Entwicklung und Erprobung einer Advanced Learning Platform – Mobile Learning in der Hochschullehre

Hochschule Mittweida

Prof. Dr. Thoralf Gebel | Christian Ulbrich | Fakultät Wirtschaftsingenieurwesen

Stefan Dahlmanns | Isabelle Kuxdorf-Dixon | Fakultät Angewandte Computer- und Biowissenschaften

Abstract

Neue Methoden und Instrumente der digitalen Lehre stellen ein Schlüsselelement für die Umsetzung dezentraler, flexibler und innovativer Studienformate dar. Vor allem die Integration mobiler Applikationen bietet in diesem Kontext große Potenziale. Als Element der Ergänzung, aber auch Erweiterung der asynchronen Lehre, bietet diese relativ junge Form der Lehre viele Vorteile gegenüber klassischen, desktoporientierten Formaten.

Das Projekt Open Engineering 2 (OE2) sieht vor allem im Kontext der berufsbegleitenden Weiterbildung und innerhalb der Integration internationaler Anspruchsgruppen einen erheblichen Mehrwert für derartige Anwendungen. Im Rahmen einer Kooperation des Projektes OE2 mit der Professur für Innovations- und Changemanagement sowie der Professur für Medieninformatik konnte die Entwicklung, Gestaltung und Erprobung mit integrierter Evaluation einer anwendungsorientierten mobilen Lernapplikation mit der Bezeichnung „Advanced Learning Plattform“ (APL) realisiert werden.

Im Beitrag werden die Integrations-, Konzeptions- und Implementierungsaktivitäten beschrieben. In Ergänzung zu weiterführenden Dokumenten in der Gestaltung und Nutzung der Applikation sowie den Evaluations-Auswertungen, bilden die Beschreibung und Auswertung der Ergebnisse des Testlaufs eine Datenbasis für qualitätsorientierte Maßstäbe als Rahmenbedingungen des Einsatzes mobiler Lernanwendungen in der dezentralen asynchronen Hochschullehre.

April 2020

Das diesem Bericht zugrundeliegende Vorhaben wird mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 16OH21011 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor/bei der Autorin.

Inhalt

1. Zielstellung im Rahmen des Projektes OE2

Mit der Entwicklung der Studienplattform Open Engineering wird die Schaffung eines flexiblen, auf unterschiedliche individuelle Voraussetzungen ausgerichteten Weiterbildungsangebotes zur Verbesserung der Durchlässigkeit von Bildungswegen der akademischen Aus- und Weiterbildung forciert, in dem die Passgenauigkeit der entstehenden Studienangebote für unterschiedliche Eingangsqualifikationen der Studieninteressenten erhöht werden soll. Das Vorhaben bezieht sich auf die Schaffung vielfältiger bedarfsgerechter Zugänge in die akademische Aus- und Weiterbildung, insbesondere die berufsbegleitende Weiterbildung, und die Etablierung einer neuen bedarfsgerechten Lehrgestaltung. Dies umfasst auch die Integration und Erprobung innovativer Lehrformate, welche die Studierbarkeit im dezentralen und flexiblen Weiterbildungskontext erhöhen.

Digitale Instrumente zur Umsetzung von Blended Learning-Ansätzen, als neue Ansätze in der Hochschuldidaktik, bilden im Rahmen der Entwicklung der Studienplattform Open Engineering durch eine innovative Lehrprozessgestaltung einen wichtigen Baustein bei der Öffnung der Hochschulen durch innovative Lehr-/Lernkonzepte. Sie bieten die Möglichkeit einer individuelleren Gestaltung der Lernprozesse, die den Studierenden hilft, einen, ihren individuellen Rahmenbedingungen entsprechenden Einstieg ins Studium zu schaffen und diesen barrierearm zu absolvieren, die geforderten Studienerfolge zu erreichen und Studienabbrüche zu verringern.

Aus dieser Motivation leitet sich das Konzept der mobilen Applikation ab, die als weiterer Bestandteil einer Plattform benötigt wird. Im Beitrag werden zunächst die Rahmenbedingungen und die Aufgabenstellung des Vorhabens erläutert. Daraufhin wird das Konzept der Plattform erklärt, indem auf Anforderungen der einzelnen Komponenten eingegangen, ein Systementwurf präsentiert und die Entwicklungs- und Systemumgebung beschrieben wird sowie die Design-Richtlinien aufgezeigt werden. Anschließend folgt die Implementierung der einzelnen Bestandteile der Plattform, gefolgt von der Evaluation und der Formulierung eines Fazits.

1.1 Rahmenbedingungen

Ausgehend von Ergebnissen der privatwirtschaftlichen Weiterbildungsforschung zeigt sich ein großer Trend zu E-Learning Anwendungen, was in Abbildung 1 zu erkennen ist. Demnach haben sowohl „Micro Learning“, „Blended Learning“ und „Mobile Anwendungen“ eine hohe Bedeutung als Lernformen in Unternehmen.

Für den Bereich der wissenschaftlichen Weiterbildung ergeben sich zahlreiche Schnittmengen, da die Herausforderungen in Bezug auf die Durchführbarkeit und Vereinbarkeit mit weiteren Rahmenfaktoren recht ähnlich sind, z.B. Familie und den beruflichen Alltag. Ferner wird durch theoretische Inhaltsvermittlung viel Zeit im Vorlesungsbetrieb beansprucht, sodass die Herausforderung besteht, die theoretischen, repetitiven Inhalte zusammenzufassen und in kleinen verständlichen „Paketen“ mobil abrufbar anzubieten. Dabei werden die Konzepte des Blended Learning, Micro Learning und Mobile Learning verbunden, um Lerninhalte zu digitalisieren, unterwegs verfügbar zu machen sowie interaktiv und verständlich zu gestalten. Damit ergeben sich Möglichkeiten, diese neben Präsensveranstaltungen zu verwenden, in denen genauer auf die vermittelten Inhalte und Praxisbeispiele eingegangen werden kann.

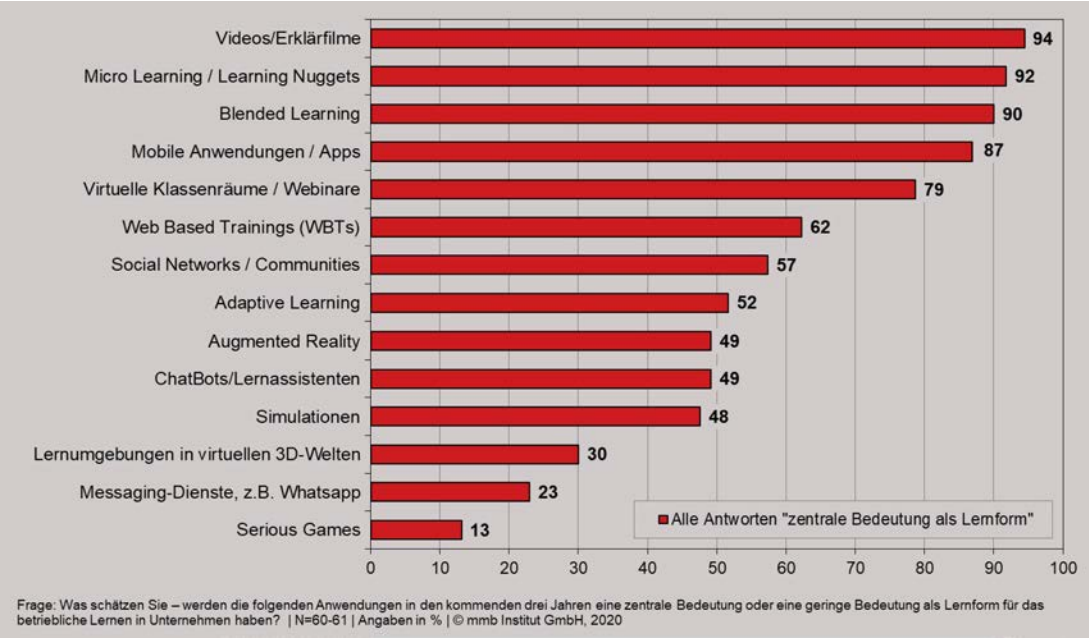

Abbildung 1: Bedeutung von Anwendungen als Lernform in Unternehmen[1]

1.2 Begriffsbestimmung Mobile Learning

Der Begriff des „Mobile Learning“ umfasst sämtliche Systeme, welche den Lernenden erlauben, auf dezentrale Lernumgebungen zuzugreifen und miteinander zu kommunizieren. Diese Umgebungen sind hinsichtlich ihrer Nutzeroberfläche vor allem auf die Nutzung auf drahtlosen Endgeräten zugeschnitten, wie z.B. Laptops, Tablets und Smartphones.[2]

Der Begriff wird im Lehr-Lern-Kontext des Blended Learning und Micro Learning (Mikrolernen) angesiedelt. Blended Learning soll in diesem Zusammenhang als ein integriertes Lernarrangement verstanden werden, in dem die verfügbaren Möglichkeiten der Vernetzung über Internet und Intranet in Verbindung mit klassischen Lernmethoden und -medien optimal genutzt werden.[3] Mikrolernen beschreibt wiederum die Umsetzung von Lehre durch relativ kleine Lerneinheiten und kurzfristig ausgerichtete Aktivitäten, welche sich durch Zeitaufwand und Frequenz vom klassischen Lernen unterscheiden.[4]

Mobile Lerntechnologien unterscheiden sich von anderen E-Learning-Anwendungen insbesondere durch die speziellen technischen Bedingungen. So zeichnen sich die mobilen Technologien durch einen kleineren Bildschirm, geringere Bandbreite und geringeren Speicherplatz aus und wurden speziell für den Einsatz unterwegs entwickelt. E-Learning Anwendungen, die für stationäre Geräte entwickelt wurden, z.B. Desktop-PC`s, sind in der Darstellung meist auf großformatige Bildschirme ausgelegt. Folglich müssen Inhalte für mobile Geräte in Form von Apps oder responsiven Webseiten aufgebaut sein,

[1] mmb-Trendmonitor 2019/2020 des mmb Institut (2020)
[2] vgl. Karran 2003
[3] vgl. Erpenbeck, Sauter, Sauter (2015)
[4] vgl. Hug, Theo (2005)

um die Nutzbarkeit durch den Endanwender auch auf geringen Bildschirmgrößen zu optimieren. Responsive Autorentools bieten mittlerweile die Option, dieselben Inhalte und Funktionen sowohl auf dem Computer, als auch auf der mobilen Plattform abruf- und darstellbar zu gestalten. Hierfür muss das Lernangebot entsprechend auf den mobilen Einsatz angepasst werden.[5]

1.3 Entwicklung der Zielstellung

Zu Beginn der Kooperation zwischen den beiden Fakultäten der Hochschule im Herbst 2019 bestand die ursprüngliche Zielstellung des Projektes darin, eine mobile und interaktive Applikation zur Vermittlung der wissenschaftlichen Grundlagen zum Thema Entrepreneurship zu entwickeln. Die Applikation sollte primär repetitive Lehrinhalte automatisieren und durch Überprüfung des Gelernten den Lernerfolg steigern. Zusätzlich dazu sollte die Applikation im zweiten Semester internationalisiert und eine Möglichkeit zur Erweiterbarkeit durch Einpflegen neuer Inhalte von Dozenten ermöglicht werden. Dabei entstand ein erster funktioneller Prototyp, der die mobile Applikation und eine dazugehörige Autoren-Oberfläche umfasste.

Im Laufe des Vorhabens sollte diese Anwendung erweitert und optimiert werden, sodass sie in Lehreinrichtungen ergänzend zu traditionellen Lehrveranstaltungen zur Vermittlung von digitalen Lehrinhalten genutzt werden kann. Dafür sollte, um auch die zukünftige Weiterentwicklung zu vereinfachen, das Projekt in eine neue Entwicklungsumgebung eingebettet werden. Dabei mussten vor allem auch die Datensicherheit und Usability besondere Beachtung finden und verbessert werden. Außerdem wurde die Anwendung um eine Administrationsoberfläche erweitert, welche es ermöglicht, Apps zu erstellen und zu verwalten, um verschiedene Lernfelder abdecken zu können. Zudem wurde die mobile Applikation um Funktionen erweitert, die die Motivation des Lernenden durch die Implementation verschiedener Gamification-Ansätze steigert. Zusätzlich sollte die Plattform um ein Analyse-Tool ergänzt werden, welches dem Lehrenden eine Möglichkeit zur Analyse des Nutzerverhaltens und der Leistung der Studenten bietet, sodass er gezielt in Lehrveranstaltungen Defizite ansprechen und ausgleichen kann.

2. Darstellung der Konzeptbestandteile

Im Folgenden wird zuerst das angestrebte Endprodukt durch eine Anforderungsanalyse definiert und detaillierter spezifiziert. Darauf folgt der Systementwurf, der die neue Systemarchitektur des Gesamtsystems und die Festlegung der Entwicklungs- und Systemumgebung beschreibt. Zum Abschluss wird das Design-Konzept erläutert. Ziel ist, den Qualitätsrahmen der Anwendung am konkreten Beispiel zu definieren.

2.1 Anforderungsanalyse

Mit der detaillierten Beschreibung der funktionalen und nicht funktionalen Anforderungen an die Plattform werden besonders die für das Projekt relevanten Anforderungen

[5] vgl. De Witt, 2013

fokussiert. Vorherige Anforderungen aus dem Prototypen bleiben erhalten. Dazu werden zuerst die für die gesamte Anwendung und danach jeweils die für die einzelnen Komponenten gültigen Anforderungen definiert.

Im Rahmen der Projekterweiterung lag der Fokus insbesondere auf der Verbesserung des bestehenden Konzepts in Hinsicht auf Sicherheit und Usability, um es für die Anwendung im Hochschulbetrieb tauglich zu machen. Auch die geräte- und browserübergreifende Kompatibilität ist eine der wichtigsten Anforderungen und sollte bei der Umstellung auf die neue Systemarchitektur berücksichtigt werden. Getestet wurde die Anwendung clientseitig unter folgenden Spezifikationen:

- Mobile App: Android 7, Chrome/ IOS 13.3, Safari/ Emuliert - Windows 10, Chrome V81.0
- Desktop Anwendungen: Windows 10, Chrome V81.0/Microsoft Edge 44.1.

2.1.1 Admin-Oberfläche

Die Admin-Oberfläche ist eine neue Komponente, die es ermöglichen soll, mehrere App-Instanzen einer Installation der „Advanced Learning Plattform“ (ALP) zu verwalten. Diese soll die praktische Anwendung innerhalb einer Lehreinrichtung ermöglichen. Das Verwalten lässt sich dabei in mehrere Kernfunktionalitäten unterteilen:

Zum einen das Anlegen einer neuen App, bei welcher der Name und eine URL für eine App-Instanz angegeben werden soll, woraufhin diese dann automatisch erstellt und konfiguriert wird. Dies soll beliebig oft ausführbar sein, solange die Hardware-Ressourcen ausreichend sind. Dadurch lassen sich die Lernfelder leicht diversifizieren und es können verschiedene Themengebiete oder auch Module parallel bedient werden.

Für die vorhandenen App-Instanzen sollen zum anderen Updates gemacht werden können. Dabei soll sowohl die Möglichkeit gegeben sein, alle App-Instanzen zur gleichen Zeit sowie auch einzelne Instanzen unabhängig vom Rest zu updaten. Zur besseren Übersicht sollen dafür die Versionsnummer und der Zeitpunkt des letzten Updates für jede App-Instanz angezeigt werden. Um den Admin über neue Versionen der Anwendung zu informieren, soll zudem in einem Feld in der Übersicht die neuste, verfügbare Version angezeigt werden, mit dem zugehörigen Veröffentlichungs-Datum und einer Zusammenfassung der neuen Funktionalitäten und Anpassungen. Die Admin-Oberfläche soll dabei auf die Nutzung an einem Desktop-PC ausgelegt und die grafische Oberfläche entsprechen gestaltet sein.

2.1.2 Autoren-Oberfläche

Die Anforderungen der Autoren-Oberfläche beziehen sich größtenteils auf Funktionen, die einen erweiterten oder verbesserten Funktionsumfang im Gegensatz zum Prototypen bieten. Multimediale Inhalte, dazu zählen Bilder und Videos, sollen getrennt voneinander in Galerien verwaltet werden können. Das Ziel dieser Galerien ist es, den Vorgang des Einfügens von Multimedia-Inhalten einfacher zu gestalten. Dies erfolgt, indem zunächst mehrere Bilder oder Videos in der jeweiligen Galerie hochgeladen werden können und daraufhin an der gewünschten Stelle in einem Unterkapitel eingefügt werden. Mittels eines Knopfdrucks soll es möglich sein, die Informationen und Inhalte eines Kapitels und dessen Unterkapitel als PDF zu exportieren. Diese Funktion dient der Kontrolle der Inhalte, ohne direkten Zugriff auf die Autoren-Oberfläche zu benötigen. Über eine

Verlinkung soll ein Autor Inhalte, die der Statustest besitzt, in einer Test-Ansicht der Applikation bereits vor dem Veröffentlichen der Inhalte betrachten können. Damit können eventuelle Fehler in den Inhalten, der Anordnung oder der Formatierung bereits erkannt werden, bevor der Nutzer der Applikation diese sehen kann. Durch Drag-and-Drop-Funktionalitäten soll zum Beispiel die Anordnung von Kapiteln oder der Inhalte auf einer Seite vereinfacht werden, was die Nutzbarkeit der Autoren-Oberfläche verbessern soll. Nach einer ausgeführten Aktion soll ein Nutzer eine Rückmeldung über Erfolg oder Misserfolg dieser Aktion erhalten, um so zum Beispiel mögliche Fehler beim Speichern von Inhalten zu kommunizieren.

2.1.3 Analyse-Tool

Das Analyse-Tool ist ebenfalls eine neue Komponente. Es soll dem Lehrenden Feedback zu seinen App-Instanzen bereitstellen. So sollen zum Beispiel Nutzerstatistiken erstellt werden und inhaltliche Ergebnisse anonymisiert abrufbar sein. Diese Statistiken sollen dabei so aufbereitet sein, dass sie für den Lehrenden leicht einsehbar und verständlich sind. Um dies zu erreichen, sollen die Daten in Echtzeit visualisiert und dynamisch beziehungsweise interaktiv gefiltert werden können.

Während des Projekts sollen hierzu allerdings hauptsächlich die serverseitigen Grundvoraussetzungen entstehen, für die detaillierte Gestaltung und Entwicklung der Benutzeroberfläche werden zukünftig Anforderungen der nutzenden Lehrkräfte gesammelt und umgesetzt.

2.1.4 Mobile Applikation

Die mobile Applikation übernimmt die funktionalen und nicht-funktionalen Anforderungen der Prototypen. Zu den nicht-funktionalen Anforderungen zählen dabei, dass die Applikation auf allen gängigen mobilen Endgeräten in gängigen Browsern lauffähig sein soll, responsiv aufgebaut, modular, webbasiert und idealerweise als progressive Web App umgesetzt ist. Funktionale Anforderungen sind unter anderem das Anzeigen von Inhalten, Durchführen von Tests, ein Glossar oder ein kontinuierliches Freischalten von Inhalten durch ein Fortschrittssystem.

Im Rahmen des Projekts soll neben diesen bestehenden Anforderungen der Gamification Aspekt verstärkt werden. Dies soll mit Hilfe von neuen Fragetypen, Errungenschaften, einem Levelsystem und Benachrichtigungen dieser Errungenschaften und des Level-Fortschritts erreicht werden. Indem mehrere Inhalte auf einer Seite angezeigt werden können, kann die Nutzbarkeit verbessert und ein erhöhter didaktischer Mehrwert geschaffen werden.

Die Navigation innerhalb der Applikation soll durch Verknüpfungen zwischen Kapiteln, Unterkapiteln und Tests einfacher und schneller für den Nutzer erfolgen. Neben diesen neuen Anforderungen sollen die bestehenden Anforderungen verbessert werden, indem Tests eine verbesserte und besser verständliche Auswertung bekommen. Diese besteht darin, dass Fragen Bilder und Videos enthalten können, und das Fortschrittssystem überarbeitet wird, sodass der Nutzer durch Benachrichtigungen angezeigt bekommt, aus welchem Grund er nicht auf bestimmte Inhalte zugreifen kann. Zudem wird das Glossar durch Einfügen eines Such-Feldes erweitert.

2.2 Systementwurf

Die Systemarchitektur des Gesamtsystems (Abbildung 2), also die Advanced Learning Platform als Installation im Umfang für die Nutzung an einer Einrichtung, ist modular aufgebaut.

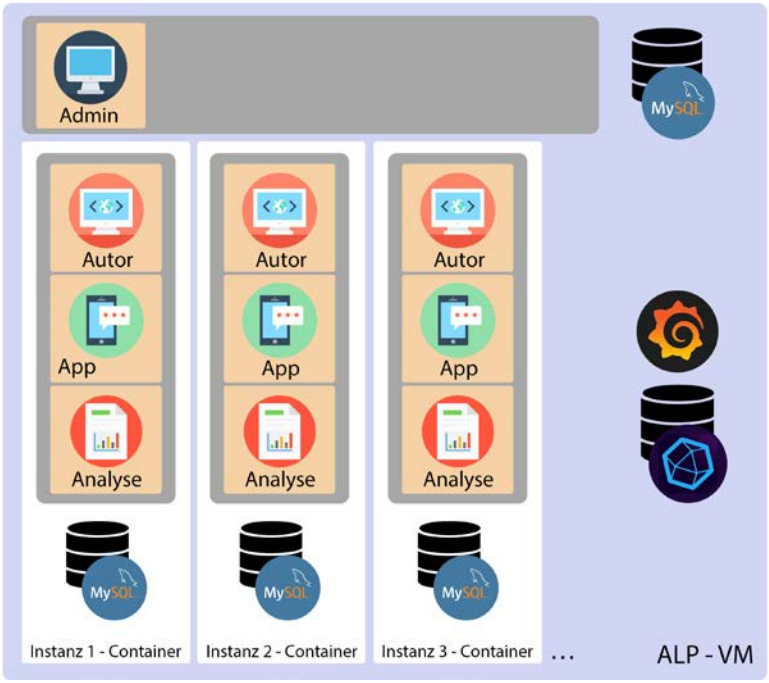

Abbildung 2: Übersicht der Systemarchitektur

Der modulare Ansatz findet dabei gleich auf zwei Ebenen Verwendung:

Zum einen wird durch *horizontale Modularität* ermöglicht, „zu Kontexten zusammengefasste Instanzen" zu erstellen. Dies wird realisiert, indem mithilfe von Container-Technologien die einzelnen Instanzen voneinander isoliert werden. Dadurch können diese zum Beispiel auf verschiedene Server verteilt werden, um die Last der einzelnen Server zu reduzieren. Im Falle eines Serverausfalls kann so eine Backup-Instanz auf einem Ersatz-Server direkt (ohne weiteren Installationsprozess) erstellt und gestartet werden. Außerdem weiß so eine Instanz nichts von der Existenz anderer Instanzen, wodurch ihre Datenbank unabhängig ist, ein eventueller Exploit1 wird so auf eine einzelne App begrenzt. Außerdem können so unabhängig Updates (inklusive der zugrunde liegenden Software) gemacht werden, Backups einer Instanz erstellt oder auch eine einzelne oder mehrere Instanzen in ein anderes System umgezogen werden.

Zum anderen werden durch *vertikale Modularität* einzelne Funktionalitäten innerhalb eines Kontexts voneinander getrennt. Dies sorgt für eine leichtere Wartbarkeit und Erweiterbarkeit, da Komponenten beliebig entfernt und hinzugefügt werden können. Auch das Hinzufügen von Funktionalitätsmodulen lediglich für ausgewählte App-Instanzen ist möglich. Außerdem beeinträchtigen Fehlfunktionen innerhalb eines einzelnen Projekts nicht die Funktionstüchtigkeit der anderen Komponenten. Eine einzelne Instanz wird von einem Debian 10-basierten Linux-Container umfasst. In diesem laufen die Komponenten

eines Webservers, um Software ausführen zu können. Dieser stellt die Rahmenbedingungen für die einzelnen Laravel-Projekte[6] zur Verfügung. Dabei benötigt jeder Kontext eine Autorenoberfläche, eine mobile App und ein Analyse-Tool. Diese liegen in separaten Ordnern und werden über mehrere vHosts2 angesprochen, greifen aber auf dieselbe Datenbank zu. Diese erlaubt das Speichern von Inhalten aus der Autorenoberfläche für die Verwendung innerhalb der mobilen App.

Damit die Instanzen nicht händisch erzeugt und verwaltet werden müssen, automatisiert und abstrahiert die zentrale Admin-Oberfläche diese Aufgabe. Da es sich bei dieser ebenfalls um ein Laravel-Projekt handelt, sind die Rahmenbedingungen ähnlich denen in den Containern. Hierbei wird auch auf einen Webserver und eine Datenbank zurückgegriffen.

Das System umfasst außerdem eine zentrale Zeitseriendatenbank und eine Installation des Grafana-Servers, die zur Datenvisualisierung für das Analyse-Tool benötigt werden.

2.3 Entwicklungs- und Systemumgebung

Die Entwicklungsumgebung umfasst alle zur Planung, Entwicklung und Verwaltung einer Software notwendigen Programme und Methoden. Im Fall der ALP wurde das Projektmanagement mithilfe von Trello3 realisiert. Für die Entwicklung wurden als Editoren Atom4 und Visual Studio Code5 und zur Versionsverwaltung wurde für das Debugging Chrome mit dem Vue-Developer Tools Extension7 genutzt. Die Softwarelösungen, die für die Kernfunktionalitäten der Anwendung notwendig sind, werden im Folgenden ausführlicher erläutert und ihre Wahl begründet.

2.3.1 Auswahl des Webservers

Um die Plattformunabhängigkeit zu gewährleisten, soll das System in Form einer Webanwendung umgesetzt werden, daher wird ein Webserver benötigt. Um die Rolle der Webserverkomponenten zu erfüllen, kamen ein Debian 10 (Linux-Distribution), ein Nginx-Webserver, eine MySQL-Datenbank auf Basis der MariaDB-Engine und PHP zum Einsatz. Diese Software-Zusammenstellung wird auch als LEMP-Stack bezeichnet. Dieser wurde dem weiter verbreiteten LAMP-Stack (Linux, Apache, MySQL, PHP) vorgezogen, da der geringere Umfang für das Projekt vollkommen ausreichend ist und so in mancher Hinsicht eine bessere Performance erzielt werden kann.

2.3.2 Virtualisierung

Um die Portabilität und Isolation der Komponenten und Instanzen untereinander in abstrakter Form zu realisieren, wurde als Technologie für die „Virtualisierung“ die in Linux integrierte Containerarchitektur LXC verwendet. Hierbei handelt es sich um eine Lösung, mithilfe derer verschiedene Linux User Spaces voneinander abgekapselt auf einer einzi-

[6] Laravel ist ein Web-Application-Framework mit grundlegenden Funktionen für Web-Anwendungen

gen Instanz des Linux Kernels parallel ausgeführt werden können. Es bietet also die Vorteile der Virtualisierung ohne erheblichen Ressourcenaufwand, wie es beispielsweise bei voll-virtualisierten Maschinen der Fall ist. Da die komplexeren Orchestrationswerkzeuge und Abhängigkeitssysteme, die Docker bietet, für das System nicht benötigt werden, wird für die Steuerung der Container die Kontrollsoftware LXC-Tools genutzt. Dadurch wird ein notwendiges Einarbeiten in eine komplexere Anwendung umgangen.

2.3.3 Web-Frameworks

Als Web-Framework wird Laravel verwendet, da es die nachfolgend beschriebenen Funktionen, speziell in den Bereichen Sicherheit, Routing und Datenbankanbindung mitliefert. Als Front-End-Framework wird Vue.js verwendet, da es als standardmäßiges Frontend-Framework direkt in Laravel-Projekten eingebunden werden kann. Laravel ist ein Web-Application-Framework, welches bereits die grundlegenden Funktionen mitbringt, die für Web-Anwendungen benötigt werden. Alle Teilbestandteile der ALP basieren zunächst auf einem standardmäßigen Laravel-Projekt, in dem durch das Paket laravel/ui und den Befehl php artisan ui vue –auth direkt die Grundlagen für Authentifizierung sowie die Einbindung von Vue.js geschaffen werden. Neben der Authentifizierung werden weitere Sicherheitsfunktionen von Laravel verwendet, wie die Autorisierung verschiedener Nutzerrollen mithilfe von Gates und API-Authentifizierung mittels Laravel Passport. Die Anbindung an die zugehörige MySQL Datenbank erfolgt zunächst, indem die Tabellen in Migrationsdateien skizziert werden und mit dem Befehl php artisan migrate in der Datenbank angelegt werden. Zum initialen Füllen einiger Tabellen wird eine Seed-Datei verwendet. Um die Daten einfach aus der Datenbank auslesen zu können, wird beim Erstellen der meisten Migrationsdateien jeweils ein Model erzeugt. Dabei erhält die Datenbanktabelle ein entsprechendes Model, über welches eine Interaktion mit der Tabelle erfolgen kann. Zum Beispiel kann das Model Chapter in einem Controller verwendet werden, um alle Kapitel über den Befehl Chapter::all() aus der Tabelle abzurufen.

Außerdem können mit Hilfe des Query Builder Datenbankoperationen über eine Schnittstelle ausgeführt werden. Die Routen der Anwendungen werden jeweils im routes Ordner definiert. Bevor der Nutzer eine Route betritt, kann mittels Middleware die Anfrage gefiltert werden, um so zum Beispiel zu prüfen, ob der Nutzer angemeldet ist. In einem Controller können mehrere Methoden definiert werden, die verschiedenen Routen zugeordnet werden. In diesen Methoden können unter anderem Daten aus der Datenbank gelesen werden und eine View zurückgegeben werden, welche in Laravel-Blade-Files erstellt wird. Blade ist dabei eine einfache Template-Engine, die von Laravel mitgeliefert wird.

Vue.js ist ein Framework zum Erstellen von Benutzeroberflächen. Vue.js, weiterführend als Vue bezeichnet, ist neben React das standardmäßige Frontend-Framework in Laravel-Projekten. Dabei wird in der app.js Datei im resources Ordner eine neue Vue-Instanz angelegt, der Komponenten zugefügt werden können. Diese Komponenten können global registriert sein, sodass sie zum Beispiel in allen Laravel-Blade- Files und Vue-Komponenten verwendet werden können oder lokal registriert werden, sodass sie nur in der jeweiligen Komponente verwendet werden können, in der sie registriert sind. Eine Komponente kann aus Template, Skript und Style Bestandteilen bestehen.

Daten können mittels Props bzw. Properties, also Eigenschaften, Daten von der Eltern-Komponente erhalten und Daten durch Events an diese zurückgeben. Im Template einer Komponente können vordefinierte Direktiven wie v-for oder v-if verwendet werden, um über Daten zu iterieren und diese in Elementen darzustellen oder Elemente nur in bestimmten Fällen zu rendern. Im Skript einer Komponente können unter Anderem Daten und Methoden definiert oder andere Komponenten registriert werden.

2.3.4 Datenvisualisierung

Um eine Grundlage zur späteren Visualisierung von Daten zu ermöglichen, wird Grafana in Kombination mit InfluxDB, einer Zeitseriendatenbank, genutzt. Dadurch wird eine vielfältige Einsatzweise offengehalten. Die Daten können – bedingt durch die Natur der Zeitreihendatenbank – beispielsweise in Echtzeit visualisiert, aber auch zur nachträglichen Auswertung aufgearbeitet werden. Der in Grafana integrierte Query-Builder erlaubt es Dozenten, selbst Metriken zu analysieren, ohne sich vorher mit der Influx-Query-Language vertraut zu machen. Dadurch sind die vom System vorgefertigten Dashboards leicht zu individualisieren bzw. auf die Bedürfnisse der Dozenten anzupassen und um viele in Grafana enthaltenen Analysemodule zu erweitern. Die individualisierten Dashboards können rückwirkend auf alle aufgezeichneten Daten angewendet werden. So sind neue Formen der Visualisierung von Daten aus vorangehenden Lehrveranstaltungen nachträglich möglich.

2.4 Design der Anwendung

Das Design der Anwendung stellt die Usability in den Mittelpunkt. Dabei orientiert es sich stark am Material Design. Es ist minimalistisch gehalten, um nicht von den Inhalten und Funktionen abzulenken, vermittelt jedoch durch den Einsatz von Schlagschatten ein Gefühl von Dreidimensionalität. Gewisse Zustandsänderungen (z.B. Klicken/ Hovern eines Buttons und Auffüllen eines Ladebalkens) werden durch Animationen hervorgehoben, um dem Nutzer ein direktes Feedback zu seiner Eingabe zu geben.

Das Colour Scheme (Abbildung 3) orientiert sich am Corporate Design (CD) der Hochschule Mittweida. Die Haus-Farbe ist aus diesem Grund als Primärfarbe vertreten. Ergänzt wird diese durch das ebenfalls im Hochschul-CD definierte Hell- und Dunkelgrau. Nach den Richtlinien eines monochromatischen Designs wurde außerdem eine Abwandlung der Primärfarbe hinzugefügt.

Zum Setzen von Akzenten und Hervorheben wichtiger Meldungen wurden zusätzlich die Signalfarben Rot und Grün gewählt. Da diese Wahl jedoch für Farbenblinde suboptimal ist, sollen diese, wenn sie einander gegenübergestellt werden, wie beispielsweise beim Anzeigen ob eine Antwort falsch oder richtig ist, zukünftig durch Symbole ergänzt werden. Auch soll in Zukunft die Farbwahl der Akzentfarben anpassbar sein, um Dozenten die Möglichkeit zu bieten, App-Instanzen individualisieren zu können, damit diese beispielsweise durch den Einsatz einer Fakultätsfarbe einer bestimmten Fakultät zugeordnet werden können.

Auch bei der Typographie kommt die Usabilty an erster Stelle. So steht hier die Lesbarkeit im Vordergrund. Um diese zu erhöhen werden Texte aufgelockert und in einer ausreichend großen Schriftgröße dargestellt, ohne dabei durch Einnahme von zu viel Platz auf einer Seite vermeidbares Scrollen zu erzeugen.

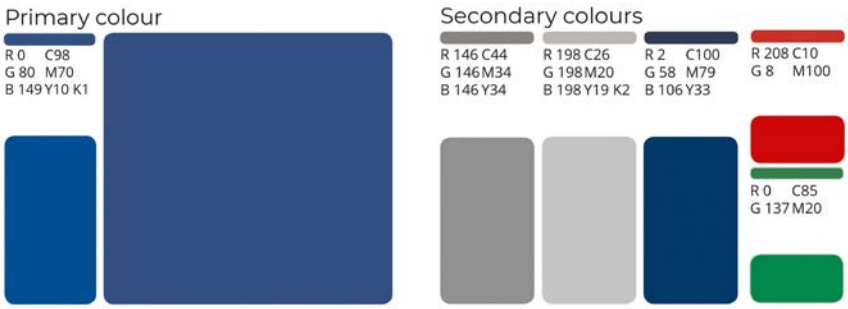

Abbildung 3: Übersicht der verwendeten Farbtöne

Für Überschriften wird die Schriftart Montserrat Regular und für Fließtext die Schriftart Open Sans Regular verwendet. Für Links wird die Schriftart Open Sans Regular unterstrichen genutzt, für die Beschriftung von Buttons die Schriftart Open Sans Bold (Abbildung 4).

Die Wahl fällt auf diese Schriftarten, da beide Sans Serif Schriften sind. Das bedeutet, dass die Buchstaben eine durchgängig gleichmäßige Strichstärke und keine Serifen besitzen. Dadurch sind die Buchstaben gut auf Bildschirmen darstellbar, weshalb Sans Serif Schriften sich gut für Webanwendungen eignen.

Bei Überschriften und Buttons ist die Schrift mittig zentriert, bei Fließtext hingegen ist sie linksbündig angeordnet, damit dieser für den Nutzer besser lesbar ist.

Abbildung 4: Verwendete Schriftarten

Bei sich häufig wiederholenden Buttons mit einer bestimmten Funktionalität, wie z.B. Speichern und Löschen, wird auf eine Beschriftung in Form von Worten verzichtet und stattdessen auf die Iconographie gesetzt. Mit der bildhaften Darstellung der Information lässt sich die Funktion eines Buttons vom Nutzer schneller erfassen.

Unterstützt wird dies zusätzlich durch das Gruppieren von Buttons zu kleinen Gruppen mit ähnlichen Funktionen oder verschiedenen Auswahlmöglichkeiten.

3. Implementierung

In diesem Kapitel werden die Teilbestandteile der ALP betrachtet. Dabei wird zunächst auf die Datenbank und deren Struktur eingegangen, daraufhin wird der Login und die Nutzerrollen betrachtet sowie jede Teil-Anwendung im Detail erklärt. Zu den Teil-Anwendungen gehört die Admin-Oberfläche, die Autoren-Oberfläche, das Analyse-Tool und die mobile Applikation. Jede dieser Komponenten basiert auf einem standardmäßigen Laravel-Projekt und Vue.js, wie in Kapitel 2.3.3 beschrieben.

Alle Projekte sind ähnlich aufgebaut und haben deutliche Parallelen. Um jede Komponente mehrsprachig umzusetzen, wird in Laravel-Blade-Dateien sowie Controllern, Middlewares und Models, die von Laravel mitgelieferte Localization verwendet. Dabei gibt es einen resources/lang Ordner, indem sich weitere Unterordner der verschiedenen

Sprachen befinden. In diesen Ordnern befinden sich php-Dateien, in denen die Übersetzungen der jeweiligen Teil-Anwendung gespeichert werden. In Vue-Dateien werden diese Übersetzungen ebenfalls benötigt, deshalb wird in diesen das Plugin vue-i18n verwendet. Um die Übersetzungen der php-Dateien in Vue nutzen zu können, wird ein weiteres Paket laravel-vue-i18n-generator benötigt, welches die Übersetzungen der php-Dateien in eine JavaScript-Datei umwandelt. Das Verwenden der Übersetzungen erfolgt auf Laravel und Vue Seite fast gleich. Beide greifen auf die Variable locale zu, die in der app.php Konfigurationsdatei definiert ist. Diese wird von einer Middleware SetLocale beim Anfordern jeder Route aus dem ersten Segment der Anfrage, welches dem Sprachkürzel in der URL entspricht, mit der Methode setLocale() gesetzt. Dadurch reicht es, beim Wechseln der Sprache in der URL ein anderes Sprachkürzel einzugeben und die Seite erneut zu laden. Auf diese Weise funktioniert das Wechseln der Sprache in allen Teil-Anwendungen, was jeweils automatisiert über eine Spracheinstellungsseite erfolgt.

3.1 Datenbank

Die Grundlage der Datenbank ist bereits in den vorherigen beiden Semestern entstanden. Im Rahmen des Projekts wurde die Datenbank zunächst, basierend auf den Erfahrungen der Grundlage, neu strukturiert und dann um Funktionen, Tabellen und Tabellenspalten ergänzt. Verwendet wird eine MySQL-Datenbank, deren Tabellen dynamisch durch Laravel-Migrations erzeugt werden, damit die Erstellung der Tabellen schnell und automatisiert erfolgt, sowohl lokal zum Testen, als auch auf dem Server. Die Migration findet dabei mit dem Befehl php artisan migrate statt, der lokal manuell und auf dem Server beim Anlegen einer App-Instanz in einer Shell-Datei ausgeführt wird.

In Abbildung 5 ist ein Auszug des Entity-Relationship-Diagramms abgebildet, in dem alle selbst erstellten und verwendeten Tabellen aufgeführt sind. Darin spiegelt sich auch teilweise der Aufbau der mobilen Applikation wider. Einem Kapitel sind mehrere Unterkapitel und ein Test zugeordnet. Den Unterkapiteln sind wiederum Inhalte zugeordnet, die jeweils aus Text, Bild, Video, einer Frage oder einer Datei bestehen können. Texte und Fragen nehmen eine spezielle Rolle in der Art des Umgangs mit Inhalten ein. Da Texte in jedem Bereich der Applikation Verwendung finden, besitzt die Tabelle eine ’link’ und eine ’link id’ Spalte, in der die Verwendung des Textes angegeben wird.

Beispielsweise kann der Text mit link = ’Content’ und link id = 4 dem Inhalt mit der ID vier zugeordnet werden. Fragen haben dafür eine Spalte ’content id’, die falls es eine Frage ist, die im Inhalt verwendet wird, die ID des jeweiligen Inhalts enthält.

Fragen können außerdem einem oder mehreren Tests zugewiesen sein und können zusätzlich ein Bild oder Video enthalten. Zudem besitzt eine Frage, je nach Typ, mehrere Antwortmöglichkeiten, die in der Tabelle ’question choices’ auffindbar sind. Handelt es sich um eine Zuordnungsfrage, dann ist einer Kategorie zusätzlich eine Antwortmöglichkeit in der Tabelle ’assign choices’ zugewiesen.

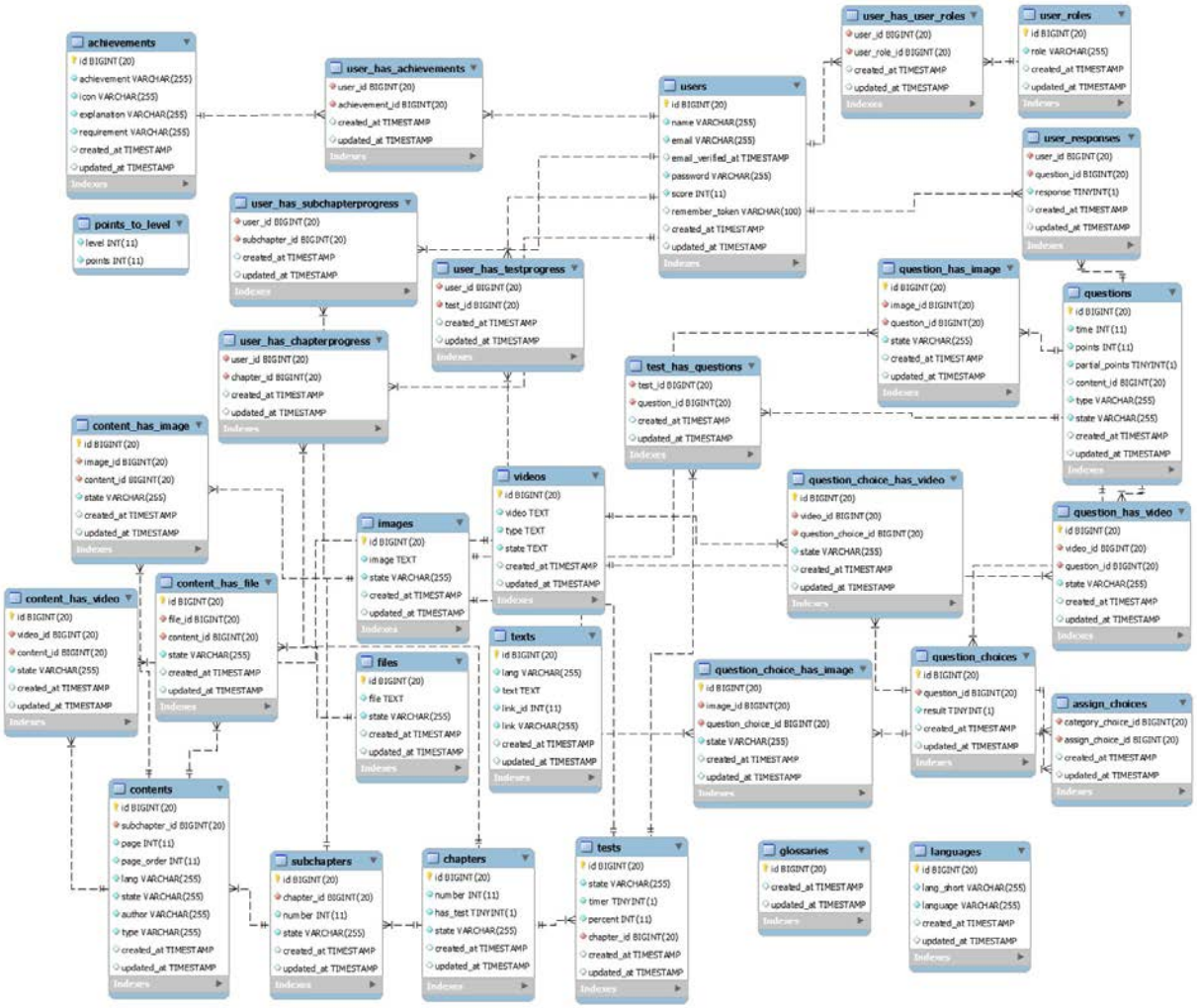

Abbildung 5: EER Model[7] der Datenbank

Ein großer Teil der Datenbank ist auf die Nutzer bezogen, wobei in der Nutzertabelle nutzerspezifische Informationen wie E-Mail-Adressen, verschlüsselte Passwörter oder aktuelle Punktzahlen gespeichert werden. Ein Nutzer kann mehrere Rollen zugeordnet bekommen, die in der 'user roles' Tabelle definiert sind und über die Tabelle 'user has user roles' zugewiesen werden.

Freigeschaltete Errungenschaften können über jeweilige Tabellen auf die gleiche Weise einem Nutzer zugewiesen werden. Der Fortschritt von Kapiteln, Unterkapiteln und Tests wird jeweils in einer Tabelle für jeden Nutzer gespeichert. Vom Nutzer beantwortete Fragen und die Ergebnisse werden ebenfalls in einer separaten Tabelle 'user responses' gespeichert.

[7] Enhanced entity–relationship model - ist in der Informatik ein übergeordnetes oder konzeptionelles Datenmodell, das Erweiterungen des ursprünglichen Entity-Relationship-Modells enthält, das beim Entwurf von Datenbanken verwendet wird

3.2 Login und Nutzerrollen

Alle Anwendungen der ALP sind durch einen Login gesichert. Damit ein Nutzer sich nicht für jede Instanz neu registrieren muss und die Passwörter nicht an mehreren Stellen gespeichert werden müssen, wird der Login über die im späteren Verlauf beschriebene Admin-Oberfläche zentralisiert. Will sich ein Nutzer bei einer App anmelden, wird mit der Admin-Oberfläche ein Sitzungstoken ausgehandelt. Dieser erlaubt es der Anwendung, die Nutzerdaten nach erfolgreicher Anmeldung von der Admin-Oberfläche anzufordern, hierzu gehören Name und E-Mail-Adresse. Anhand dessen werden dem Nutzer innerhalb der Instanz spezifische Rollen zugeteilt. Hat der Nutzer die für die angefragte Anwendung (z.B. Autoren-Oberfläche) notwendige Rolle beziehungsweise Berechtigung, wird diese ihm nun angezeigt. Andernfalls erhält er eine Fehlermeldung.

Die Admin-Oberfläche kann die Nutzer selbst (lokal) verwalten oder auf ein bestehendes Nutzerbackend (LDAP, Shibboleth) zugreifen. Die Zuweisung der Rollen für bestehende Nutzer erfolgt auf der dafür vorgesehenen Unterseite in der Autoren-Oberfläche. Die Rollen mit entsprechenden Berechtigungen sind in Tabelle 1 aufgeführt.

Tabelle 1: Übersicht der Nutzerrollen und deren Berechtigungen

Nutzerrolle	Berechtigungen
Admin	Zugriff Admin-Oberfläche; Verwaltung aller App-Instanzen; Erstellen neuer App-Instanzen
Owner	Alle Berechtigungen des Managers; Allgemeine App-Einstellungen; Zugriff auf Analyse-Tool
Manager	Alle Berechtigungen des Authors; Veröffentlichen von Inhalten
Author	Alle Berechtigungen des Users; Zugriff auf Autoren-Oberfläche; Bearbeiten und Erstellen von Inhalten; Zugriff auf Test-App
User	Zugriff auf mobile App

3.3 Admin-Oberfläche

Die Admin-Oberfläche ist ein Laravel-Projekt. Sie ist serverseitig modular aufgebaut. Die Kopfzeile befindet sich in einem Layout-Blade-File, während der Rest der Anwendung dynamisch eingesetzt wird. Je nach Route wird über den Controller das passende Blade-File inkludiert. Die nutzerseitige Liste der App-Instanzen wird dynamisch mithilfe von Informationen aus der Datenbank der Admin-Oberfläche erstellt. Die gespeicherten Informationen beinhalten unter anderem die App-URL, den Namen der App und Informationen über die aktuell installierte Softwareversion aller App-Instanzen.

Die Icons der Apps hingegen sind als Image-Elemente eingebunden, deren Quelle innerhalb der jeweiligen App-Instanz liegt. Dafür haben die Apps eine Route, die es erlaubt, das App-Icon auch ohne Anmeldung abzurufen. Neben der Übersicht befindet sich außerdem ein Button zum Erstellen einer neuen App-Instanz. Durch das Betätigen wird ein Formular geöffnet (Abbildung 6, rechts), in welchem die Instanz spezifischen Metadaten wie Name und URL eingetragen werden können.

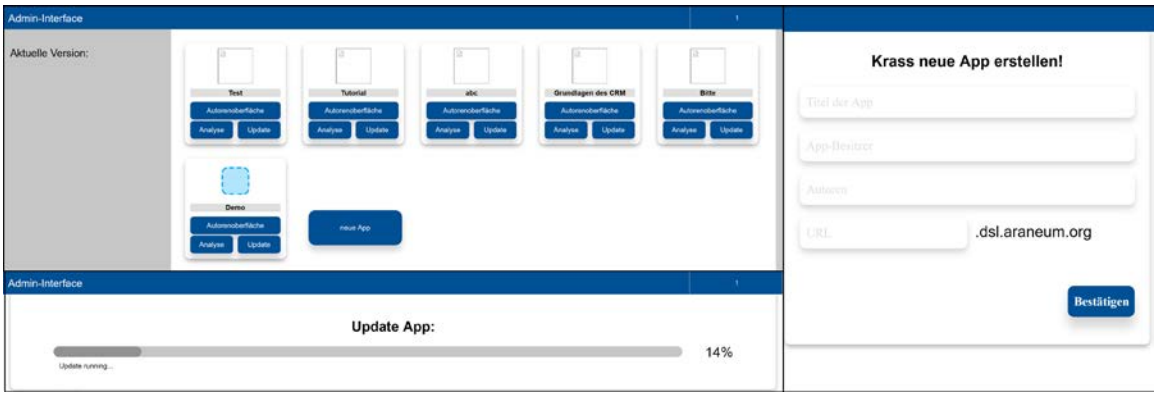

Abbildung 6: Ansichten der Admin-Oberfläche

Bestätigt man nach vollständigem Ausfüllen des Formulars, dass „Job" erstellt ist, wird dies in eine Datei gespeichert und auf dem Server abgelegt. Ein Cronjob prüft regelmäßig ausstehende Jobs und startet die entsprechenden Installationsroutinen, die in Bash-Files beschrieben sind. Dabei werden diese mittels debootstrap Container erstellt, welche anschließend gestartet, konfiguriert und für die Softwareinstallation vorbereitet werden. Beim ersten Start dieses Containers wird die Installationsroutine für Laravel ausgeführt und aus einer Git-Repository der für die Instanz notwendige Programmcode nachgeladen. Während dieses Prozesses schickt das Installationsskript regelmäßig den aktuellen Fortschritt in Prozent und eine Beschreibung des Installationsschritts an die Admin-Oberfläche. Der Prozentwert wird von einem Javascript-Snippet regelmäßig abgefragt und dem Nutzer grafisch in Form eines Fortschrittsbalkens präsentiert. Ergänzend wird der entsprechende Schritt in Schrift angezeigt.

Wird für eine App-Instanz ein Update ausgeführt, verläuft der Prozess ähnlich. Statt des Installationsskripts wird ein Updateskript ausgeführt, welches die initialen Installationen überspringt und anstelle dessen notwendige Updates zugrunde liegender Komponenten überprüft und ausführt. Auch während dieses Prozesses bekommt der Nutzer über einen Fortschrittsbalken den Stand verdeutlicht (Abbildung 6, unten links).

3.4 Autoren-Oberfläche

Die in der Admin-Oberfläche erstellten Apps können über die Autoren-Oberfläche bearbeitet und mit Inhalten gefüllt werden. Diese ist in einem separaten Laravel-Projekt realisiert, welches sowohl serverseitig in Blade-Files, als auch clientseitig durch Vue-Files modular aufgebaut ist. Die grobe Struktur der Anwendung ist im Blade Layout File festgelegt. In diesem ist die obere Navigationsleiste mit den Menüpunkten für die Unterseiten und dem Dropdown-Menu zum Umstellen der Sprache und Abmelden des Nutzers definiert. Der darunter befindliche Hauptteil der Anwendung wird dynamisch über Templating eingefügt. Dazu wird entsprechend der Route über den AuthorController das zugehörige Blade-File aktiviert. Dieses fügt einen bestimmten Trigger für Vue.js in das bestehende Layout-Template ein. Der Trigger-Tag sorgt dafür, dass nach vollständigem Laden der Daten die Vue-Komponente clientseitig in die Hauptsektion eingefügt und gerendert wird.

Inhalte

Wechselt man beispielsweise auf die Seite für das Erstellen und Bearbeiten von Inhalten, wird durch das Klicken des Links in der Navigation ein erneutes Laden der Seite veranlasst. Die hinter dem Link hinterlegte Route führt dann zu einer Methode im AuthorController, welche wiederum das entsprechende Blade-File verarbeitet. Dabei wird das Layout-Blade mit dem Trigger-Tag für das entsprechende Vue-File versehen. Daraufhin werden die Daten geladen, damit auf der Seite bereits vorhandene Inhalte angezeigt werden können. Dies geschieht, indem das Skript einen XMLHttpRequests an den Server schickt, die entsprechende API-Route verweist auf den APIController, welcher die Datenbank-Queries generiert. Die zurückgegebenen Daten werden an den Client geschickt und dort in Variablen gespeichert. Anschließend werden diese vom Vue-File verwendet, um den Trigger-Tag durch entsprechendes HTML zu ersetzen.

Wurde die Seite erfolgreich entsprechend der Daten gerendert, kann der Nutzer zunächst ganz links die Seitennavigation für Kapitel sehen. In dieser werden untereinander alle Kapitel mit Nummer und Titel aufgeführt, die in der Datenbanktabelle 'chapters' mit dem Status 'live' oder 'test' eingetragen sind.

Unter den Buttons der aufgelisteten Kapitel und des Glossars befindet sich ein Button, über den der Nutzer ein neues Kapitel anlegen kann. Beim Klicken auf diesen wird eine Methode ausgeführt, die einen modalen Dialog öffnet. Alle Dialoge basieren auf einer Komponente, die aus dem Paket vue-js-modal stammt. Ein Dialog wird geöffnet, indem der Methode this.$modal.show() der Name des Dialog-Elements übergeben wird. In diesem Element wird über alle Sprachen iteriert und für jede ein Eingabefeld für den Titel des Kapitels in der jeweiligen Sprache erstellt. Darunter befinden sich ein Speichern-Button und ein Abbrechen-Button, die beide den Dialog schließen. Beim Speichern werden vor dem Schließen des Dialogs die Titel an den Server geschickt und in der Datenbank gespeichert.

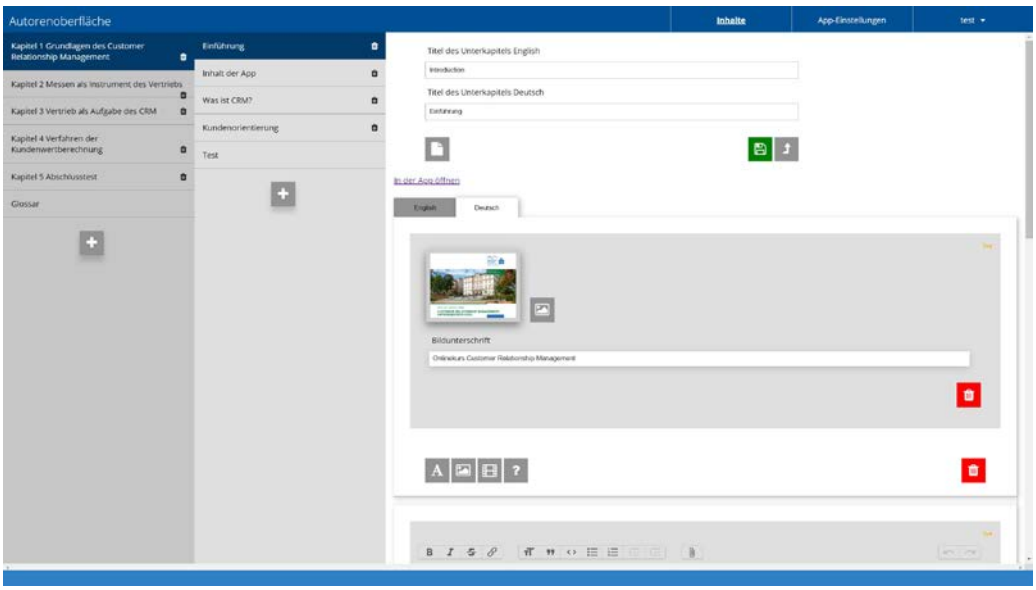

Abbildung 7: Ansicht der Inhalt-Seite in der Autoren-Oberfläche

Wählt man durch Klicken eines der Kapitel aus, öffnet sich rechts daneben eine weitere Navigationsspalte. Diese enthält analog alle Unterkapitel, die das ausgewählte Kapitel beinhaltet, also alle, die als Fremdschlüssel die 'chapter id' dessen und außerdem den Status 'live' oder 'test' besitzen. Im Hauptbereich der Anwendung erscheinen außerdem

Textfelder, welche das Bearbeiten des Kapitel-Titels ermöglichen. Nachdem der neue Titel eingetragen und der Speichern-Button betätigt wurde, wird der vorherige Eintrag in der Datenbank aktualisiert. Die Reihenfolge der Kapitel lässt sich einfach durch eine Drag-and-Drop-Funktion beliebig anpassen. Diese ist mithilfe einer Drag-and-Drop Komponente für Vue umgesetzt, welche auf der Bibliothek SortableJS basiert. Wählt man eines der Unterkapitel an, öffnet sich rechts die Oberfläche zum Bearbeiten der Inhalte dieses Unterkapitels (Abbildung 6). Oben findet man analog zum Bearbeiten des Kapitel-Titels auch hier Textfelder, um den Titel des Unterkapitels anzupassen. Darunter sind die Inhalte für das Unterkapitel in Tabs für die entsprechenden Sprachen aufgeteilt, zwischen denen sich durch einfaches Klicken auf die Sprache umschalten lässt.

Der Nutzer kann innerhalb dieser Tabs beliebig viele Seiten für das Unterkapitel anlegen und mit verschiedenen Inhalten, die frei kombiniert werden können, füllen oder bereits vorhandene Inhalte bearbeiten. Die Art des Inhalts wird beim Anlegen durch Betätigen des entsprechenden Buttons festgelegt und bestimmt so die angezeigten Bedienelemente. So wird für Texte beispielsweise ein Editor angezeigt, der alle rudimentären Textverarbeitungsfunktionen zur Verfügung stellt. Dieser Editor ist mithilfe der Vue-Trix Komponente umgesetzt.

Fügt man auf einer Seite ein Bild oder Video ein, so bekommt man einen Button zur Medienauswahl und ein Textfeld um eine Video- bzw. Bildunterschrift hinzuzufügen. Durch das Betätigen des Medienauswahl-Buttons lässt sich eine Galerie öffnen, jeweils für Videos oder Bilder. Diese Galerien sind auch im modalen Dialog umgesetzt. In den Galerien sind alle in der Instanz bereits hochgeladenen Videos oder Bilder in einer Galerie-Ansicht angezeigt. Darunter befindet sich ein Upload-Button, welcher vom Browser gesteuert wird. Dieser erlaubt dem Nutzer die Auswahl einer Bild- oder Videodatei, welche nach einer Bestätigung durch den Nutzer hochgeladen wird. Diese Datei wird von Laravel angenommen und in einen entsprechenden Ordner gespeichert. Während des Uploads wird der Button entsprechend des Fortschritts prozentual eingefärbt, um dem Nutzer diesen zu signalisieren. Danach kann der Nutzer die hochgeladene Datei in der Galerie-Ansicht sehen. Wählt er eines der Bilder durch Anklicken aus, sodass es bunt umrandet wird, und bestätigt seine Auswahl mit dem Speichern-Button, wird das entsprechende Bild dem Inhalt zugewiesen und ist nun über der Bildunterschrift in einer Vorschau zu sehen.

Bei Videos gibt es jedoch den Sonderfall, dass neben dem gerade beschriebenen Prozess für lokale Videos, auch der Video-Typ “YouTube-Video“ ausgewählt werden kann. In diesem Fall erscheint statt des Medienauswahl-Buttons ein Textfeld, in welches ein Einbettungslink des gewünschten YouTube-Videos kopiert werden muss. Daraufhin wird auch von diesem eine Vorschau generiert und im Inhalt angezeigt.

Der letzte Inhaltstyp, der zurzeit implementiert ist, sind die Fragen. Dabei ist der Prozess des Bearbeitens dem der Fragen eines Tests sehr ähnlich und es stehen auch dieselben Fragetypen zur Auswahl. Dies wird im Abschnitt zum Test (S. 15) noch detaillierter erläutert. Lediglich die Einstellungen für eine zeitliche Begrenzung und die Punktevergabe fällt weg, da die Fragen innerhalb der Inhalte zur Auflockerung und interaktiven Gestaltung dienen und nicht den Zweck des Überprüfens von Gelerntem erfüllen sollen, wie es die Tests tun. Um nachträgliche Änderbarkeit zu gewähren, lässt sich sowohl die Reihenfolge der Seiten als auch die Anordnung der Inhalte über Drag-and-Drop ändern.

Beim **Speichern des Unterkapitels** wird beim Klicken auf den entsprechenden Button eine Methode ausgeführt, die ein Promise-Element zurückgibt, nachdem das Speichern serverseitig abgeschlossen ist. An den Server werden mittels einer HTTP-Post Anfrage die Informationen des gewählten Unterkapitels und der Inhalte dieses Unterkapitels übergeben. Dabei ist zu beachten, dass das Speichern von Inhalts-Fragen in der jeweiligen Kind-Komponente erfolgt. Mit der Direktive ref können Kind-Komponenten referenziert werden und in diesen können über die Referenzen Methoden in den Komponenten ausgeführt werden. Somit wird die Speichern-Methode in allen Inhalts-Fragen-Komponenten separat ausgeführt.

Im entsprechenden Controller, der der API-Route zugewiesen ist, werden beim Speichern des Unterkapitels zunächst die Titel in den verschiedenen Sprachen gespeichert. Dazu wird über alle Titel iteriert und mit dem Laravel-Model Text jeweils ein neuer Text in der Datenbank angelegt. Die Informationen der Inhalte werden in geschachtelten Objekten, die in einer Liste befindlich sind, an den Server übergeben.

Beim Speichern der Inhalte wird über alle Sprachen, Seiten und Inhalte iteriert, welche in der Reihenfolge die Schachtelungsstufen darstellen. Dabei wird zunächst der Inhalt mittels dem Model Content anhand der ID geladen und mit den Änderungen gespeichert. Danach erfolgt je nach Inhaltstyp das Speichern der Texte, Bilder und Videos. Da lokal gespeicherte Bilder und Videos bereits beim Zuweisen in der Galerie gespeichert werden, müssen nur die Bild- bzw. Videobeschriftungen gespeichert werden, was analog dem Speichern der Texte über das Text Model erfolgt. Ausnahme ist das eingebettete Video, da es nicht über die Galerie zugewiesen wird, sondern das einzubettende iFrame Element gespeichert werden muss, welches mit dem Model Video in der Datenbank gespeichert wird.

Inhaltsfragen werden auf die gleiche Weise gespeichert wie Fragen der Kapiteltests. Diese werden im Controller in einer Methode gespeichert, die alle Informationen der Frage mit Hilfe der verschiedenen Models in die entsprechenden Datenbanktabellen speichert. Allgemeine Informationen der Frage, wie die Anzahl der Punkte oder die Zeit für die Frage werden in die 'questions' Tabelle gespeichert. Die Frage und die Erklärung, sowie Texte der Antwortmöglichkeiten werden in der 'texts' Tabelle gespeichert und die Informationen der Antwortmöglichkeiten werden in Abhängigkeit des Fragentyps auf unterschiedliche Arten in der 'question choices' Tabelle gespeichert.

Glossar

Auf der Glossar-Seite (Abbildung 8) sind alle Einträge in einer Tabelle angeordnet. In der linken Spalte befindet sich das zu definierende Wort und in der Spalte rechts davon die zugehörige Erklärung. Die Sprache, in der dies angezeigt wird, ist durch den aktuell eingestellten Sprach-Tab bestimmt, der ähnlich wie bei den Inhalten, hier die gesamte Tabelle einschließt. Rechts neben der Tabelle befinden sich für jeden Eintrag in der entsprechenden Zeile zwei Buttons: Zum Löschen des Eintrags bzw. zum Bearbeiten des Eintrags. Wählt man den Bearbeiten-Button oder den Button oben, öffnet dieser einen modalen Dialog. In diesem sind, ebenfalls in Tabs nach Sprachen sortiert, Textfelder eingebunden, um das Wort und die Erklärung einzutragen oder zu bearbeiten. Über den Speichern-Button kann man seine Änderungen bestätigen, die daraufhin in der Tabelle aktualisiert sind. Sollen die Änderungen nicht übernommen werden, kann der Abbrechen-Button gedrückt werden.

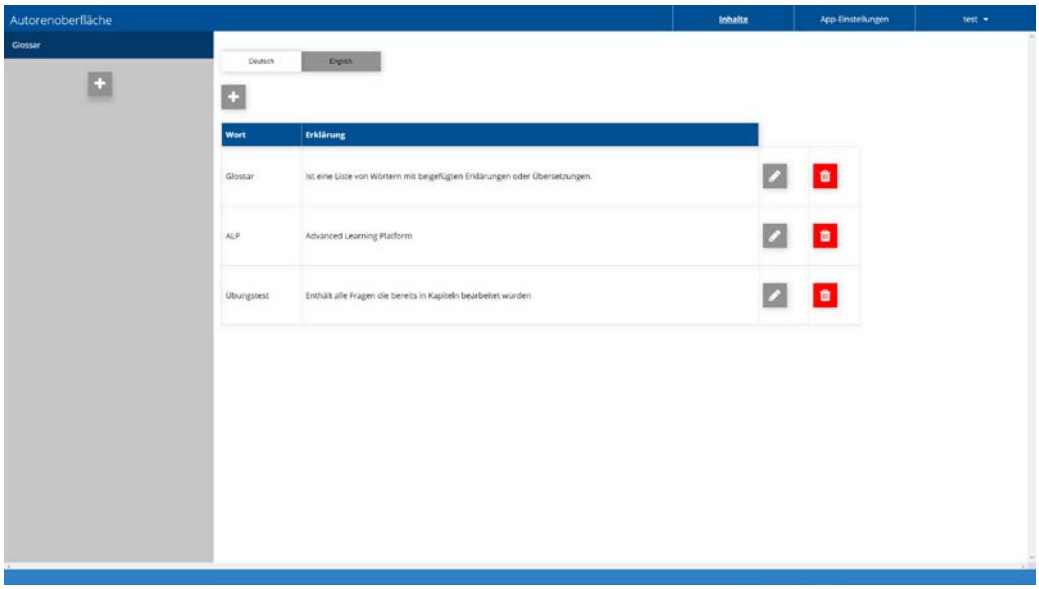

Abbildung 8: Ansicht der Glossar-Seite in der Autoren-Oberfläche

Test

Der Test zum Abschluss eines Kapitels ist immer als letztes Unterkapitel in der seitlichen Navigation zu finden. Soll kein Test in ein Kapitel eingebunden werden, lässt sich dieser durch einen Toggle bei der Bearbeitung des Kapitels ausschalten. Ein Test kann aus beliebig vielen Fragen bestehen. Diese sind untereinander aufgelistet, wenn ein Test zum Bearbeiten geöffnet ist. Darüber befinden sich allgemeine Einstellungen für den Test. Es kann beispielsweise festgelegt werden, ob bei der Bearbeitung des Tests die Fragen mit einer maximalen Zeit zur Beantwortung begrenzt sind und wieviel Prozent des Tests vom Nutzer richtig beantwortet werden müssen, um das nachfolgende Kapitel freizuschalten.

Darauf folgen für jeden möglichen Fragetyp Buttons, mit denen neue Fragen zugefügt werden können. Die einzelnen Fragen lassen sich zur weiteren Bearbeitung aufklappen (Abbildung 9).

Die dynamisch erstellte Fragemaske bietet die Möglichkeit, die zur Beantwortung zur Verfügung stehende Zeit, die durch Beantwortung möglichen erreichbaren Punkte sowie die Entscheidung zur anteiligen Punktevergabe bei multiplen Antwortmöglichkeiten einzustellen. Wie bereits bei der Bearbeitung der Inhalte, ist der textlastige Part der Frage in verschiedenen Tabs für die Sprachen sortiert. Unabhängig von der Art der Frage – welche beim Anlegen der Frage durch Klicken des entsprechenden Buttons festgelegt wird – finden sich zuerst Textfelder für die Eingabe der Frage und einer Erklärung, um die Frage in der Auflösung genauer zu spezifizieren und auf zusätzliche Lehrinhalte hinweisen zu können.

Darunter folgen, ebenfalls in einem zur Übersichtlichkeit ausklappbaren Abschnitt, die Antwortmöglichkeiten. Diese sind abhängig vom Fragentyp gestaltet:

- Sowohl bei einfachen Multiple-Choice-Fragen als auch bei Mehrfachwahl-Fragen wird für jede Antwortmöglichkeit ein Textfeld und ein Toggel zum Markieren der richtigen Antworten zur Verfügung gestellt.
- Bei einer Sortierfrage hingegen gibt es lediglich ein Textfeld für jede Antwortmöglichkeit, diese lassen sich jedoch über Drag-and-Drop in die richtige Reihenfolge bringen.

- Bei der Texteingabe-Frage können in einem Textfeld alle validen Schreibweisen für die richtige Antwort angegeben werden.
- Bei der Sortier-Frage ist das Eintragen der Antwortmöglichkeiten vergleichsweise komplexer, da hierbei die Kategorien festgelegt werden müssen, denen die Begriffe zugeordnet werden sollen und daraufhin die Begriffe eingetragen und einsortiert werden. Um auch hier möglichst viel Komfort beim Bearbeiten zu bieten, kann dies ebenfalls per Drag-and-Drop passieren.

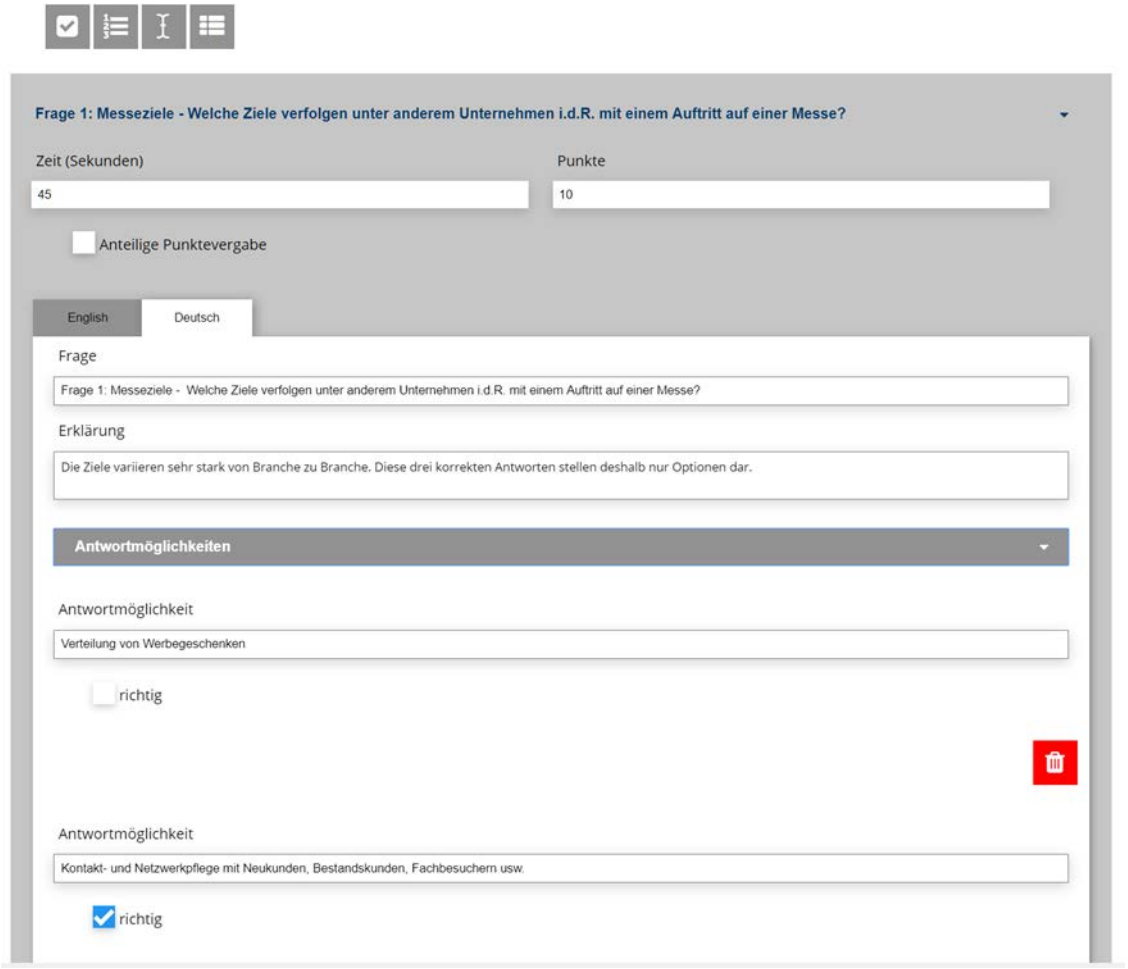

Abbildung 9: Ansicht des Bearbeitens einer Frage in der Autoren-Oberfläche

Nutzerverwaltung

Um Zugang zur Nutzerverwaltung zu besitzen, muss ein Nutzer die Rolle Owner besitzen. Nur dann wird in der oberen Navigationsleiste der zugehörige Button angezeigt. Dazu wird in der author.blade.php Datei um den Button herum eine @can Direktive verwendet, die überprüft, ob der Nutzer die erforderliche Autorisierung besitzt. Die Nutzerverwaltung ist über eine Vue-Komponente umgesetzt, die von der Blade-Datei, in dem sie eingebunden ist, die Nutzer und zugehörige Rollen als Prop übergeben bekommt. Diese Informationen werden in einer Tabelle angezeigt, indem über alle Nutzer iteriert wird. Über eine Suchleiste können die Nutzer gefiltert werden, um schnell den gewünschten Nutzer zu finden.

Die Zuordnung einer neuen Rolle erfolgt über entsprechende Buttons mit der Bezeichnung der Rolle, die beim Klicken eine Methode aufrufen, die die ID des Nutzers und die zuzuweisende Rolle übergeben bekommt und mittels einer POST-Request an den Server

schickt. Nach dem überprüfen, ob die Rolle valide ist und der Nutzer diese nicht schon besitzt, speichert die jeweilige Controller-Methode die Neuzuweisung in der Datenbanktabelle user has user roles. Das Entziehen einer Rolle erfolgt auf gleiche Weise mit der parallelen Löschung des entsprechenden Eintrages aus der Tabelle.

PDF-Export

Um Inhalte der Autorenoberfläche als PDF zu exportieren, wird die JavaScript Bibliothek jspdf verwendet. Beim Klicken auf den entsprechenden Button wird eine async-Methode aufgerufen, die mit der genannten Bibliothek ein neues PDF-Dokument erstellt. Dabei wird zunächst über alle Sprachen, dann über alle Unterkapitel, alle Seiten und alle Inhalte des aktuellen Kapitels iteriert. Diese Abstufungen werden im PDF durch eine Einrückung veranschaulicht. Größtenteils werden nur Texte ins PDF geschrieben. Bilder können neben Texten sinnvoll ins PDF eingefügt werden, weshalb die Methode async sein muss. Es muss gewartet werden, bis das Bild geladen wurde, was in einer anderen Methode erfolgt, die den Pfad zum Bild übergeben bekommt. Ein Promise-Objekt wird zurückgegeben, welches nach Laden des Bildes mit dem Base64-String, welcher das Bild repräsentiert, aufgelöst wird und dieses in das PDF einsetzt. Das Iterieren über die Fragen des Kapiteltests erfolgt ähnlich wie bei verschiedenen Inhaltstypen, indem je nach Typ die Inhalte verschieden eingerückt und aufgelistet werden.

3.5 Analyse-Tool

Das Frontend des Analyse-Tools ist ebenfalls in einem Laravel-Projekt umgesetzt (Abbildung 10). Die Funktionalität hingegen ist über die Installation eines Grafana-Servers realisiert. Parallel zur Installation der Admin-Oberfläche erfolgt die Installation einer Grafana-Instanz mit einer dazugehörigen InfluxDB. Daten, die bei der Benutzung der mobilen Applikation anfallen, können mit einem Zeitstempel versehen in dieser festgehalten werden. Dafür werden die Messungen von der Clientanwendung zunächst über eine API per HTTP-Request an den Server geschickt, welcher dann einen HTTP-Request mit den Daten an die InfluxDB sendet. Dieser Prozess ist üblich, um eine gewisse Datensicherheit zu gewährleisten, da so die Login- Daten der InfluxDB nicht im Client hinterlegt werden müssen.

Abbildung 10: Ansicht des Analyse-Tools

Für die Visualisierung der Daten in Grafana gibt es sogenannte Dashboards, eine Sammlung verschiedener Panels. Für jede App-Instanz wird ein solches Dashboard erstellt, dessen Bezeichnung der ID der App entspricht. Jedes dieser Dashboards beinhaltet dabei die gleichen Panels. Ein Panel stellt eine Messung oder die Zusammenfassung einer solchen grafisch dar. Hierfür gibt es in Grafana eine Auswahl an verschiedenen Darstellungsformen, wie zum Beispiel Balkendiagramme und Histogramme. Das Zusammenstellen eines Dashboards und Erstellen der einzelnen Panels erfolgt komfortabel über

die grafische Nutzeroberfläche von Grafana. In den Einstellungen der Grafana-Instanz kann die Existenz verschiedener Datenquellen eingetragen werden. So können neben der eigens dafür angelegten InfluxDB auch die MySQL-Datenbanken der App-Instanzen angebunden werden. Die eingetragenen Datenquellen können bei den einzelnen Panels ausgewählt werden und aus diesen können einzelne Parameter ausgewählt werden. Aus der resultierenden Zusammenstellung von Parametern baut Grafana eine Query. So können in jedem Panel verschiedene Messungen und Parameter-Daten dargestellt werden.

Die Anzeige des Grafana-Dashboards ist mit einer Vue-Komponente umgesetzt, die in einer Blade-View eingebunden wird. Der Aufbau ist ähnlich der Autorenoberfläche, da die optische Gestaltung und der Aufbau zu großen Teilen übernommen wurden (Abbildung 11).

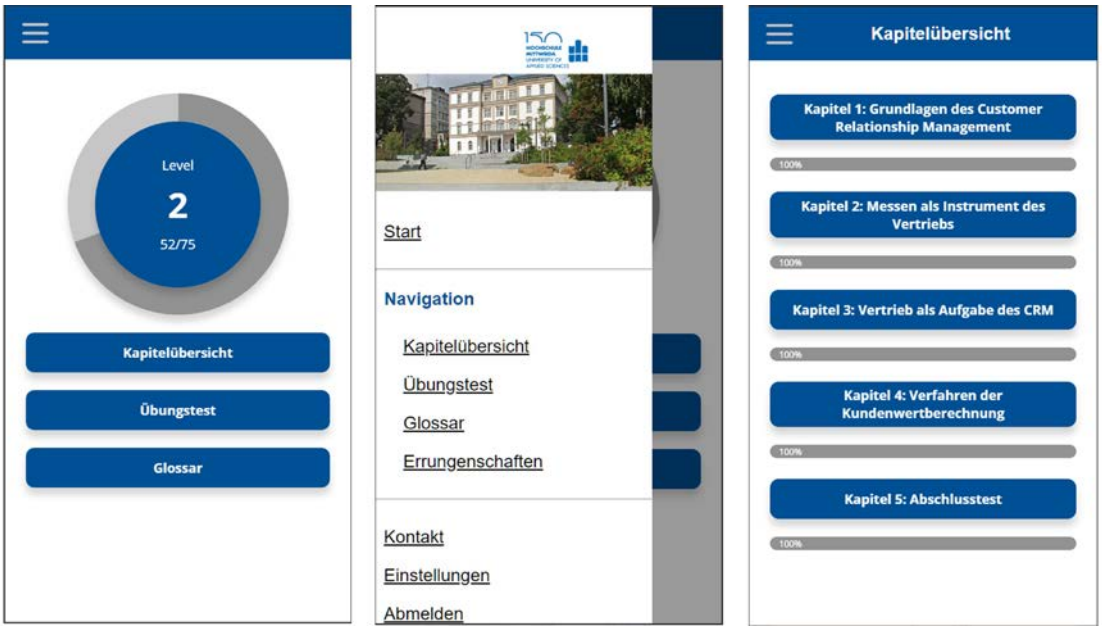

Abbildung 11: Ansicht Startseite, Seitenmenü, Kapitelübersicht der mobilen App

Die Einbindung des Dashboards erfolgt, indem ein iFrame-Element des jeweiligen Dashboards in der Vue-Komponente eingefügt wird. Da jede Instanz bei Erstellung eine ID erhält, mit der auch automatisch ein individuelles Dashboard angelegt wird, muss beim Angeben der Quelle des iFrames lediglich die App-ID, welche aus der .env -Datei gelesen werden kann, dynamisch übergeben werden.

Um die eingebundene Grafana-Instanz optisch an die restliche Anwendung anzupassen, können die CSS-Files dieser geändert werden.

3.6 Mobile Applikation

Die mobile Applikation ist ähnlich wie die Autoren-Oberfläche aufgebaut. Es gibt ein Laravel-Blade-Layout-File, in welchem die obere Navigationsleiste eingebunden ist, in der sich der Zurück-Button, der Button zum Öffnen des Seitenmenüs sowie der Titel der aktuellen Seite und das Seitenmenu befindet. Titel und Zurück-Button werden als Section aus der jeweiligen View bezogen, sodass der Titel immer zum angezeigten Inhalt passt und der Zurück-Button auch immer zur gewünschten, angegebenen Seite zurücknavigiert. Der eigentliche Inhalt wird aus dem jeweiligen Blade File eingebunden, welche im

Folgenden Views genannt wird und der zur aktuellen URL Route zugeordnet wurde. Beim Einbinden dieser Views muss das Eltern-Layout-File und die Sektionen angegeben werden sowie welche Inhalte in den zugehörigen Bereich des Layouts eingefügt werden sollen.

Startseite

Nach dem Anmelden wird der Nutzer im Normalfall zunächst auf die Startseite weitergeleitet. Auf dieser ist ein Fortschrittskreis zu sehen, der die Punkte des Nutzers sowie sein Level anzeigt. Außerdem befinden sich auf der Seite drei Buttons, um in andere Bereiche der Applikation zu gelangen. Diese Elemente sind alle direkt in das entsprechende Blade File eingebunden, wobei der Fortschrittskreis eine gesonderte Vue-Komponente ist. Diese wird in einem gesonderten Vue-File definiert und wird im app.js File als globale Komponente deklariert, sodass sie innerhalb der Applikation eingebunden werden kann. Diese Komponente bekommt die Punkte des Nutzers als Props (Properties) aus dem Laravel-Blade-File mittels der Authentifizierungs-Facade übergeben. Diese Punkte werden dann umgerechnet in das zugehörige Level. Die Liste, welcher Punktebereich zu welchem Level gehört, wird über eine HTTP Get Anfrage direkt über eine zugeordnete API-Route angefragt und vom zugehörigen Laravel-Controller beantwortet. Die Punkte und das Level des Nutzers können somit angezeigt werden.

Kapitelübersicht

Mit dem obersten Button unter dem Fortschrittskreis kann in die Kapitelübersicht navigiert werden (Abbildung 12). Beim Öffnen der Route wird eine zugehörige Methode in einem Laravel-Controller ausgeführt. In dieser werden zunächst alle Kapitel, nach Nummer sortiert, aus der Datenbank ausgelesen, die den Status 'live' besitzen. Zusätzlich wird für jedes Kapitel geprüft, ob der angemeldete Nutzer dieses Kapitel bereits freigeschaltet hat und zu wieviel Prozent das Kapitel schon bearbeitet wurde. Dazu werden ebenfalls die Fortschrittsinformationen aus den entsprechenden Datenbanktabellen herangezogen. Die Methode gibt die anzuwendende View zurück und übergibt dieser die Liste der Kapitel als Variable. In der View kann mit Hilfe der @foreach Direktive über alle Kapitel iteriert werden, wobei für jedes Kapitel ein Button und ein Fortschrittsbalken angelegt wird. Je nachdem, ob das jeweilige Kapitel freigeschalten ist, wird mittels einer @if Direktive der Button ausgegraut oder nicht.

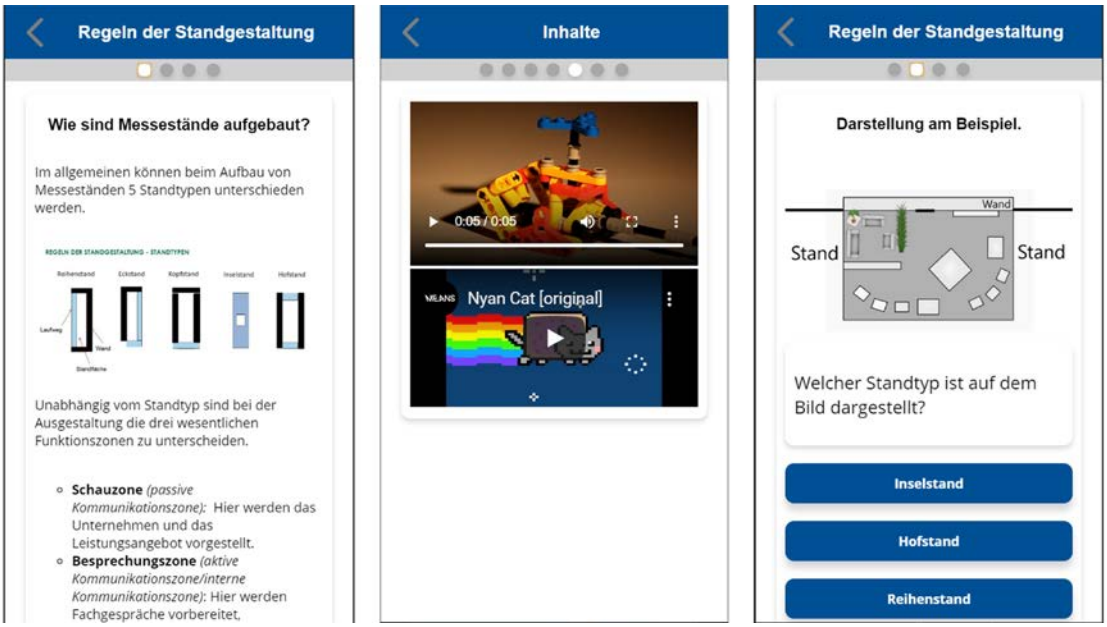

Abbildung 12: Ansicht Inhaltstypen der mobilen App

Unterkapitelübersicht

Ist ein Kapitel freigeschaltet, wird der Nutzer beim Klicken zur Unterkapitelübersicht weitergeleitet. Die ID des Kapitels wird als Parameter in der Route angegeben. Zum Beispiel öffnet.../chapters/1 die Unterkapitelübersicht zum Kapitel mit ID eins. Dadurch, dass nicht nur über die Buttons der Kapitelübersicht in eine Unterkapitelübersicht gelangt werden kann, sondern auch eine beliebige Nummer in die URL eingegeben werden kann, muss mit Hilfe einer Middleware geprüft werden, ob das angegebene Kapitel überhaupt existiert und falls es existiert, ob der Nutzer es bereits freigeschalten hat. Falls das Kapitel nicht existiert oder es nicht freigeschaltet ist, wird der Nutzer mit der jeweiligen Benachrichtigung zurück zur Kapitelübersicht geleitet. Für die Benachrichtigungen in Laravel wird das Paket uxweb/sweet-alert verwendet. Über die Hilfsfunktion alert() dieses Paketes kann das Aussehen, der Titel, der Anzeigetext und weitere Optionen der Benachrichtigung übergeben werden.

Existiert das Kapitel und der Nutzer hat es freigeschaltet, darf die angefragte Route betreten werden.

Die Unterkapitelübersicht funktioniert ähnlich der Kapitelübersicht. Es gibt ebenfalls eine Controller Methode, die alle Unterkapitel nach Reihenfolge sortiert aus der Datenbank liest. Zusätzlich wird auch der Kapiteltest abgerufen. Die Unterkapitel werden ebenfalls im View File iteriert. Ob ein Unterkapitel bzw. der Kapiteltest freigeschalten ist, wird über eine jeweilige Methode der User Klasse bestimmt. Beim Test gibt es mehrere Abfragen, da es auch möglich ist, dass ein Kapitel keinen Test enthält, ein Test existieren kann, aber nicht den Status 'live' besitzt und er kann noch nicht freigeschaltet sein. Ist der Test freigeschaltet, dann ist der zugehörige Button bedienbar. Sollte das nicht der Fall sein, ist der Button ausgegraut und in den anderen beiden Fällen wird kein Test-Button angezeigt.

Lehrinhalte

Wird auf den Button eines freigeschalteten Unterkapitels geklickt, wird eine Route wie zum Beispiel.../chapters/1/subchapter/2 angefragt. Dabei ist die eins die ID des Kapitels und die zwei die ID des Unterkapitels. Da es auch in diesem Fall wieder möglich ist, die URL manuell einzugeben, müssen sowohl Kapitel und Unterkapitel mittels jeweiliger Middlewares auf Existenz und Freischaltung überprüft werden.

Werden beide Middlewares erfolgreich durchlaufen, wird der Nutzer zu den Inhalten des gewählten Unterkapitels weitergeleitet (Abbildung 13). Dabei wird im Hintergrund wieder eine Methode eines Controllers ausgeführt, die zur Route zugeordnet ist. Es werden aus der Datenbank alle Inhalte ausgelesen, die den Status 'live' besitzen und als Sprache die aktuelle Sprache der Applikation angegeben haben. Abhängig davon, welchen Typ der jeweilige Inhalt besitzt werden über die Facaden Text, Image, Video und Question mittels individueller Methoden Inhalte aus den jeweiligen Tabellen bezogen und an die Controller Methode zurückgegeben. Die Inhalte werden nach Seiten aufgeteilt und innerhalb der Seiten nach Reihenfolge sortiert. Zusätzlich wird das nächste Unterkapitel ermittelt, falls dieses existiert, um die Route zum nächsten Unterkapitel generieren zu können. Diese Informationen werden an die View übergeben, wo sie als Props direkt weiter an eine Vue-Komponente übergeben werden. Die Komponente stellt den äußeren Rahmen der Inhalte dar und zeigt selbst noch keine Inhalte, sondern die Navigationspunkte zum Wechseln zwischen den verschiedenen Seiten. Außerdem befindet sich in dieser Komponente ein v-touch Element des Plugins vue-touch, welches ein Wrapper für Hammer.js ist, was unter anderem eine Bibliothek zur Erkennung von Touch-Gesten ist.[][HammerJS] Auf diesem Element wird mittels v-on ein Event-Listener für die Events swiperight und swipeleft registriert, die dann die jeweiligen Methoden aufrufen. In diesem v-touch Element befindet sich eine weitere Komponente, welche die individuellen Seiten verwaltet und ein Button, welcher genutzt wird, um zum nächsten Unterkapitel zu gelangen.

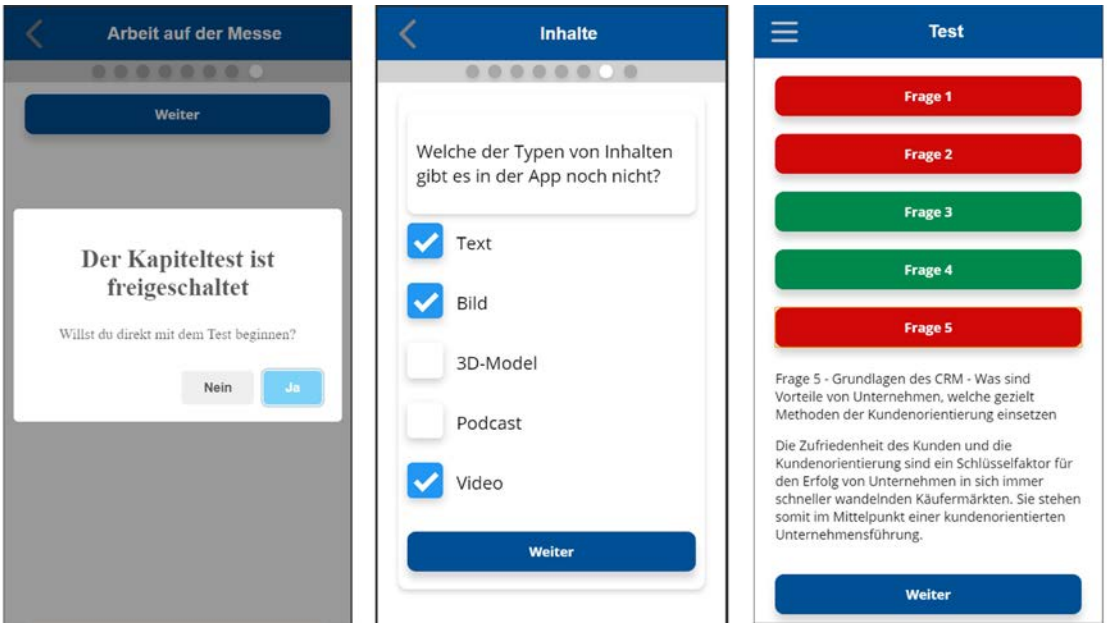

Abbildung 13: Ansicht Test der mobilen App

Wird die Seite über die Navigationspunkte oder eine Wischgeste gewechselt, ändert sich der Wert der aktuellen Seite, welche ein Objekt in der Liste aller Seiten ist. Die Informationen zu Inhalten einer Seite werden über Props an die Komponente übergeben. In der Komponente, welche die Seite repräsentiert, wird über alle Inhalte der Seite iteriert und je nach Typ des Inhalts wird die Information des Inhalts an eine weitere Komponente übergeben. Für jeden Inhaltstyp (Text, Bild, Video, Fragen) existiert eine individuelle Komponente.

Die Text-Komponente zeigt den übergebenen Text. Da es sich bei Inhaltstexten um einfaches HTML handelt, wird die v-html Direktive verwendet, sodass der Nutzer den formatierten Text sieht und nicht das rohe HTML. In der Bild-Komponente wird die Quelle aus einem vorgegebenen Datei-Pfad und dem individuellen Dateinamen an ein Bild-Element übergeben und der Text für die Bildunterschrift wird angezeigt. In der Video-Komponente wird überprüft, ob das Video lokal verfügbar oder eingebettet ist. Ist das Video lokal, wird wie bei der Bild-Komponente die Quelle an ein Video-Element übergeben. Handelt es sich um ein Video, welches eingebettet wird, dann ist als Information des Videos nicht der Dateiname, sondern zum Beispiel im Fall von YouTube ein iFrame mit den erforderlichen Informationen hinterlegt, welches angezeigt wird. Die Fragen-Komponente des Inhaltes enthält weitere Komponenten der einzelnen Fragetypen sowie Methoden, um alle Fragetypen auswerten zu können. Da diese Komponente ähnlich der Komponente ist, die für die Kapitel- und Übungstests verwendet wird, wird auf die Umsetzung der Fragen-Komponente später genauer eingegangen.

Zusammenfassend besteht die Inhaltsseite aus einer Inhalts-Komponente, welche das Wechseln der Seiten verarbeitet und nach Ende des Unterkapitels entweder zum nächsten Unterkapitel, direkt zum Kapiteltest oder zur Unterkapitelübersicht weiterleitet. In der Inhaltskomponente befindet sich die Seiten-Komponente, die die einzelnen Inhalte einer Seite verwaltet und diese je nach Inhaltstyp in separaten Komponenten anzeigt.

Kapiteltest

Der Kapiteltest funktioniert grundlegend wie die Inhalte. Wird die Route des Kapiteltests erfolgreich angefragt, ruft eine Controller-Methode die Informationen des Tests und der zugehörigen Fragen aus den jeweiligen Datenbanktabellen ab. Es wird über alle Fragen des Tests iteriert und je nach Fragentyp werden Antwortmöglichkeiten oder andere Informationen der Frage, wie die Fragetexte, die Erklärungen der Frage und zugehörige Bilder und Videos über Methoden der jeweiligen Klasse abgerufen.

Diese Informationen werden an das View Blade File und dort direkt weiter an eine Vue-Komponente übergeben. Diese Test-Komponente hat eine ähnliche Funktion wie die Inhalts-Komponente, da sie alle Fragen verwaltet, diese auswertet und den Ablauf des Tests regelt. Für jeden Kapiteltest kann eingestellt werden, ob die Fragen mit oder ohne Zeit bzw. Timer beantwortet werden sollen. Besitzt der Test einen Timer, wird dieser direkt nach dem Rendern der Seite gestartet. Der Timer funktioniert mithilfe der setInterval() Methode, durch die jede Sekunde geprüft wird, ob der Timer schon abgelaufen ist. Ist das der Fall, wird die aktuelle Frage mit einer ungültigen Antwort belegt, sodass diese falsch beantwortet ist. Nachdem das Intervall zurückgesetzt wurde, wird eine Methode aufgerufen, welche auf die nächste Frage wechselt und gegebenenfalls den Timer direkt neu startet. Gibt es keine weitere Frage, wird der Test beendet. Dafür gibt es eine

Methode, die das Testergebnis generiert. Bei der Auswertung werden die Punkte betrachtet, die der Nutzer für einzelne Fragen erlangt hat. Sind die erlangten Punkte prozentual über einem vom Ersteller des Tests vorgegebenen Grenzwert, gilt der Test als bestanden und es wird, falls existent, das nächste Kapitel freigeschaltet. Zusätzlich werden alle Antworten des Nutzers mittels einer POST-Anfrage an den Server geschickt. Es wird geprüft ob der Nutzer die Frage bereits richtig beantwortet hat, um keine doppelten Punkte für eine Frage zu vergeben. Die Anzahl der Punkte, die dem Nutzer gutgeschrieben werden, werden direkt als Antwort vom Server zurückgeschickt. Außerdem wird geprüft, ob der Nutzer durch diese Punkte im Level aufgestiegen ist und ob er sonstige Errungenschaften freigeschaltet hat. Dazu wird die Achievement Klasse genutzt. Diese Informationen werden ebenfalls in der Antwort des Servers zurückgegeben. In der Test-Komponente werden zwei weitere Komponenten eingebunden. Eine Frageseite-Komponente erhält die aktuelle Frage und leitet diese je nach Fragentyp an die zugehörige Komponente, die zu diesem Fragentyp passt. Aktuell gibt es die fünf Fragetypen Einfachauswahl, Mehrfachauswahl, Sortieren, Eingabe und Zuordnen. Diese werden auf sechs Komponenten aufgeteilt, da es für die Zuordnungsfrage eine Unterscheidung gibt, ob es zwei oder mehrere Kategorien gibt. Im Folgenden wird auf die verschiedenen Komponenten der Fragetypen allgemein eingegangen.

Fragetypen

Die Einfachauswahlfrage besitzt wie alle anderen Fragetypen ein Feld, in welchem der Fragetext angezeigt wird. Darunter wird über alle Antwortmöglichkeiten iteriert und für jede ein Button erzeugt. Wird der Button einer Antwortmöglichkeit geklickt, wird das Click-Event des Buttons ausgeführt und damit eine zugehörige Methode sendResponse(), welche die ID der Antwortmöglichkeit und die der Frage übergeben bekommt. Welche Funktion diese Methode genau hat, wird betrachtet, nachdem die anderen Fragetypen-Komponenten betrachtet wurden, da jede diese Methode enthält.

Bei der Mehrfach-Auswahlfrage sind statt den Buttons Checkboxen mit den entsprechenden Antwortmöglichkeiten beschriftet. Jedes Checkbox-Element ist über die v- model Direktive mit dem gleichen Array verbunden, der alle als richtig markierten Antworten bzw. deren IDs enthält. Zusätzlich gibt es unter den Antwortmöglichkeiten einen Button, mit dem der Nutzer die Bearbeitung der Frage beenden kann und seine Antwort finalisiert. Beim Klicken wird ebenfalls die sendResponse() Methode ausgeführt, welche die Liste mit den IDs der richtig markierten Antwortmöglichkeiten und die ID der Frage übergeben bekommt.

Um die Sortierfrage umzusetzen, wird die Komponente draggable genutzt. Dieser wird eine Liste mit Antwortmöglichkeiten übergeben. Damit können die Indizes der Antwortmöglichkeiten mittels drag-and-drop geändert werden, um diese in die richtige Reihenfolge zu bringen. Damit das funktioniert, muss die Liste mit Antwortmöglichkeiten in den Daten der Komponente verfügbar sein, da es über Props nicht möglich ist, diese zu mutieren. An die sendResponse() Methode wird die Liste mit vom Nutzer sortieren Antwortmöglichkeiten und die ID der Frage übergeben.

Für die Eingabefrage existiert ein Eingabefeld, in welches der Nutzer seine Antwort eingibt. Der Wert dieses Eingabe-Elements ist mit der v-model Direktive an eine Datenvariable gebunden, welche zusammen mit der ID der Frage an die sendResponse() Methode übergeben wird.

Wie schon erwähnt, gibt es für die Zuordnungsfrage zwei Komponenten. Gibt es nur zwei Kategorien, denen der Nutzer zugehörige Antworten zuordnen kann, wird dies mit Wischgesten umgesetzt. Funktionell ist es gleich mit dem Wischen der Inhaltsseiten. Je nach dem, in welche Richtung gewischt wird, wird die angezeigte Antwort zur links- oder rechtsstehenden Kategorie zugeordnet, indem die Antwort in eine Liste der Kategorie hinzugefügt wird. Daraufhin wird die angezeigte Antwort auf die nächste geändert. Gibt es keine weitere Antwort mehr, wird die sendResponse() Methode aufgerufen, welcher die Kategorien mit den zugewiesenen Antworten des Nutzers und der Frage ID übergeben wird.

Gibt es mehr als zwei Kategorien, werden diese in einer gesonderten Komponente untereinander aufgelistet und die zuzuordnenden Antworten werden in einem Pool ebenfalls aufgeführt. Mittels drag-and-drop können dadurch die Antworten den Listen der Kategorien zugeordnet werden. Dabei hat jede Kategorie und der Pool mit Antworten eine draggable Komponente. Da diese Listen nicht alle von Vue observiert werden, muss bei Änderungen in den Listen die Methode $forceUpdate() aufgerufen werden, um die A¨ Änderungen neu zu rendern. Über einen Button kann der Nutzer die Bearbeitung der Frage beenden und es wird die sendResponse() Methode, wie bei der Zuordnungsfrage mit zwei Kategorien ausgeführt.

Auswertung

Da es sich bei Vue zwischen Eltern- und Kind-Komponenten um einen Einwegdatenfluss handelt, können Kind-Komponenten nicht die Daten einer Eltern-Komponente ändern. Um die Daten von Kind- zu Eltern-Komponenten zu transferieren, werden Events verwendet. Dabei wird das Event in der Kind-Komponente emittiert und von einem Event-Listener der Eltern-Komponente entgegengenommen. Im Anwendungsfall der Fragen und des Tests müssen die Antworten des Nutzers aus der Komponente des jeweiligen Fragetyps in die Komponente des Tests übertragen werden, da die Fragen dort ausgewertet werden. Das ist die Aufgabe der sendResponse() Methode: Sie emittiert ein Event, auf das zunächst die Frageseiten-Komponente hört und aus dieser wird direkt ein weiteres Event emittiert, worauf wiederum die Test-Komponente hört. Je nachdem, aus welcher Komponente welchen Fragetyps das erste Event ausgelöst wurde, wird in der Test-Komponente eine Methode ausgeführt, um die jeweilige Frage des zugehörigen Fragetyps auszuwerten.

Bei der Auswertung der verschiedenen Fragetypen werden prinzipiell die richtigen Antworten, die vom Server beim Abrufen der Fragen mit übergeben werden, mit den Antworten des Nutzers verglichen. Zum Beispiel wird in der Auswertung einer Einfachauswahlfrage die ID der richtigen Antwort mit der ID der Antwort verglichen, welche der Nutzer gegeben hat. Das Ergebnis jeder Frage wird, u.a. in einem Objekt zu einer Ergebnisliste hinzugefügt. Bei Eingabefragen werden Strings bzw. Zahlen statt IDs verglichen, bei Sortierfragen wird die Reihenfolge der IDs der Antwortmöglichkeiten verglichen und bei Zuordnungsfragen wird überprüft, ob die zuzuordnenden Antworten zur korrekten Kategorie zugeordnet sind. Bei Mehrfachauswahlfragen ist es durch anteilige Punkte auch möglich, dass die Frage als richtig gewertet wird, wenn nicht alle richtigen Antwortmöglichkeiten vom Nutzer als solche erkannt werden. Dabei werden dem Nutzer nicht die gesamten Punkte der Frage angerechnet, sondern anteilige Punkte prozentuell von allen richtigen Antworten. Gibt es beispielsweise für eine Frage maximal zehn

Punkte bei zwei richtigen Antworten und der Nutzer wählt nur eine Richtige, dann erhält er fünf Punkte für diese Frage. Sind alle Fragen beantwortet, sieht der Nutzer eine Auswertungsseite mit Angaben, welche Fragen richtig oder falsch beantwortet wurden.

Dazu wird über die Ergebnisliste iteriert und jedes Ergebnis in einer Ergebnis-Komponente angezeigt.

Übungstest

Der Übungstest setzt sich aus allen Inhalts- und Kapiteltestfragen der Kapitel zusammen, die der Nutzer bereits freigeschaltet hat. Sie werden wie bei einem Kapiteltest vom Server abgerufen und an eine Vue-Komponente übergeben. Der Aufbau dieser Komponente ist ähnlich der des Kapiteltests. Allerdings werden die Komponenten der einzelnen Fragetypen direkt eingebunden, da die Seitenkomponente nicht benötigt wird und weil die Auswertung nach jeder Frage erfolgt und jede Frage somit für sich steht. Außerdem ist im Übungstest kein Timer enthalten und es werden dem Nutzer keine Punkte angerechnet.

Glossar

Das Glossar ist wie der Übungstest über die Startseite und das Seitenmenü zugänglich. Die Worte und Erklärungen werden direkt aus einer Vue-Komponente mittels einer HTTP-Anfrage vom Server abgerufen, wobei nur die Worte der aktuellen Sprache der Applikation berücksichtigt werden. Jedes Wort stellt ein einzelnes Element dar, welches sich beim Klicken öffnet, damit die Erklärung sichtbar wird. Diese Elemente sind alle über eine Vue-Komponente eingebunden. Die Suchfunktion ist mittels einer Computed-Property umgesetzt, die alle Worte filtert und nur die Worte als Liste zurückgibt, die den Such-String enthalten. Über diese Liste wird iteriert und die Information des Wortes an die Elementen-Komponente übergeben.

Test-Applikation

Innerhalb der Test-App können die Autoren Inhalte betrachten, die noch entwickelt werden, bevor sie veröffentlicht werden. Damit stellt die Test-App einen gesonderten Bereich in der Applikation dar, zu dem nur Autoren Zugriff haben, was mittels Laravel Middleware und Gates umgesetzt ist. Die Routen sind über den Prefix/test nach der aktuellen Sprache der App definiert. Die genutzten Laravel-Blade-Files, Vue-Komponenten und Laravel-Controller sind ähnlich der Live-App. Diese sind jedoch so abgewandelt, dass Inhalte mit den Status ’live’ und ’test’ sichtbar sind, aber der Nutzer keinen Fortschritt benötigt, um diese zu sehen und damit auch keine Kapitel freischalten kann. Dadurch sind alle Inhalte automatisch freigeschaltet und sichtbar und der Fortschritt des Nutzers wird nicht beeinflusst.

4. Evaluation

4.1 Ziel der Evaluation der mobilen App im Studienprozess

Ziel der Evaluation war die Erprobung der mobilen Applikation und deren Wirkung auf die Probanden. Dadurch sollten Erfahrungen im Bezug auf die Akzeptanz derartiger Lernanwendungen durch die Studierenden erfasst werden.

Die Evaluation des erstmaligen Einsatzes der mobilen App im Studienprozess gliedert sich in zwei Teilbereiche: die User Experience und die Funktionalität der Applikation.

Im Bereich der User Experience wird die Attraktivität und Benutzerfreundlichkeit der Applikation getestet. Die gewonnenen Ergebnisse werden bei der Verbesserung der Bedingung und grafischen Umsetzung einbezogen.

Bei der Funktionalität sollen mögliche Fehlfunktionen der Applikation aufgedeckt werden. Dazu sollen aus dem Feedback der Probanden Rückschlüsse gezogen werden. Zudem soll überprüft werden, ob die verschiedenen Funktionen erwartungsgemäß funktionieren.

4.2 Rahmenbedingungen und Vorgehensweise

Die Erprobung und Evaluation fanden am 21.01.2020 zwischen 11:30 und 13:00 Uhr in Haus 2 statt. Im Rahmen einer Prüfungsvorbereitung im Fach Wirtschaftsingenieurwesen des Studienganges Informatik testeten 42 Probanden die mobile Applikation der Advanced Learning Platform. Alle Probanden studierten im ersten Semester Informatik.

Inhalte in der Applikation betrafen das Thema Customer-Relationship-Management. Die Inhalte umfassten fünf Kapitel und es war geplant, dass die Bearbeitungszeit bei ca. 30 Minuten liegen sollte.

Innerhalb des Erprobungszeitraums wurde zunächst in 15 Minuten das Projekt zusammengefasst vorgestellt. Daraufhin wurden Zettel an alle Probanden verteilt, auf denen sich die Links und QR-Codes zur Applikation und zur AttrakDiff Umfrage sowie die individuellen Zugangsdaten zur Applikation und Platz für schriftliches Feedback befanden.

Alle Probanden konnten die Applikation für ca. 45 Minuten testen und die Inhalte bearbeiten. Im Anschluss wurden die Probanden gebeten, den AttrakDiff Fragebogen auszufüllen, was ca. 10 Minuten dauerte. Danach folgte eine Feedbackrunde, in der die Probanden nach Problemen der Applikation sowie Verbesserungsmöglichkeiten gefragt wurden, was ca. 20 Minuten dauerte.

Zum Testen der User Experience wurde ein AttrakDiff Fragebogen verwendet. AttrakDiff ist ein standardisierter Fragebogen zur Messung wahrgenommener hedonischer und pragmatischer Qualität von Software Produkten basierend auf wissenschaftlichen Grundlagen[8].

Die Bedeutung der Qualitäten besteht darin, dass gute Usability im Sinne einer optimalen Bedienbarkeit heute bei interaktiven Produkten vorausgesetzt wird. Ziel ist daher

[8] AttrakDiff

eine attraktive User Experience. Sie beschreibt das Nutzungserlebnis, das der Nutzer bei der Bedienung erfährt. Dem wird AttrakDiff gerecht.

Die hedonischen Qualitätsaspekte basieren auf den menschlichen Bedürfnissen nach Stimulation und Identität, während bei der pragmatischen Qualität der Bedarf zur kontrollierten Manipulation der Umwelt im Vordergrund steht. Stimulation meint dabei das menschliche Streben nach persönlicher Entwicklung, was durch Produkte stimuliert werden kann. Identität bedeutet in diesem Kontext, dass sich Menschen auch durch Objekte selbst zum Ausdruck bringen. Somit kann ein Produkt helfen, die gewünschte Identität zu kommunizieren.

Der Fragebogen besteht aus 21 Fragen, in denen die Probanden eine Tendenz zwischen zwei Attributen, wie z.B. stilvoll und stillos angibt. Abbildung 14 zeigt einen Auszug aus diesen Fragebogen.

Abbildung 14 - Auszug aus dem Attrakdiff Fragebogen[9]

In der Portfolio-Darstellung ist vertikal die Ausprägung der hedonischen Qualität zu sehen (unten = geringe Ausprägung). Horizontal ist die Ausprägung der pragmatischen Qualität zu sehen (links = geringe Ausprägung). Je nach Ausprägung der beiden Dimensionen fällt das Produkt in einen oder mehrere Charakterbereiche. Je größer das Konfidenz-Rechteck ausfällt, desto geringer ist die Sicherheit, mit der das Produkt einem bestimmten Bereich zugeordnet werden kann.

Ein kleines Konfidenz-Rechteck ist von Vorteil, da dies bedeutet, dass die Untersuchungsergebnisse mit höherer Sicherheit auf das Produkt zutreffen und weniger zufällig sind. Zudem spiegelt das Konfidenz-Rechteck auch wider, wie einig sich die Nutzer bei der Beurteilung des Produkts sind. Je größer das Konfidenz-Rechteck ist, desto unterschiedlicher wird das Produkt bewertet.

Im Diagramm der Mittelwerte sind die mittleren Ausprägungen der Dimensionen des AttrakDiff bei dem untersuchten Produkt dargestellt. In dieser Darstellung wird die hedonische Qualität zusätzlich nach den Gesichtspunkten Stimulation und Identität differenziert. Außerdem wird auch die Beurteilung der Attraktivität dargestellt.

Im Profil der Wortpaare sind die mittleren Ausprägungen der einzelnen Wortpaare des AttrakDiff für das untersuchte Produkt dargestellt. Hier sind vor allem Extremwerte interessant, da sie zeigen, welche Eigenschaften besonders kritisch oder besonders gut gelöst sind.

[9] Eigene Darstellung

4.3 Auswertung der Ergebnisse

Beim Testen der Applikation durften alle Probanden ihre eigenen Geräte verwenden. Von den 42 Teilnehmenden füllten 33 den AttrakDiff Fragebogen aus. In der Umfrage wurde eine zusätzliche Frage definiert, bei der die Probanden angeben sollten, welches Endgerät zum Testen verwendet wurde. Angaben zu den verwendeten Gerätetypen sind Tabelle 2 zu entnehmen.

Tabelle 2: Absolute Häufigkeiten der Gerätetypen

Gerätetyp	**Anzahl**
Android Smartphone	13
Laptop	6
Nicht spezifiziertes Smartphone	5
iOS Smartphone	5
iOS Tablet	4

Abbildung 14 enthält die Portfolio-Darstellung der durchgeführten Evaluation, vom 21.01.2020.

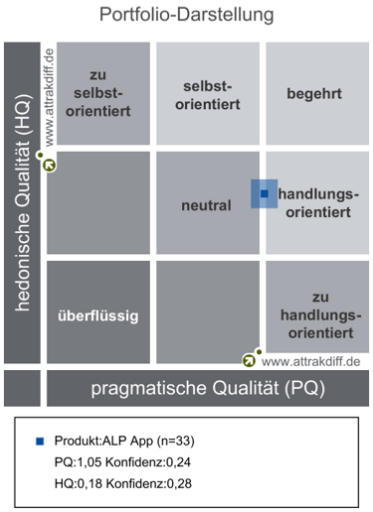

Abbildung 15: AttrakDiff Portfolio-Darstellungen vom 21.01.2020

Bei der Evaluation liegt die mittlere Ausprägung zwischen dem neutralem und handlungsorientiertem Bereich. Das kleine Konfidenz-Rechteck ist ein positiver Aspekt, da dies bedeutet, dass die Untersuchungsergebnisse mit höherer Sicherheit auf das Produkt zutreffen und weniger zufällig sind.

Die Benutzeroberfläche des Produkts wurde mit starker Tendenz als "handlungsorientiert" eingestuft. Diese Zuordnung ist für die pragmatische Qualität eindeutig. Die Anwendung ist also sehr pragmatisch. Das bedeutet, dass die Nutzer durch die App in der Anwendung bereits gut unterstützt werden.

Für die hedonische Qualität trifft die Charakterzuordnung ebenfalls recht eindeutig zu, da das Konfidenzintervall eindeutig im Charakterbereich liegt. Es zeigt sich, dass der Nutzer durch das Produkt zwar angeregt wird, allerdings erreicht die Ausprägung der hedonischen Qualität bei dem Produkt lediglich mittlere Werte. Es besteht

somit noch Verbesserungspotenzial hinsichtlich hedonischer Aspekte. Es besteht somit noch Verbesserungspotenzial hinsichtlich hedonischer Aspekte.

Im Diagramm der Mittelwerte (Abbildung 15) liegen alle Werte im durchschnittlichen Bereich, wobei die Werte für pragmatische Qualität und Attraktivität noch etwas höher liegen als die Werte der hedonischen Qualität. Daraus lässt sich schließen, dass die Applikation attraktiv auf Nutzer wirkt und sie durch die vorhandenen Funktionen bereits als nützlich angesehen wird.

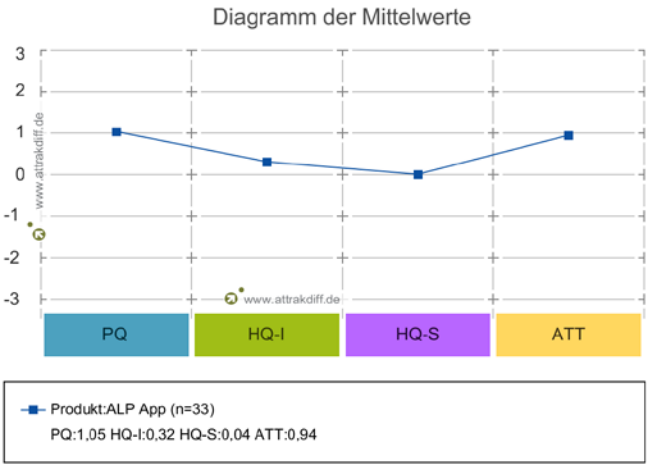

Abbildung 16: AttrakDiff Diagramm der Mittelwerte

Um die Werte der hedonischen Qualität zu erhöhen und damit den Nutzer mehr an die Applikation zu binden und mehr zu motivieren, könnten weitere Gamification Elemente eingefügt werden.

Das Profil der Wortpaare (Abbildung 16) zeigt, dass in der pragmatischen Qualität und der Attraktivität die meisten Wortpaare positiv bewertet wurden. Speziell sind die Extremwerte, da sie besonders kritische oder besonders gut gelöste Eigenschaften zeigen. Die negativen Ausschläge beziehen sich auf die Attribute „technisch versus menschlich“ und „trennt mich versus bringt mich näher“. Diese geben Hinweise, dass versucht werden könnte, Funktionen in die Applikation zu integrieren, welche Interaktionen zwischen Nutzern erzeugen.

Das Feedback der Probanden zeigt, dass die Applikation von den meisten Probanden gut angenommen wurde.

In großen Teilen der Aussagen der Probanden bestand das Feedback aus Verbesserungsvorschlägen, z.B. automatisches Weiterleiten zum nächsten Kapitel. Ca. zwei Drittel der Probanden sahen die Applikation als sinnvolle Unterstützung zum Unterricht, was den Ansatz des Blended Learnings und Mobile Learnings bestätigt.

Einige Wünsche bezogen sich auf neue Features, wie zum Beispiel eine Exportfunktion der Aufgaben, die sich ca. 42% der Probanden wünschten. Andere Aussagen bezogen sich auf die Inhalte, wobei sich 50% der Probanden Video-Inhalte wünschten.

Eine ausführliche Liste des Feedbacks aus der aktuellen Evaluation ist im Anhang1 zu finden. Die gesamten Ergebnisse der Feedbackrunde sowie individuelles, schriftlich eingereichtes Feedback einiger Probanden sind zusammengefasst im Anhang 2 dargestellt.

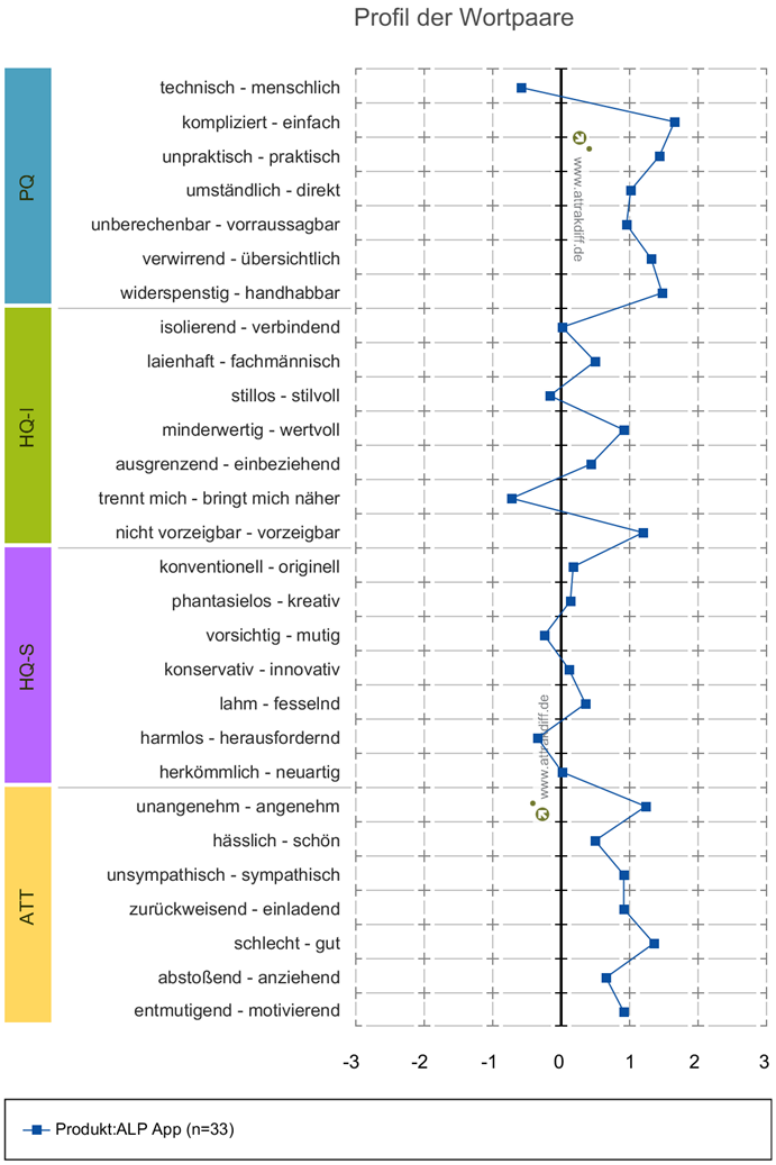

Abbildung 17: AttrakDiff - Profil der Wortpaare

5. Fazit

Im Rahmen des der App-Entwicklung wurde die bestehende mobile Applikation sowie die Autoren-Oberfläche in das Laravel-Framework umgezogen. Zusätzlich zu diesen Bestandteilen wurde die ALP um eine Admin-Oberfläche für die Verwaltung und Erstellung von Instanzen und ein Analyse-Tool, für welches die Grundvoraussetzungen geschaffen wurden, ergänzt. Die mobile Applikation wurde erweitert, sodass zum Beispiel Inhalte verschiedener Typen auf einer Seite angezeigt werden können und mehr Fragetypen verfügbar sind. Ebenfalls wurde die Autoren-Oberfläche erweitert, wodurch zum Beispiel das Einfügen von Multimedia-Inhalten durch Galerien erleichtert wurde oder das Glossar über die Autoren-Oberfläche bearbeitet werden kann.

Die mobile Applikation ist im aktuellen Zustand noch nicht für den finalen Einsatz, bedingt durch Integrationshürden im Zuge der Corona Pandemie vorbereitet. Die Ergebnisse zeigen jedoch, welche Qualitätsparameter für dieses Instrument der digitalen Lehre von Relevanz sind.

Mobile Applikationen können, wie der Test zeigte, ein wertvoller Baustein im Kontext der dezentralen, asynchronen Lehre sein. Vor allem die Unabhängigkeit der Endgeräte und die Option der Mehrsprachigkeit können hier als wichtiger Aspekte in der Ansprache Weiterbildungsstudierender gesehen werden. Die bisherigen Ergebnisse stellen nur einen ersten Schritt dar, welcher jedoch bereits jetzt die Vermutung nahelegt, dass die sich ergebenden Potenziale positive Resultate erzielen können.

Ferner zeigt sich, dass die interdisziplinäre Kooperation mit Partnern der Hochschule Mittweida bei der Bewältigung komplexer Problemstellungen einen erheblichen Mehrwert bietet. Somit empfiehlt es sich auch künftig, durch die Bündelung von Kompetenzen und Ressourcen die Herausforderungen des „Lebenslangen Lernens“ als systemübergreifende Aufgabe zu verstehen und gemeinsam an Lösungskonzepten zu arbeiten.

Um die ALP zu verbessern, sollen in Zukunft die geplanten Features – die noch nicht umgesetzt wurden – ergänzt und das Feedback aus den Evaluationen berücksichtigt werden. Besonderes Augenmerk liegt dabei weiterhin auf der Nutzerfreundlichkeit, weshalb beispielsweise die Gestensteuerung erweitert und die optische Gestaltung angepasst werden sollen.

Um den Lernerfolg zu optimieren, ist außerdem eine Überarbeitung der Auswertung von Tests und eine Ausarbeitung des Analyse-Tools notwendig. Um bestmögliche Performance zu erhalten und die Möglichkeit zum Offlinebetrieb zu geben, soll die mobile Webapp außerdem zu einer Progressiven Web App (PWA) erweitert werden. Somit soll ein qualitativer Standard für die Umsetzung in anderen Lehrformaten in Szenarien des Mobile Learnings geschaffen werden.

6. Ausblick und Verwertung

Vor allem in dezentralen, berufsbegleitenden Studienszenarien sowie in Kontext der internationalen Kooperation kann die dargestellte Variante des Mobile Learnings als barrierearme Plattform für die Lehr- und Lerngestaltung genutzt werden. Durch die Offenheit und Modularität der Anwendung bietet es sowohl den Lehrenden, als auch Lernenden eine zielorientierte Ergänzung zur Anreicherung der asynchronen Wissensvermittlung.

Den Lehrenden bietet sich die Möglichkeit repetitive Lehr- und Lerninhalte in einer attraktiven Form aufzubereiten und in das individuelle Lehrkonzept aufzunehmen. Durch die Integration der Test- und Analysetools besteht die Option, bezogen auf die Gesamtgruppe der Lehrenden, den Fortschritt beim Wissenserwerb in Echtzeit zu überprüfen und proaktiv Defizite zu identifizieren, welche in anschließenden synchronen Formaten aufgegriffen werden können. Durch diese zusätzlichen Informationen, können Lehreinheiten nutzerorientiert vorbereitet werden, was einen Mehrwert für die Qualität der Lehre haben kann. Die Möglichkeit Inhalte gespiegelt auch in anderen Sprachen, sowie andere Medienformate zu integrieren, steigert sich zusätzlich den Mehrwert, da in dieser strukturierten Umgebung das volle Potenzial der multimedialen Lehre mit einfachen Mitteln umgesetzt werden kann.

Die Lernenden profitieren, abhängig von den individuellen Rahmenbedingungen, im Speziellen von der Flexibilität und Endgeräteunabhängigkeit. Vor allem Letzteres stellt

im Zeitalter sich schnell wandelnder Endgeräte und Nutzerbedürfnisse einen erheblichen Mehrwert dar. Die Nutzenden können das internetfähige Gerät verwenden, was ihnen gefällt oder schlicht zur Verfügung steht. Dadurch wird niemand ausgeschlossen und alle Lernenden erhalten ein optimales Nutzererlebnis. Das Fehlen der Systemgrenzen, bei gleichzeitig hoher Usability, sowie die verschiedenen Einstelloptionen, bieten folglich allen ein motivierendes Lernerlebnis. Ziel muss es sein, dass die Lernenden die Applikation mit Freude verwenden, was aus Sicht der bisherigen Testergebnisse auch der Fall ist.

Literaturverzeichnis

AttrakDiff (2020) - User Interface Design GmbH, online verfügbar unter: http://www.attrakdiff.de/ Abrufdatum: 13.01.2020

de Witt, Claudia (2013): Vom E-Learning zum Mobile Learning - wie Smartphones und Tablet PCs Lernen und Arbeit verbinden: In: de Witt, C., & Sieber, A. (Hrsg.): Mobile Learning. Potenziale, Einsatzszenarien und Perspektiven des Lernens mit mobilen Endgeräten, Heidelberg (Springer) 2013, S.13-26. Online verfügbar unter: http://www.weiterbildungsblog.de/2013/07/23/vom-e-learning-zum-mobile-learning-wie-smartphones-und-tablet-pcs-lernen-und-arbeit-verbinden/ zuletzt geprüft am 15.11.2019

Erpenbeck, John; Sauter, Simon; Sauter, Werner: (2015) E-Learning und Blended Learning: Selbstgesteuerte Lernprozesse zum Wissensaufbau und zur Qualifizierung, Springerverlag Wiesbaden

Hug, Theo (2005): Micro Learning and Narration: Exploring possibilities of utilization of narrations and storytelling for the designing of micro units" and didactical micro-learningarrangements. online verfügbar unter: http://hug-web.at/drupal/sites/default/files/2005_Microlearning-and-Narration_Hug.pdf, Abrufdatum: 13.01.2020

Karran, Terence et al. (2003): Mobile Learning: Passing Fad or Pedagogy. In: Kynäslahti, H.: Mobile Technologies and Learning: Helsinki, ITPress: 51-61

mmb Institut (2020) - mmb Learning Delphi 2019/2020 – KI@Ed noch nicht in der Fläche angekommen; online verfügbar unter: https://www.mmb-institut.de/mmb-monitor/mmb-trendmonitor/ Abrufdatum: 13.01.2020

Abbildungs- und Tabellenverzeichnis

Abbildungen

Tabellen

Anhang

Ergänzende Dokumentationen der Evaluation

Anhang 1: Feedback der Fragerunde

- Überschriften in blauen Balken sind nicht zentriert
- Lösungen werden nicht richtig angezeigt bei der Auswertung der Tests
- automatisches Weiterleiten zum nächsten Kapitel hinzufügen
- Teilweise fehlen entsprechende Bilder in einzelnen Aufgaben
- Bilder sind teilweise zu klein
- App funktionierte bei 3 Anwendern nicht, davon zwei iPhone-Besitzer und ein Samsung
- unverschlüsselte Datenübertragung -> 6 von 43 Probanden legen auf Datensicherheit wert
- Zoom beim Eingabefeld löste Probleme im Weiterklicken der App aus
- Bei einer Farbschwäche der Probanden sind die als richtig und falsch farblich markierten Antworten (rot/grün) unglücklich gewählt, vielleicht besser mit fett, kursiv oder Symbolen wie Häkchen und Kreuz gestalten
- Probanden, die mit einem Laptop am Test teilgenommen haben, empfanden die Bedienelemente zu klein dargestellt
- Ein Weiterkommen per Touch ("wischen") ist nur im Textfeld der Aufgabe möglich, nicht darunter
- Probanden sahen zum Großteil einen Mehrwert in der App (1) als unterstützend zum Unterricht (zwei Drittel), (2) als Ersatz für Präsenzunterricht (Theorie der Lehrveranstaltung ausschließlich digital per App) ein Drittel
- Folgende Medienformate können sich die Probanden in der App noch vorstellen bzw. wünschen sich diese
    - Videos ca. 50%
    - Exportfunktion der Aufgaben ca. 42%
    - Notizfunktion 35%
    - Community 35%
    - Podcasts ca. 23%
- Help Button mit E-Mail an Professor 1 Proband
- 7 Probanden könnten sich vorstellen in Zukunft selbst an einer App wie dieser zu arbeiten
- Videolängen empfanden bis 3 Min. 5%, bis 10 Min. 50%, mehr als 10 Min. 12% der Probanden als angenehm
- Zusätzlich sollte es eine Regulierung für die Geschwindigkeitseinstellung bei Videos geben
- Einzelinhalte a` 5 Minuten als Learning Nuggets sind ebenfalls gewünscht
- Lösungen für Aufgaben (Feedback benötigen ca. 50% der Probanden)
- Es sollte eine Einstellmöglichkeit geben, ab wann die Lösungen angezeigt werden sollen (nach jeder Frage, nach 5 Fragen, usw.)
- 14% finden, dass der Wissensstand innerhalb der Seminargruppe sichtbar gemacht werden sollte
- 5% findet ein Rankingsystem interessant
- 25% könnten sich eine Nutzung der App während einer Präsenzveranstaltung vorstellen und 66% eine Nutzung in der Selbstlernphase

Anhang 2: Individuelles Feedback der Teilnehmenden

- offensichtlicher, wenn man etwas richtig gemacht hat
- abgeschnittene Worte (iPhone 8 Safari)
- bessere Navigation Formatierung von Text
- native App, damit Adresszeile nicht so viel Platz wegnimmt
- Bilder beim Übungstest fehlen
- Antworten bei Multiple Choice-Fragen besser formatieren
- Formatierungsfehler
- Test: nur eine der Antworten 0% der Punkte? warum nicht 50%?
- Home-Button
- Auswertung bei Tests nicht rot-grün, sondern mit Icons oder unterschiedlichen Schriftstilen