglichen, ohne den Arbeitsaufwand signifikant zu erhöhen.

[22] Johannes Bastian, Arno Combe und Roman Langer. *Feedback-Methoden - Erprobte Konzepte, evaluierte Erfahrungen*. neu ausgestattete Sonderausgabe. Beltz Verlag, 2007. ISBN: 9783407254689.

Kapitel 3

Technische Implementierung eines neuen Feedback Systems

3.1 Datenspeicherung

Die gesamte Datenspeicherung läuft über eine Datenbank, welche über die *Idiorm*-Bibliothek von verschiedenen Datenbanksystemen verwaltet werden kann. Zur Speicherung der Daten in einer relationalen Form sind einige Regeln und Abläufe notwendig, welche in dem folgenden Abschnitt näher behandelt werden.

Sämtliche Daten werden vor der Verwendung in eigenen Programmkomponenten verarbeitet, um sie für den Einsatz besser geeignet zu machen. Diese sogenannten *Wrapper* verhindern somit mehrfach geschriebenen Code und verringern damit sowohl den gesamten Programmieraufwand, als auch die möglichen Fehlerquellen. Außerdem entlasten sie die Datenbank, da sie häufig benötigte Daten, wie zum Beispiel den Nutzernamen, welcher auf jeder Seite angezeigt wird, direkt in der PHP-Session zwischenspeichern. Weiters verwenden sie für Datenbankzugriffe möglichst wenig Datenbankaufrufe und auch nur eine einzelne Datenbankverbindung pro Nutzer.

3.1.1 BenutzerInnenaccounts

BenutzerInnenaccounts werden in einer Tabelle namens `users` gespeichert. Die Passwörter der einzelnen User werden über einen Algorithmus von PHP in einen Hash gespeichert, was sie für eventuelle Angriffe nicht mehr auslesbar macht. Die verschiedenen Rechte der NutzerInnen werden über einen JavaScript Object Notation (JSON)-String in der Datenbank festgelegt, um eine dynamische Rechtevergabe und schnelles Auslesen der einzelnen Berechtigungen zu ermöglichen. Zwei Beispiele für verschiedene Rechtelevels folgen.

Rechte einer AdministratorIn

```
{
    "edit_questionset": true,
    "create_question": true,
    "rateable": true,
    "edit_question": true
}
```

Listing 2: Rechte einer AdministratorIn

Rechte einer BenutzerIn

```
{
    "edit_questionset": false,
    "create_question": false,
    "rateable": true,
    "edit_question": false
}
```

Listing 3: Rechte einer bewertbaren BenutzerIn

Unterschiede zwischen den Rechten

Die einzelnen Einträge in den Code-Ausschnitten beschreiben die jeweiligen Rechte der NutzerInnen. Die Bedeutungen der Einträge folgen.

edit_questionset

Die BenutzerIn kann bestehende Fragensets bearbeiten und neue Sets erstellen.

create_question

Die BenutzerIn kann neue Fragensets erstellen.

rateable

Die BenutzerIn kann eigene Umfragen erstellen.

edit_question

Die BenutzerIn kann bereits erstellte und verwendete Fragen bearbeiten und umschreiben.

Die Tabellenstruktur der `users`-Tabelle kann im Appendix in A unter 3 gefunden werden.

3.1.2 Tickets

Tickets werden nur einmal abgespeichert und dann bei jeder neuen Umfrage auf die ID der jeweiligen Neuen gesetzt. Sobald eine UmfrageteilnehmerIn ein Ticket auf der Startseite eingibt, wird der fünfstellige Buchstabencode (zum Beispiel `ABCDE`) mit der internen Datenbank verglichen. Falls geschützte Tickets in der Konfiguration aktiviert wurden, wird der vierstellige Zahlencode, welcher am Ticket hängt, ebenfalls kontrolliert.

Wenn die Buchstabenkombination mit einem Eintrag in der Datenbank übereinstimmt und auch der Zugangscode korrekt ist, wird die NutzerIn auf ihre Umfrageseite weitergeleitet, ohne dass weitere Eingaben nötig sind.

Die Tabellenstruktur der `tickets`-Tabelle kann im Appendix in A unter 4 gefunden werden.

3.1.3 Ergebnisse

Die Ergebnisse aller Umfragen werden gesammelt in einer einzelnen Tabelle gespeichert. Um die Anonymität der UmfrageteilnehmerInnen zu gewährleisten, enthalten sie keine persönlichen Daten.

Sämtliche Daten werden in JSON-Strings gespeichert, um die generierten Ergebnisse unabhängig von der Tabellenstruktur zu halten.

Die Tabellenstruktur der `results`-Tabelle kann im Appendix in A unter 5 gefunden werden.

3.1.4 Fragen

Die einzelnen Fragen werden in eigenen Tabellenspalten gespeichert und in den jeweiligen Fragensets zu Sammlungen kombiniert. Diese individuelle Speicherung ermöglicht dynamische Änderungen der Texte, selbst nachdem diese bereits in Umfragen verwendet werden, um eventuelle Ungenauigkeiten auch im Nachhinein ausbessern zu können.

Die verschiedenen Fragetypen werden neben dem Text in der Datenbank gespeichert und haben während dem Ausfüllen des Fragebogens eine Auswirkung auf die verwendeten Eingabemethoden. Die Tabelle 1 zeigt die standardmäßig implementierten Typen. Eine genauere Darstellung der verschiedenen Formate befindet sich in 2.3.4.

Die Tabellenstruktur der `questions`-Tabelle kann im Appendix in A unter 6 gefunden werden.

#	Name des Typs	Beschreibung
0	0-100% Fragen	Ein Balken mit 0-100% Werten, mit 5% Schritten.
1	Ja/Nein Fragen	Ein Schalter zum Auswählen zwischen Ja & Nein.
2	Volltext-Fragen	Eine Textbox zum Eintragen eigener Texte.

Tabelle 1: Fragetypen

3.1.5 Fragensets

Ein Fragenset beschreibt eine spezifische Sammlung einzelner Fragen, welche kombiniert in einem JSON-String in der Datenbank abgespeichert wird. Sobald eine Umfrage erstellt wird, werden die Fragen aus dem Fragenset kopiert und in den Datenbankeintrag der Umfrage geschrieben. Dies verhindert eventuelle Extremfälle, in welchen ein bereits verwendetes Fragenset während einer laufenden Umfrage verändert wird, und die Ergebnisse der einzelnen Teilnehmer nicht übereinstimmen würden.

Die Tabellenstruktur der `questionsets`-Tabelle kann im Appendix in A unter 7 gefunden werden.

3.1.6 Umfragen

Jede neue Umfrage wird mit einem neuen Eintrag in die Datenbank geschrieben. Umfragen speichern dabei von sich aus keine Tickets oder Ergebnisse, diese werden in der Tickets- (3.1.2) und in der Ergebnisse-Tabelle (3.1.3) gespeichert. Das verwendete Fragenset wird beim Erstellen der Umfrage ausgelesen und als einzelne Fragen-IDs in die Tabelle gespeichert, um eventuelle Fehler bei einer Änderung des Fragensets zu verhindern.

Die Tabellenstruktur der `surveys`-Tabelle kann im Appendix in A unter 8 gefunden werden.

#	Name des Typs	Beschreibung
0	`Notice`	Ein kleiner Hinweis an die NutzerIn.
1	`Success`	Eine Erfolgsmeldung (z.B. eine abgeschlossene Umfrage).
2	`Important`	Eine wichtige (mehr als ein Hinweis) Meldung.
3	`Critical`	Eine sehr wichtige Information.
4	`Warning`	Eine Warnung. Sollte dringend beachtet werden.

Tabelle 2: News-Typen

3.1.7 Mitteilungen

Mitteilungen sind kleine Update-Meldungen, wie zum Beispiel Informationen über abgeschlossene Umfragen oder Nachrichten der AdministratorIn. Es gibt hierbei mehrere Typen, welche im Code in einer Aufzählung (Enum), definiert in `app/includes/NewsType.php`, festgelegt werden.

Die standardmäßig implementierten Typen können mit den dazugehörigen Beschreibungen in der Tabelle 2 gefunden werden.

Die Tabellenstruktur der `news`-Tabelle kann im Appendix in A unter 9 gefunden werden.

3.2 Konfigurierbarkeit für AdministratorInnen

Ein System mit viel Kontakt zu verschiedenen Zielgruppen und häufig wechselnden Verantwortlichen, muss über eine gute Konfigurierbarkeit verfügen, um auch den unterschiedlichen Wünschen und Anforderungen der nachfolgenden NutzerInnen ohne Programmieraufwand gerecht werden zu können. Hierfür wurde in diesem System eine statische Konfigurations-Klasse verwendet, welche in jedem Programmteil aufgerufen werden kann und die ge-

wählten Optionen ohne zusätzliche Wartezeit zurückgibt. Diese Klasse ist in `app/ioncludes/Config.php` definiert.

Die Einstellungsmöglichkeiten sind in der zentralen `app/bootstrap.php` zu finden, welche bei jedem Seitenaufruf geladen werden. Diese Konfigurationsdatei ist bereits als Beispiel in der Software-Distribution mitgeliefert und sieht wie folgt aus:

```
1  <?php
2
3  /* Copyright (C) Max Heisinger - All Rights Reserved
4   * Unauthorized copying of this file, via any medium is
        strictly prohibited
5   * Proprietary and confidential
6   * Written by Max Heisinger <mail@maximaximal.com>, July 2014
7   */
8
9  \define("APP_DIR", \dirname(__FILE__)."");
10 \define("ASSETS_DIR", APP_DIR."static/");
11 \define("INCLUDES", APP_DIR."/includes/");
12 \define("VIEWS", APP_DIR."/views/");
13 \define("BASE_DIR", APP_DIR."/../");
14
15 require_once BASE_DIR.'vendor/autoload.php';
16 require_once INCLUDES.'Config.php';
17
18 use App\Config;
19
20 /**
21  * Setup the configuration.
22  */
23
24 date_default_timezone_set("Europe/Vienna");
```

```
25
26 //Debug
27 Config::set("DEBUG", true);
28
29 //Database
30 Config::set("DB_HOST", 'localhost');
31 Config::set("DB_USER", 'test');
32 Config::set("DB_PASS", 'test');
33 Config::set("DB_NAME", 'FeedbackSystem');
34 Config::set("DB_PORT", '3306');
35
36 //Miscellanous variables
37 Config::set("SITE_TITLE", "Online Professoren Feedback");
38 Config::set("TICKETS_REMAINING_FOR_SECONDS", (5 * 60));
39 Config::set("SECURELY_KEEP_SURVEYS_FOR_DAYS", 0);
40 Config::set("MAX_TICKETS_PER_SURVEY", 50);
41
42 //Start the session.
43 session_start();
44
45 //Setup the database.
46 ORM::configure('mysql:host='.Config::get("DB_HOST").';dbname='.
      Config::get("DB_NAME"));
47 ORM::configure('username', Config::get("DB_USER"));
48 ORM::configure('password', Config::get("DB_PASS"));
49
50 /**
51  * Setup the whoops error reports..
52  */
53 use Whoops\Handler\PrettyPageHandler;
54 if(Config::get("DEBUG") == true)
55 {
```

```
56      $run     = new Whoops\Run;
57      $handler = new PrettyPageHandler;
58      $run->pushHandler($handler);
59      $run->register();
60  }
```

Listing 4: Beispiel für eine `app/bootstrap.php` Datei

Es folgt eine genauere Beschreibung der einzelnen Bedeutungen der Konfigurations-Variablen:

DEBUG

Aktiviert den Debugging-Modus. Falls aktiv, werden Fehlermeldungen mit Whoops[23] ausgegeben. Ansonsten erfolgt die Ausgabe, so wie sie in PHP definiert wurde.

DB_HOST

Der Datenbankserver, der verwendet werden soll.

DB_USER

Der Benutzername, der bei der Datenbankverbindung verwendet werden soll.

DB_PASS

Das Password, das bei der Verbindung zur Datenbank verwendet werden soll.

DB_NAME

Der Name der Datenbank, in der die Daten gespeichert werden sollen.

DB_PORT

Der Port, unter welchem der Datenbankserver erreichbar ist.

SITE_TITLE

Der Seitentitel, welcher auf der Hauptseite und im Browserfenster verwendet werden soll.

[23] Filp, *whoops!*

TICKETS_REMAINING_FOR_SECONDS

Die Zeit (in Sekunden), wie lange Tickets nicht nochmals in Umfragen verwendet werden können, nachdem sie gerade benutzt wurden, um einen Fragebogen abzuschicken.

SECURELY_KEEP_SURVEYS_FOR_DAYS

Die Anzahl der Tage, wie lange eine Umfrage nach dem Abschließen nicht gelöscht werden kann. Dies ist nützlich, um eventuelle Irrtümer zu vermeiden.

MAX_TICKETS_PER_SURVEY

Die höchste Menge von TeilnehmerInnen bei einer einzelnen Umfrage.

3.3 Präsentation der Ergebnisse

Um die Daten in den Browser der NutzerIn zu bringen, wird in diesem System auf die serverseitige Generierung von JSON-Arrays zurückgegriffen, welche während dem Seitenaufruf in den Quelltext der Seite eingefügt werden. Die *Chart.js*[24]-Bibliothek liest diese Daten danach aus und generiert daraus grafische Diagramme.

3.4 Testen des Systems

Da Softwareprojekte über viele verschiedene Komponenten verfügen, können im Zusammenspiel der einzelnen Teile schnell Fehler auftreten, die in sogenannten *Integration Tests* ausfindig gemacht werden müssen. Hierfür gibt es verschiedene Methoden und Werkzeuge, um sowohl die typischen Problemzonen von Webapplikationen, als auch die individuellen Fehlerquellen zu finden.

[24] Downie, *Chart.js*.

3.4.1 Funktionalität

Zum Testen der Funktionalität des Systems, müssen alle Funktionen von mehreren NutzerInnen geprüft werden. Es ist sehr wichtig, das Programm von mehr als nur einer einzelnen Testperson ausprobieren zu lassen, da verschiedene Menschen immer verschiedene Ansätze haben, ihre individuellen Ziele zu erreichen.

Die kritischsten Stellen sind Eingabefelder für Nutzer (hier: Fragebögen), Auswahlfelder (zum Beispiel die Erstellung von Umfragen) und komplizierte Menüs. Hierbei ist nicht nur die Gefahr einer falschen Speicherung der Eingaben zu beachten, sondern auch die einfache Verständlichkeit für die zukünftigen NutzerInnen. Besonders die Eingabe von Daten in den Fragebogen und das Ändern oder Erstellen von Fragebögen fiel den Testpersonen schwer, daher wurden die jeweiligen User-Interfaces um erklärende Elemente erweitert. Vor allem kleine Erklärungen für die einzelnen Fragetypen in Listen, als auch die Anzeige des eingestellten Prozentwerts neben den Schiebereglern während dem Ausfüllen von Fragebögen, trugen maßgeblich zu einer besseren Nutzererfahrung bei.

3.4.2 Ladezeiten

Lange Ladezeiten, also lange Wartezeiten bis die gewünschten Aktionen geschehen oder die geforderten Daten auf dem Bildschirm erscheinen, können den Gesamteindruck einer Software sehr schnell verschlechtern. Die Zeiten, welche benötigt werden, um die einzelnen Teile der Seite zu laden, können heutzutage in der *Web Developer Toolbar*, früher hauptsächlich in *Firebug*, untersucht werden. [25]

Durch die Analyse der einzelnen Seiten und der dafür benötigten Ladezeiten, wenn das System auf einem schwächeren Server lief, war es möglich, auf die

[25] nach Hoffmann, *Modernes Webdesign: Gestaltungsprinzipien, Webstandards, Praxis*, Seite 342

aufwändigsten Seiten des Systems zu schließen. Anfänglich benötigte die Software auf einem schwächeren Server bis zu 20 Sekunden, um die gesammelten Feedbackergebnisse an den Nutzer zu übermitteln. Ein weiterer „Flaschenhals“war das Anzeigen von Fragebögen, wofür ursprünglich ebenfalls mehrere Sekunden benötigt wurden. Durch einigen Optimierungen in den Bereichen Datenbankzugriff und der darauffolgenden Datenverarbeitung, konnten die benötigten Ladezeiten um den Faktor 10 reduziert werden.

Kapitel 4

Einführung des neuen Systems im Euopagymnasium Auhof

Um das System nicht nur zu erstellen, sondern auch produktiv einzusetzen, wurde es Schritt für Schritt in die IT-Landschaft und später auch in den Schulalltag des Europagymnasiums Auhof eingegliedert.

4.1 Installieren des Systems in die IT-Landschaft der Schule

Noch während der Erstellung des Systems, wurde bereits kontrolliert, ob das definierte Ziel der einfachen Installation in der IT-Landschaft der Schule erreicht werden konnte. Hierfür wurde es testweise in der Schule installiert und auf eventuell auftretende Fehler hin untersucht. Während dieser Phase können Geschwindigkeitsprobleme, Kompatibilitätsprobleme oder auch andere, unerwartete Schwierigkeiten auftreten. Durch die in 3.4.2 beschriebenen Optimierungen, konnte das System schließlich auch außerhalb der Entwicklerlandschaft lauffähig gemacht werden.

Spätere Aktualisierungen der Datenbank können automatisch mit *Phinx*[26] und Zugriff auf die Kommandozeile des Servers durchgeführt werden. Die Distribu-

[26] Morgan, *Phinx*.

tion des Quelltextes wird währenddessen manuell ausgeführt, da zusätzliche Aktualisierungssoftware den Server unnötig belasten würde.

4.2 Präsentation im SchülerInnenparlament

Um das System schließlich auch offiziell in den Schulalltag zu integrieren und Aufmerksamkeit auf diese neue Entwicklung zu lenken, wurde eine Präsentation während des SchülerInnenparlaments 2014 gehalten. Diese Vorstellung enthielt Informationen zum damaligen Entwicklungsstand, dem Funktionsumfang und den geplanten Zielen des Systems, welche über die Plattform des SchülerInnenparlaments direkt an alle SchülerInnen der damaligen Oberstufe, einige LehrerInnen und die SchülerInnenvertretung gebracht werden konnten.

Durch die dadurch gewonnene Aufmerksamkeit, konnte das System nach einer Prüfung des Direktors schließlich als die offizielle Feedbackmethode der Schule eingeführt werden. Nach dieser Einführung kamen noch einige Funktionswünsche hervor, welche in der nächsten Version implementiert wurden und im neuen Schuljahr auch in der Schule aktiv geschaltet werden konnten.

4.3 Präsentation im Schulgemeinschaftsausschuss

Nachdem das System von den SchülerInnen akzeptiert wurde und von der SchülerInnenvertretung nun auch gefordert wurde, konnte ein Termin im nächsten SGA vereinbart werden. Die neuen Funktionen, welche aufgrund der Rückmeldungen aus dem SchülerInnenparlament umgesetzt wurden, waren bei der Vorstellung vor dem SGA bereits enthalten. Die Präsentation überzeugte die TeilnehmerInnen dadurch zusätzlich von der Funktionalität und der Sicherheit des Systems und bestätigte ihre Meinungen.

4.4 Weitere Schritte und Verbreitung des Systems

Ein System, das für die Verwendung an möglichst vielen Schulen gestaltet ist, sollte nicht nur auf eine einzelnen Bildungseinrichtung beschränkt bleiben. Mit der Hilfe der SchulsprecherInnen, wurden die Informationen über das Feedbacksystem auch an andere Schulen weitergeleitet, in denen die Stimmen nach einem eigenen LehrerInnenfeedback ebenfalls bereits lauter wurden. Viele LehrerInnen haben (berechtigte) Bedenken über die Datensicherheit von extern laufenden Feedbacklösungen, welche von dem System aus dieser VWA direkt angesprochen und gelöst werden.

Mit steigender Verbreitung von Feedbacksystemen und der damit einhergehenden, steigenden Debatte um die Sicherheit der erhobenen, personenbezogenen Daten, wächst auch die Nachfrage nach einem derartigen, lokalen System. Durch die weitere Verbreitung kann letztendlich hoffentlich die Unabhängigkeit der Schulen und die Qualität der einzelnen Bildungseinrichtungen angehoben werden, was das Ziel dieser Arbeit erfüllen würde.

Kapitel 5

Zusammenfassung und Ausblick

Auch wenn die Entwicklung eines webbasierten Feedback-Systems viel Zeit in Anspruch nimmt, ist es den Aufwand laut vielen TesterInnen dennoch wert. Es schafft Möglichkeiten zur Auswertung, zur mobilen Durchführung, zur Zeitersparnis und zur Zusammenarbeit, welche ohne eine derartige Entwicklung nicht immer möglich wären.

Die größten Bedenken einer digitalen Lösung dieses Problems haben Verantwortliche in Zusammenhang mit der Datensicherheit. Diese Bedenken sind auch durchaus berechtigt, da derartige Systeme meistens viele personenbezogene Daten speichern. Durch die Verwendung eines lokal in der Schule installierten Programms, kann die Kontrolle, und damit die Datenhoheit, allerdings innerhalb der Schule gehalten werden. Somit kann jede Bildungseinrichtung ein eigenes LehrerInnenfeedback durchführen, ohne vorgefertigte Fragen akzeptieren zu müssen, oder sich Sorgen um die Datensicherheit zu machen.

Geschwindigkeitsprobleme und unpassende Benutzeroberflächen konnten durch geschicktes Optimieren beseitigt werden und lassen diese Feedbacklösung auch auf sehr günstiger Hardware mit guter Geschwindigkeit laufen. Mit diesen Optimierungen konnten auch die überdurchschnittlich langen Ladezeiten der Ergebnisseiten vom Schulserver im Europagymnasium Auhof stark reduziert

werden. Die verwendeten Softwarebibliotheken ermöglichen einen reibungslosen Betrieb auch auf älteren Systemen (bis zu PHP 5.4), was einen Einsatz auf älteren Servern und in Schulen nochmals stark begünstigt.

Ein großes Problem war das Unterstützen von veralteten Systemen und PHP-Installationen. Um dieses Problem zu lösen, habe ich versucht, mich möglichst gut an etablierte Standards zu halten und das Feedback System auch auf älteren Systemen (wie zum Beispiel eigene PHP Installationen auf Test-Servern oder in einer Virtual Machine (VM)) zu testen. Das Problem von fehlenden PHP-Funktionen, konnte durch den Einsatz spezieller Kompatibilitätsbibliotheken (siehe 2.2.3) ebenfalls gelöst werden.

Während diesem Projekt und dieser VWA konnte stieß ich auf einige Probleme, welche sich mir in der Vergangenheit noch nicht in den Weg gestellt hatten. Durch das Erarbeiten neuer Lösungen und das Finden von etablierten Praktiken, konnte ich allerdings die meisten Probleme lösen und die verbleibenden Schwierigkeiten durch andere Herangehensweisen umgehen, weshalb mir dieses Projekt viele neue Erfahrungen liefern konnte. Die Optimierungen für leistungsschwächere Systeme und die Methoden zur Benutzung älterer PHP-Versionen waren besondere Highlights dieser neuen Techniken, da ich diese Probleme zuvor auf meiner eigenen Hardware nicht antreffen konnte. Außerdem war die starke Kommunikation mit ProfessorInnen und SchülerInnen, sowie das häufige Präsentieren meiner Arbeit stets eine gute Erfahrung, die in diesem Ausmaß ebenfalls neu für mich war.

Ein letztes Problem ist der individuelle Zeitaufwand der benötigt wird, um ein solches System zu erstellen. Da ich mich nicht vollständig auf ein einzelnes Softwareprojekt konzentrieren will, ohne dafür einen Gegenwert zu erhalten, ist dies der letzte limitierende Faktor. Das System ist bereits seit dem Schuljahr 2015/16 einsatzbereit und in Verwendung, Ansatzpunkte zum Verbessern der Software gibt es aber weiterhin in großer Zahl. Mit weiterer Unterstützung

würde sich das System, nach meiner persönlichen Vermutung, zu einer universalen Lösung für viele Bildungseinrichtungen entwickeln. Zum Zeitpunkt dieser VWA ist dieser letzte, große Schritt aber noch weiter entfernt.

Trotz dieser Verbesserungsmöglichkeiten wird das System allerdings auch jetzt bereits für viele andere Schulen gut geeignet sein und auch noch anderen ProfessorInnen beim Einholen von Meinungen über ihre Unterrichtsmethoden behilflich sein können. Diese neuen NutzerInnen haben dann eventuell wieder Anforderungen, welche bisher noch unbekannt waren. So entwickelt sich diese dezentrale Feedbacklösung immer weiter und bekommt ein immer größeres Potential, den Schulalltag vieler ProfessorInnen und SchülerInnen angenehmer zu gestalten.

A Tabellenstrukturen

Dieser Anhang beinhaltet alle Tabellen, welche eventuell zum Verständnis der Datenspeicherung des Systems hilfreich sein können.

Tabelle 3: Struktur der Tabelle `users`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
password	varchar(255)	Nein		Passworthash
salt	varchar(20)	Nein		Hashing-Salt
email	varchar(255)	Nein		E-Mail
fullname	text	Nein		Voller Name
iquestions	text	Nein		Individuelle Fragen
permissions	text	Nein		Rechte

Tabelle 4: Struktur der Tabelle `tickets`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
idcode	varchar(5)	Nein		Ticket-Code (ABCDE)
survey	int(11)	Nein		dazugehörige Umfrage
free	tinyint(1)	Nein	1	Ticket zugeordnet
lastupdate	int(11)	Nein		Letzte Aktualisierung/Zugriff
inUse	tinyint(1)	Nein	0	Wird gerade ausgefüllt
accessCode	varchar(4)	Nein		Zugangscode (1234)

Tabelle 5: Struktur der Tabelle `results`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
results	text	Nein		Ergebnisse (Fragebogen)
iresults	text	Nein		Ergebnisse (Individualfragen)
sresults	text	Nein		Ergebnisse (Benutzerfragen)
survey	int(11)	Nein		ID der zugehörigen Umfrage
timestamp	timestamp	Nein		Abgabedatum

Tabelle 6: Struktur der Tabelle `questions`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
question	text	Nein		Fragentext
type	int(11)	Nein	0	Fragentyp
created	timestamp	Nein		Erstellungsdatum

Tabelle 7: Struktur der Tabelle `questionsets`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
name	varchar(200)	Nein	*Undefined Name*	Fragebogenname
questions	text	Nein		Fragenids
created	timestamp	Nein		Erstellungsdatum

Tabelle 8: Struktur der Tabelle `survey`

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
name	varchar(250)	Nein		Umfragename

Tabelle 8: Struktur der Tabelle survey (Fortsetzung)

Spalte	Typ	Null	Standard	Kommentare
questions	text	Nein		Fragenids (Fragebogen)
iquestions	text	Nein		Fragenids (Umfrage)
squestions	text	Nein		Fragenids (Nutzer)
questionset	int(11)	Nein		Verwendetes Fragenset
creator	int(11)	Nein		Umfrageersteller
done	tinyint(1)	Nein	0	Umfrage abgeschlossen
timestamp	timestamp	Nein		Erstellungsdatum

Tabelle 9: Struktur der Tabelle news

Spalte	Typ	Null	Standard	Kommentare
id	int(11)	Nein		ID-Nummer
text	text	Nein		Nachrichteninhalt
issued	timestamp	Nein	CURRENT_TIMESTAMP	Erstellungsdatum
type	int(11)	Nein		News-Typ
read	tinyint(1)	Nein	0	Lesebestätigung
user	int(11)	Nein		Zielnutzer

B Persönliche Erklärung zum Layout dieser VWA

Ich habe in dieser VWA sehr viele layouttechnische Entscheidungen auf Basis meiner persönlichen Präferenzen getroffen und versuche damit, diese Arbeit so zu gestalten, wie ich sie persönlich für richtig halte. In diesem Anhang will ich auf kleinem Raum ausführen, weshalb ich diese Entscheidungen getroffen habe, anstatt einen anderen Weg einzuschlagen.

B.1 LaTeX

LaTeXist eine Alternative zu Programmen wie *Microsoft Word* oder *LibreOffice*. Mit diesem Programm kann man sich als technikbegeisterter Autor voll und ganz auf den Text konzentrieren, ohne sich Sorgen um das Layout machen zu müssen. Man arbeitet praktisch getrennt am Text und an dessen Layout, wodurch die Sorge um eine richtige Textpräsentation, wie sie viele *Word*-Nutzer kennen, viel geringer wird.

Für mich ist die Entscheidung aufgrund von vielen schlechten Erfahrungen mit grafischen Texteditoren und wegen der geschlossenen und schwer versionierbaren Struktur der Dateiformate von *Word* oder *LibreOffice* auf LaTeXgefallen. Besonders als Programmierer ist die Syntax der Sprache nicht schwer zu erlernen, deshalb war diese Entscheidung für mich optimal.

B.2 Weißraum und Seitenlayout

Meiner Meinung nach sollte die Gestaltung von Seiten den Lesern dabei helfen, den Text möglichst gut und schnell lesen zu können. Da das Auge bei fehlendem Weißraum leicht die aktuelle Zeile verlieren kann[27], habe ich mich für viel Platz zwischen Paragraphen und Überschriften entschieden. Diese Abstände erleichtern meiner Meinung nach das Lesen drastisch, weshalb diese Gestaltung auch in der finalen Formatierung gewählt wurde.

B.3 Positionen der Abbildungen

Die Abbildungen in dieser VWA sind nicht immer unter der Textstelle, welche auf sie Bezug nimmt. Stattdessen sind sie immer am oberen Ende einer Seite und es wird im Text auf sie verwiesen. Diese Gestaltung wurde gewählt, um eine gewisse Konsistenz zwischen den Seiten zu schaffen. Bilder stechen auf diese Weise nicht zu sehr hervor und der Text wirkt, meiner Ansicht nach, harmonischer und zusammengehörender.

Sobald auf eine Abbildung verwiesen wird, weiß man als Leser immer, an welcher Stelle auf den umliegenden Seiten nach der Abbildung gesucht werden muss. Falls die Illustration allerdings nicht im Interesse des Lesers steht, kann sie auf diese Weise ohne Aufwand übersprungen werden.

[27] Hoffmann, *Modernes Webdesign: Gestaltungsprinzipien, Webstandards, Praxis*, S. 74.

C Abkürzungsverzeichnis

CAPTCHA Completely Automated Public Turing test to tell Computers and Humans Apart

SQL Structured Query Language

JSON JavaScript Object Notation

Enum Aufzählung

SGA Schulgemeinschaftsausschuss

VM Virtual Machine

D Abbildungsverzeichnis

E Tabellenverzeichnis

F Quelltextverzeichnis

G Literaturverzeichnis

Bastian, Johannes, Arno Combe und Roman Langer. *Feedback-Methoden - Erprobte Konzepte, evaluierte Erfahrungen*. neu ausgestattete Sonderausgabe. Beltz Verlag, 2007. ISBN: 9783407254689 (siehe S. 30).

Composer Project. *Composer*. 2015. URL: https://getcomposer.org/ (siehe S. 18).

Downie, Nick. *Chart.js*. 2015. URL: http://www.chartjs.org/ (siehe S. 23, 40).

enuvo GmbH. *Umfrage Online*. 2015. URL: https://www.umfrageonline.com (siehe S. 16).

Filp. *whoops!* 2015. URL: https://filp.github.io/whoops/ (siehe S. 21, 39).

Gaigalas, Alexandre Gomes. *Respect Validation*. 2015. URL: https://respect.github.io/Validation/ (siehe S. 20).

Hagenbuchner, Karl. *Schoolfeedback.at*. 2014. URL: http://schoolfeedback.at/ (siehe S. 16).

Hoffmann, Manuela. *Modernes Webdesign: Gestaltungsprinzipien, Webstandards, Praxis*. Zweite Ausgabe, 1. Überarbeitung. Galileo Design, 2010. ISBN: 9783836215022 (siehe S. 22, 41, 53).

HTML Purifier Community. *HTML Purifier*. 2015. URL: http://htmlpurifier.org/ (siehe S. 20).

ircmaxwell. *password_compat*. 2015. URL: https://github.com/ircmaxell/password_compat (siehe S. 21).

Mathews, Jamie. *Idiorm & Paris*. 2015. URL: https://j4mie.github.io/idiormandparis/ (siehe S. 19).

Morgan, Rob. *Phinx*. 2015. URL: https://phinx.org/ (siehe S. 22, 43).

Phillips, Drew. *Securimage PHP Captcha*. 2015. URL: https://www.phpcaptcha.org/ (siehe S. 20).

staffadvance GmbH. *123lehrerfeedback.de*. 2015. URL: https://www.123lehrerfeedback.de (siehe S. 15).

The PHP Group. *PHP: Hypertext Preprocessor*. 2015. URL: https://secure.php.net/ (siehe S. 18).

The PHP Group. *PHP: Password Hashing - Manual*. 2015. URL: https://secure.php.net/password (siehe S. 21).

Twitter & Contributors. *Bootstrap*. 2015. URL: http://getbootstrap.com/ (siehe S. 23).

Underscore Contributers. *Underscore.js*. 2015. URL: http://underscorejs.org/ (siehe S. 19).

Wassermann, Tobias. *Sichere Webanwendungen mit PHP*. Erste Ausgabe. mitp, 2007. ISBN: 9783826617546 (siehe S. 20).

WordPress Organisation. *WordPress: Blog Tool, Publishing Platform, and CMS*. 2015. URL: https://wordpress.org/ (siehe S. 18).