ERNST-ABBE-HOCHSCHULE JENA

Fachbereich Wirtschaftsingenieurwesen

Studiengang E-Commerce

Bachelorarbeit zum Thema :

Konzeption und Implementierung von Push Notifications für eine mobile App unter besonderer Berücksichtigung des Datenschutzes

Zur Erlangung des Grades: Bachelor of Science

Vorgelegt von:

Moritz Karsch,

Stauffenbergstr. 146,

07318 Saalfeld,

moritz_karsch1@yahoo.de

Matrikelnummer: 638475

Fachsemester: 10

Abgabedatum: 25.09.2019

Verteidigung: 30.09.2019

Erstgutachter: Prof. Dr.-Ing. Michael Stepping

Zweitgutachter: Simon Wolf

Referat:

Die vorliegende Arbeit befasst sich mit der Ausarbeitung und Umsetzung eines serverseitigen Service für die EAH Jena App. Das Hauptziel dieses Projektes ist die Schaffung einer Schnittstelle für die App, damit Nutzer sich einen individuellen Stundenplan zusammenstellen können und über Änderung via Push Benachrichtigungen informiert werden. Im Blickpunkt dieser Arbeit stehen dabei besonders die Thematiken Datenschutz und Systemsicherheit.

Inhaltsverzeichnis

Nomenklatur III
Abbildungsverzeichnis IV
Listingsverzeichnis V
Tabellenverzeichnis VI
1 Einleitung 1
1.1 Problemstellung 2
1.2 Struktur dieser Arbeit 3
2 Rahmenbedingungen 4
2.1 Ausgangssituation 4
2.2 Zielsetzung 6
3 Analyse 8
3.1 Anforderung 8
3.2 Nutzbarkeit der Stundenplan Daten 10
4 Konzeption der Systemarchitektur 15
4.1 LAMP-Architektur 15
4.2 Docker 17
4.3 Technische Aspekte 19
5 Analyse von Angriffsmöglichkeiten 26
5.1 Distributed Denial-of-Service Attacke 26
5.2 SQL-Injection 27
5.3 Absicherung der Datenbank durch Verschlüsselung 31
6 Push Benachrichtigung 33
6.1 Push vs. Pull Prinzip 33
6.2 Google Firebase Push Notification Service 34
6.3 Senden einer Push Benachrichtigung 36
7 Datenschutz 38
7.1 Datenschutzgrundverordnung 39
7.2 Grundsätze 39
7.3 Einschätzung des geplanten Service 43
8 Umsetzung 44
8.1 Einstellung 44
8.2 Datenbank 45
8.3 Interner Prozess Ablauf 46

8.4 Umsetzung der Schnittstelle 50
8.5 Probleme und Bewertung 54
8.6 Testen des Service 55
8.7 Bewertung der Umsetzung 56
9 Ausblick und Fazit 57
9.1 Ausblick 57
9.2 Fazit 58
Literaturverzeichnis 59
Anlagen 61
Ehrenwörtliche Erklärung und Einverständniserklärung 62

Nomenklatur

Abb.	Abbildung
BDSG	Bundesdatenschutzgesetz
bspw.	beispielsweise
bzw.	beziehungsweise
DBMS	Datenbank Management System
DSGVO	Datenschutzgrundverordnung
etc.	et cetera
LAMP	Linux Apache MariaDB PHP
PHP	Hypertext Preprocessor
u.a.	unter anderem
vgl.	Vergleiche
HTML	Hypertext Markup Language
z.B.	zum Beispiel
XML	Extensible Markup Language
iCal	iCalendar
UTF-8	8-Bit UCS Transformation Format
REST	REpresentational State Transfer
cURL	Client for URLs

Abbildungsverzeichnis

Abbildung 1 Stundenplan Aufbau EAH Jena | Quelle: eigene Abbildung 4
Abbildung 2: Zusammenspiel der Serverkomponenten in einer LAMP-Architektur 16
Abbildung 3: Architektur von Docker | Quelle: (Docker Inc, 2019).....18
Abbildung 4: Beispiel einer Request/Response-Session |Quelle: ((vectorsoft), 2012).....19
Abbildung 5 Sichere HTTPS Verbindung im Browser | Quelle: eigene Abbildung 21
Abbildung 6: HTTP vs. HTTPS | Quelle (Cloudflare, 2019).....22
Abbildung 7: Informationsflüsse im Internet | Quelle: (Singh, 2019).....33
Abbildung 8 Datenbankaufbau 45
Abbildung 9 Workflow Eventänderung 48

Listingsverzeichnis

Listing 1 - Aufbau der EAH Jena OPML Datei 11
Listing 2 - Aufbau einer Kalenderdatei 13
Listing 3: Dockerfile zum Bilden eines Images 18
Listing 4: Beispiel JSON-Datei 24
Listing 5: Beispiel einer SELECT-Abfrage 28
Listing 6: angepasste SELECT-Abfrage 28
Listing 7: Übergabe einer Variable einer SELECT-Abfrage 28
Listing 8: Beispiel einer WHERE-Injection 29
Listing 9: angepasste WHERE-Injection 29
Listing 10: Beispiel einer INSERT-Anfrage 30
Listing 11: angepasste INSERT-Injection 30
Listing 12: Beispiel zur Anlage einer verschlüsselten Tabelle 31
Listing 13: Beispiel eines verschlüsselten INSERT-Statements 31
Listing 14: Beispiel eines verschlüsselten SELECT-Statements 31
Listing 15 Beispiel Push Benachrichtigung via cURL 37
Listing 16 Beispiel vom unangepassten regulären Ausdruck 47
Listing 17 Beispiel vom angepassten regulären Ausdruck 47
Listing 18 SELECT Anfrage mithilfe von LOCATE 49
Listing 19 Senden einer Push B. mithilfe der php-FCM Bibliothek 49
Listing 20 Abhängigkeit Datenbank und Verschlüsslung 50
Listing 21 Beispiel /change 51
Listing 22 JSON Beispiel für einen Body - /changes 51
Listing 23 Verschlüsslung, Entschlüsselung 52

Tabellenverzeichnis

Tabelle 1: Übersicht http-Methoden.....20
Tabelle 2 HTTP Statuscodes.....21

1 Einleitung

Das Smartphone ist der tägliche Begleiter von vielen Personen in Deutschland. Dieses mobile Endgerät wird genutzt, um Daten jeglicher Art mobile und jederzeit abrufen, verwalten und ändern zu können. Jährlich steigt die Nutzeranzahl von Smartphones in Deutschland. Nach aktuellen Zahlen der Bitkom.ev, aus dem Jahre 2018, nutzten rund 81 % (Haas, 2018, p. 3) der Deutschen ab 14 Jahren regelmäßig ein Smartphone. Dies ist eine Steigerung zum Vorjahr um 3 Prozentpunkte. Besonders die Verfügbarkeit von zahllosen Apps, aus dem sogenannten Playstore für Android und dem App Store für Apple Smartphones, begünstigen diese steigende Nachfrage an mobilen Endgeräten.

Die Benutzerfreundlichkeit von Anwendungen auf den kleinen Touch-Bildschirmen von Smartphones, im Vergleich zur Bildschirmgröße eines Desktop PCs in Kombination mit Maus und Tastatur, wird mithilfe einer App stark verbessert. Darüber hinaus nutzen Apps die internetfähigen Smartphones, um dem Nutzer jederzeit auf Wunsch live beispielsweise über neu eingegangen E-Mails, aktuelle Informationen und Nachrichten, Terminänderungen und Weiteres zu informieren.

Die EAH Jena offeriert, in Kooperation mit der Fachhochschule Erfurt, seit 2014 die App EAH Jena für Android und IOS. Diese App ist besonders für die Nutzung mit dem Smartphone und Tablet optimiert und bietet allen Studenten und Mitarbeitern der EAH Jena ausführliche Informierungen über den Hochschulalltag. Dabei stehen alltägliche Themen im Mittelpunkt, sodass es möglich ist, sich aktuelle Nachrichten aus verschiedenen Fachbereichen anzeigen zu lassen, seinen eigenen Set-Stundenplan auszuwählen und zu betrachten, Informationen zu Telefonnummer, E-Mail-Adresse, Fakultät und Raum von Personen und Professoren der EAH Jena einzusehen, mithilfe einer Mensaübersicht alle Speisekarten der laufenden Woche jeder Mensa in Jena zu erhalten, durch einen Campusplan kann jedes Gebäude der EAH Jena gefunden werden und ein Terminplaner, welcher alle wichtigen Termine des Semesters bereitstellt.

1.1 Problemstellung

Seit der Veröffentlichung dieser neusten EAH Jena App im Jahr 2014, hat sich an der Android App nur wenig getan. Bis zur letzten bekannten Android Version 2.1.1 vom 05.10.2015 wurden mehrere kleine Optimierungen eingebaut und Fehler behoben. Nach diesem Datum wurden keine weiteren Aktualisierungen oder Updates mehr veröffentlicht. Zu einem Heute unbekannten Tag, wurde diese App komplett aus dem Playstore und dem App Store entfernt und war somit vor Projektstart für keinen Studenten oder Mitarbeiter verfügbar.

Zum Ende des Wintersemesters 2018/2019 entschied sich ein Projektteam aus Studenten, Mitarbeitern und ein Professor der EAH Jena diese in die Jahre gekommene Android App zu überarbeiten, um neue Funktionen zu erweitern und das Ziel zu verfolgen, diese App den Interessierten wieder anzubieten und im Playstore zu veröffentlichen.

Eine besondere Anforderung des Projektes ist die Erweiterung der App um einen individuellen Stundenplan je Student.

Ein Student der EAH Jena wird in seinem jeweiligen Fachsemester einem bestimmten Set zugeordnet. Ein Set ist eine Gruppe aus vier bis sechs Studenten, welche gewisse Übungen, Seminare, Praktika und weitere Module zusammen belegen. Durch diese Einteilung in ein Set wird jedem Student ein fester Stundenplan vorgegeben, welcher alle Module des jeweiligen Fachsemesters umfasst. Da dem Studenten der EAH Jena die Möglichkeit eingeräumt wird, Module aus früheren Fachsemstern zu wiederholen, Module aus späteren Fachsemestern oder anderen Studiengängen, auch aus anderen Fachbereichen, zu belegen, reicht oft der feste Set-Stundenplan nicht aus. Hierbei muss der Student mehrere verschiedene Stundenpläne im Auge behalten, um keine Änderung einer Veranstaltung zu verpassen. Diese Problematik soll durch die App gemindert, wenn nicht gelöst werden.

Somit soll es dem Studenten in der App ermöglicht werden einen eigenen Stundenplan zu erstellen, jegliche Module diesem hinzuzufügen und über Änderung an diesen Modulen informiert zu werden.

Diese Anforderung soll im Laufe dieses Bachelorarbeit serverseitig bedacht und umgesetzt werden. Im Rahmen des Bearbeitungszeitraums müssen technische und besonders datenschutzrechtliche Zusammenhänge erkannt und in das Konzept eingearbeitet werden. Somit müssen alle Voraussetzungen des Projektes geschaffen und zum anderen das Konzept des individuellen Stundenplans serverseitig umgesetzt werden.

1.2 Struktur dieser Arbeit

Diese Bachelorarbeit besteht aus neun aufeinander aufbauenden und ausführlichen Kapiteln. Nach der Einleitung und Problemstellung in diesem Kapitel, werden in Kapitel 2 die allgemeinen Rahmenbedingungen dargestellt. Woraufhin in Kapitel 3 eine Analyse des geplanten Projektes durchgeführt wird. In Kapitel 4 wird die Konzeption der Systemarchitektur, des umzusetzenden Service, vorgenommen. Da das Thema Sicherheit besonders in Vordergrund steht, werden im Kapitel 5 verschiedene Angriffsmöglichkeiten und ihre Vereitlung betrachte. Im Anschluss wird in Kapitel 6 die Funktionsweise von Push Benachrichtigungen beschrieben. Im Kapitel 7 wird das heute besonders wichtige Thema Datenschutz analysiert. Woraufhin in Kapitel 8 die eigentliche Umsetzung des geplanten serverseitigen Service erläutert wird. Schlussendlich wird im Kapitel 9 ein Ausblick auf mögliche Weiterentwicklungen gegeben. Des Weiteren wird die Bachelorarbeit und das umgesetzte Projekt reflektiert und ein Fazit über diese Arbeit gezogen.

2 Rahmenbedingungen

2.1 Ausgangssituation

In diesem Abschnitt wird kurz die grundlegende Situation des Stundenplanaufbaus der EAH Jena betrachtet und erläutert. Darüber hinaus werden alle öffentlich zugänglichen Möglichkeiten vorgestellt, um einen Stundenplan abzurufen.

2.1.1 Stundenplanaufbau

Der allgemeine Aufbau der Fachbereiche der EAH Jena ist klar durch Abhängigkeiten nach oben geregelt. Die Abbildung 1 stellt im oben Teil, diese klare Relation dar. Hierbei ist jeder Fachbereich der übergeordneten Organisation EAH Jena zugeordnet. Ein Fachbereich hat dabei ein oder mehrere Studiengänge unter sich, welche dieser verwaltet. Jeder Studiengang hat je nach Erstbeginn und Start im Wintersemester und/oder Sommersemester mehrere aktuell laufende Semester in sich, welche einzeln ihre Fachsemester-Anzahl des jeweiligen Studienganges abbilden. Um Studenten in gewisse Gruppen einzuteilen, werden diese in sogenannte Sets eingeordnet. Diese Sets und ihre Einteilung sollen strikt durch die Studenten angewandt werden, um den reibungslosen Ablauf aller nötigen Veranstaltungen zu gewährleisten.

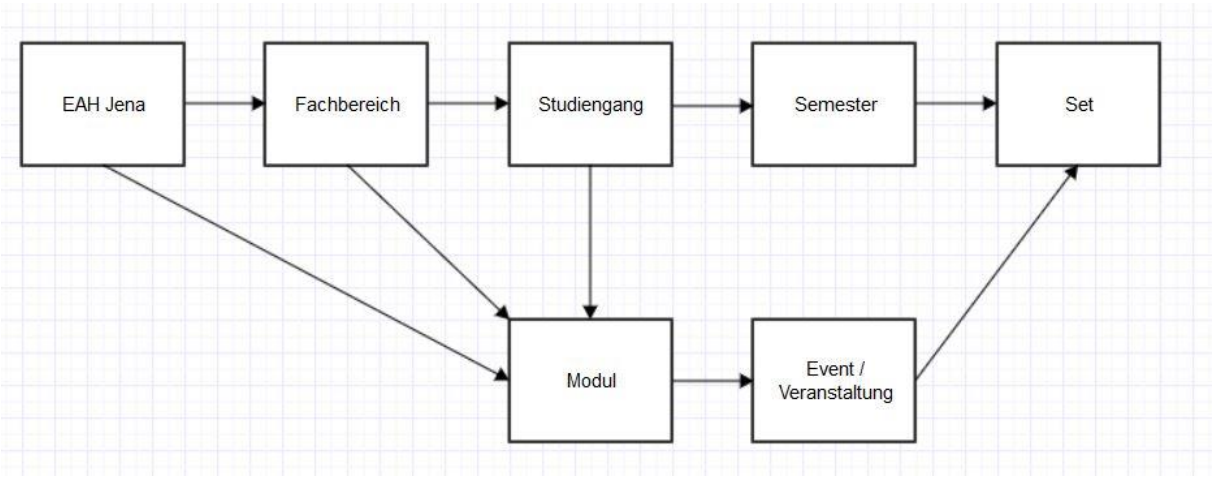

Abbildung 1 Stundenplan Aufbau EAH Jena[1] | Quelle: eigene Abbildung

Im Weiteren wird in der Abbildung 1 der Zusammenhang zwischen der Organisation EAH Jena, den Fachbereichen und den einzelnen Studiengängen in Bezug auf Module dargestellt.

[1] Eigene Darstellung des der verschiedenen Relationen

Dabei kann ein Modul entweder von der Organisation selbst, einem Fachbereich oder einem gewissen Studiengang abhängig und untergeordnet sein. Ein Modul selbst umfasst mehrere einzelne Events, welche sich über das ganze Semester erstrecken können. Diese Events, zusammengefasst in Modulen, werden Sets zugeordnet. Diese zugeordneten Sets können alle aus einem Semester, aber auch aus verschiedenen Semestern, Studiengängen und Fachbereichen sein.

2.1.2 Stundenplan Daten

Den Studenten der EAH Jena werden zwei Möglichkeiten angeboten, ihren eigenen Set-Stundenplan abzurufen.

Möglichkeit eins ist das Nutzen der Webseite der EAH Jena, welche jeden Studenten und darüber hinaus auch Mitarbeiten, es ermöglicht sich nach Auswahl des Studienganges, des Semester und des jeweiligen Sets, sich seinen festgelegten Stundenplan anzuzeigen zu lassen. Dieser Stundenplan wird in Form von mehreren HTML Tabellen ausgegeben und stellt den aktuellen Stundenplan dar. Aufgetreten Änderungen im Verlauf eines Semesters von Modulen und Events, in Form von Terminänderungen, Raumänderungen, Hinzufügungen und Löschungen, werden immer in den aktuellen Stundenplan eingearbeitet. Information und Grund der Änderung werden nur in ausgewiesenen Änderungstabellen aufgezeigt. Diese Änderungshinweis Tabellen, welche auf einer anderen Unterseite der EAH Jena Stundenplan Webseite gefunden werden können, stellen dabei ausschließlich kurzfristige Änderungen dar und informieren den Nutzer nur indirekt über Änderungen des eigenen Stundenplanes.

Die zweite Möglichkeit ist das Nutzen von gewissen Kalenderdateien, welche im Dateityp iCal[2] bereitgestellt werden. Dieses Format beinhalten in standardisierter Form, alle Kalendereinheiten, also Events eines Set-Stundenplanes. Diese Datei kann vom Nutzer heruntergeladen und in ein geeignetes Kalendertool, wie Microsoft Outlook auf dem PC und der vorinstallierten Kalender App des Android Betriebssystem, eingebunden und täglich manuell aktualisiert werden. Bei dieser Methode hat der Nutzer keine Möglichkeit aufgetretene Anpassungen des Stundenplans wahrzunehmen, da keine Änderungshistorie in diesen Kalenderdateien gepflegt wird. Somit sieht der Nutzer seinen zwar aktuell richtigen Stundenplan, er wird aber nicht über durchgeführte Änderungen informiert.

[2] Abkürzung für iCalendar

2.2 Zielsetzung

Im Vorfeld des Bachelor Projektes wurden bereits verschiedene Ziele, welche sich in primäre und sekundäre Ziele einteilen lassen, definiert. Zu den Primärzielen zählen, neben der Konzeption und Entwicklung eines serverseitigen Services, welcher dem Nutzer der neu geplanten EAH Jena App über Änderungen in seinen individuell zusammengestellten Stundenplan durch sog. Push-Benachrichtigungen informiert, die Erstellung und Führung einer Änderungshistorie, über jede Anpassung aller Stundenpläne. Darüber hinaus wird eine datenschutzrechtliche Abschätzung durchgeführt, mit dem Ziel keine personenbezogenen Daten zu erfassen.

Eine Einteilung der sekundären Ziele wurde wie folgt vorgenommen:

- Softwareorientierte Ziele
- Datenschutzorientierte Ziele
- Zukunftsorientierte Ziele

Da die Konzeption eines serverseitigen Services im Vordergrund dieses Bachelor Projekt steht, werden diese softwareorientierte Ziele zuerst benannt. Zu diesen Zielen gehört die Auswahl von standardisierten und allgegenwärtigen Technologien, welche im Verbunden die Architektur und Basis des geplanten serverseitigen Services darstellen. Darüber hinaus soll der Service in seine Gesamtheit keine große Komplexität annehmen, um die Implementierung dessen in die neue EAH App so einfach wie möglich zu gestallten.

Ein weiteres wichtiges Ziel dieser Bachelor Arbeit ist die datenschutzrechtliche Einschätzung dieses Services. Dabei sollen allgemeine und projektbezogenen Szenarien und Themen betrachtet werden. Es soll eine klar begründete Empfehlung in der Konzeption herausgearbeitet werden, welche Daten der Service empfangen und verarbeiten sollte. Ein weiteres sekundäres Ziel ist hierbei die strikte Datenminimierung des Services, während der Kommunikation mit der App, auf das Nötigste.

Um den geplanten serverseitigen Service auch in Zukunft sinnvoll nutzen und weiterentwickeln zu können, müssen in einem Softwareprojekt verschiedene Ziele und Anforderung bedacht und umgesetzt werden. Zu diesen Zielen gehört die Einbindung von aktuellen und nach Einschätzung zukunftsorientierten Technologien, ein gut strukturierter, sauber und aufgeräumter Quelltext, eine ausreichende Dokumentation des Projektes, sowie zweckmäßige Kommentare im Quelltext, um diesen zu erklären. Darüber hinaus soll das Ziel bedacht werden, das andere Hochschulen mit gleichartigen oder vergleichbaren System diesen Service ohne großen Aufwand in ihre Systemlandschaft integrieren können.

Am Ende des Bachelor Projektes sollen alle primären Ziele, also die Umsetzung eines Datenschutzes konformen Service, umgesetzt werden, sowie die Sekundärziele beachtet und weitestgehend realisiert werden.

3 Analyse

In diesem Kapitel wird die Projektanalyse, welche vor der Projektumsetzung stattfand, erläutert. Es wird Stellung bezogen zu nötigen Anforderungen, welche der geplante serverseitige Service abbilden können muss, um den Ablauf eines individuellen Stundenplanes in Kombination mit Push Benachrichtigungen zu verwirklichen. Darüber hinaus wird analysiert, ob und wie die Stundenplandaten der EAH Jena strukturiert und für dieses Projekt nutzbar sind.

3.1 Anforderung

Jedes Programm benötigt eine klare Aufschlüsslung aller Anforderungen, welche der Service im produktiven Alltag abbilden können muss. Anforderungen im Allgemeinen sind Eigenschaften, gewisse Abläufe oder Reaktionen, welche vom System umgesetzt und beherrschbar seinen müssen.

Dabei werden diese in funktionale und nicht funktionale Anforderungen eingeteilt. Funktionale Anforderungen sind Anforderungen, welchen dem direkten Zweck eines Service dienen. Nicht funktionale Anforderungen sind unspezifisch und nur indirekt dem eigentlichen Bestreben des Service zugeordnet. (Johner, 2013) Im Folgenden werden herausgearbeitete Anforderungen benannt und kurz beschrieben.

Funktionale Anforderungen an den Service sind:

- Der Service muss in einem einfachen Serverumfeld genutzt werden können. Dabei ist zu beachten, dass der Service eigenständig in einer verfügbaren Serverlandschaft der FH Erfurt eingebunden werden soll.
- Der Service muss in einer verständlichen und weit verbreiteten Programmiersprache umgesetzt werden. Hierbei ist klar der Bezug zur Weiterentwicklung des Service zu sehen, welche in nachfolgenden Zeit durch Studenten ausgeführt werden soll.
- Der Service muss selbst-ständig die Stundenplan Daten der EAH Jena abrufen, verstehen, auswerten und speichern können.
- Die neu geplante EAH Jena App soll einfach und unkompliziert mit dem serverseitigen Service kommunizieren können. Dies sollte über eine Schnittstelle erfolgen, welche in den Service integriert ist und die App nutzen kann.
- Der Service muss ermöglichen, dass Nutzer der EAH Jena App gewisse Module des Stundenplanes abonnieren können. Alle Informationen über diese Abonnements sollte der Service intern speichern und verwalten.

- Der Service muss beim Finden einer Änderung im Stundenplan, gewisse Nutzer mithilfe von Push Benachrichtigung informieren. Dies sollte auf den Informationen aller gespeicherten Abonnements beruhen.
- Der Service muss datenschutzrechtliche Bestimmungen einhalten.
- Bestimmte Änderungen im Stundenplan müssen an die neue App der EAH Jena übermittelt werden können. Hierbei sollte ein für beide Systeme verständliches Dateiformat der übermittelten Daten eingesetzt werden.
- Der Service muss heutige sicherheitsrelevante Standards nutzen und einsetzen.

Nicht funktionale Anforderungen sind:

- Der Service verschlüsselt alle Daten eines Nutzers.
- Nicht mehr benötigte Daten werden gelöscht, um die Speicherbedarf des Service so gering wie möglich zu halten.
- Anfragen an den Service sollten in anwenderfreundlicher Zeit zurückgegeben werden.

Diese Anforderungen werden in der weiteren Analyse und Konzeptphase, aber auch direkt in der Umsetzung, bedacht und umgesetzt.

3.2 Nutzbarkeit der Stundenplan Daten

Die Nutzbarkeit von Informationen, welche durch automatisierte Prozesse oder einem Service geholt, gelesen und verarbeitet werden sollen, ist unabdingbar. Dabei sollten Daten mehrere Eigenschaften erfüllen, um einfach und unkompliziert von IT-Systemen verstanden zu werden. In diesem Abschnitt wird anhand von mehreren Kriterien analysiert, welcher der beiden Quellen der Beschaffung der Stundenplan Daten der EAH Jena aus dem Abschnitt 2.1.2, für die Umsetzung dieses serverseitigen Service genutzt werden sollte. Die Kriterien sind:

- **Verfügbarkeit**

  Daten, welche von automatisierten Prozessen genutzt und eingesetzt werden, sollten jederzeit dem Dienst zur Verfügung stehen. Eine Authentifizierung ist hierbei nicht ausschlaggebend.

- **Lesbarkeit**

  Die verfügbaren Daten sollten in einem für dem automatisierten Prozess lesbaren Format vorliegen. Besonders die Zeichencodierung spielt in Deutschland, mit seinen Sonderzeichen ä, ö und ü, eine große Rolle. Als Zeichencodierung sollte stets UTF-8[3] verwendet werden.

- **Struktur**

  Die Daten sollten in einer einfach verwertbaren Datenstruktur vorliegen. Hierbei sollten gewisse definierte Standards eingehalten und eingesetzt werden, um den Aufwand des einlesen der Daten in ein internes Objekt der Programmiersprache, zu minimieren.

- **Aktualität**

  Die vorliegenden Daten sollten stets so aktuell wie möglich sein, um die reale Wirklichkeit, aus welcher die Daten entstanden und diese modellieren sollen, sinnvoll darzustellen.

- **Aufbau**

  Der Aufbau sollte im kompletten Datensatz gleich oder gleichartig sein. Darüber hinaus sollten Datenwerte eine gewisse Datenbezeichnung besitzen, welchen den Datenwert inhaltlich einteilt und dessen vorliegenden Datentyp definiert.

[3] Abkürzung für **8**-Bit UCS Transformation Format

- **Inhalt und Zusammenhang**
  Es sollten ausreichend Informationen je Datensatz enthalten sein, um diese in ihrer Komplexität auswerten zu können. Die Daten sollten jeweils einzigartige Identifizierungsnummer aufweisen und Zusammenhänge zu anderen Datensätzen sollten klar über diese Identifizierungsnummern oder anderweitige Werten erkennbar sein.

Die beiden möglichen Quellen, die Nutzung der Stundenplan-Webseite und zum anderen das Heranziehen der iCal Kalenderdateien je Stundenplan-Set, werden im Folgenden anhand der definierten Kriterien betrachtet, um anschließend zu entscheiden, welche Stundenplan Daten Quelle für die Umsetzung des serverseitigen Service genutzt werden sollte.

Bevor alle Set-Stundenpläne bezogen werden können, muss erst in Erfahrung gebracht werden, welche Studiengänge, Semester von Studiengänge und Sets dieser Semester aktuell an der EAH Jena und dessen Stundenplan verfügbar sind. Auf der Stundenplan-Webseite[4] ist dies nur durch großen Aufwand möglich, da eine Komplettübersicht aller Kombination, von Studiengängen, Semestern und Sets, nur durch mehrfaches Parsen von drei voneinander abhängigen Auswahllisten zu realisieren ist. Dieses Verfahren wäre äußerst fehleranfällig, da hierbei die Ergebnisse von ausgeführtem JavaScript abgewartet und ausgewertet werden müssten. Dagegen liegt für die Nutzung der Kalenderdateien, als Quelle der Stundenplan Daten, eine gewisse OPML[5] Datei abrufbereit parat.

```
13      <outline text="Bachelor: Augenoptik">
14        <outline text="AO(BA)1">
15          <outline text="AO(BA)1.01">
16            <outline text="ical_url_floating_times" type="link" url="http://stundenplanung.eah-
17            <outline text="ical_URL_google" type="link" url="http://stundenplanung.eah-
18            <outline text="changes_rss_feed" type="rss" xmlUrl="http://stundenplanung.eah-
19          </outline>
20          <outline text="AO(BA)1.02">
```

Listing 1 - Aufbau der EAH Jena OPML Datei

[4] URL zur Stundenplan Webseite: https://stundenplanung.eah-jena.de/

[5] URL dieser OPML Datei: http://stundenplanung.eah-jena.de/eahapp.opml

Diese OPML Datei wird in einem strukturierten XML[6] Format bereitgestellt und beinhaltet alle Studiengänge, Beispiel Augenoptik in Zeile 13, alle Semester dieses Studienganges, Beispiel AO(BA)1 in Zeile 14, und alle Sets dieses Semesters, Beispiel AO(BA)1.01 in Zeile 15 und AO(BA)1.02 in Zeile 20.

Darüber hinaus wird je Set der Link zur jeweiligen Stundenplan-Kalenderdatei übergeben, Beispiel Zeile 16 im Wert URL.

Das Nutzen der OPML Datei sollte strikt bevorzugt werden, da das Erhalten der Studiengang-, Semester- und SetStruktur der EAH Jena, mithilfe einer strukturierten Datei, die Fehleranfälligkeit des Systems stark verringert.

Beide Stundenplan Daten Quellen sind jederzeit verfügbar und abrufbereit. Beide Arten der Datenbesorgung erfordern keine Authentifizierung und sind ohne Aufwand frei über das Internet abrufbar. Somit kann der geplante Service ohne Einschränkung auf jedem beliebigen System mit Internetzugang eingerichtet und produktiv eingesetzt werden.

Die Lesbarkeit beider Stundenplan Daten Quellen unterscheiden sich nicht voneinander. Beide werden in der Zeichencodierung UTF-8 übermittelt und es treten keine Fehler bei den deutschen und internationalen Sonderzeichen auf.

Dagegen zeigten sich beim Analysieren der jeweiligen Struktur beider Datenquellen erhebliche Unterschiede. Die Stundenplan-Webseite der EAH Jena gibt die Daten in Form von HTML Tabellen aus. Diese HTML Tabellen sind stark auf die Bedürfnisse von menschlichen Nutzern ausgelegt und sind nur mit sehr großem Aufwand für ein Service nutzbar. Die gewünschten Informationen liegen verschachtelt in HTML Code, da diese HTML Webseite nicht dafür bestimmt ist zur Datenerhebung, in diesem Sinne, genutzt zu werden. Dagegen liegen die Stundenplan-Kalenderdateien in einem definierten und strukturierten Format namens iCal vor. Dieses Dateiformat wurde entwickelt, um Kalenderevents, wie eine Veranstaltung eines Studenten, zu strukturieren, abzuspeichern und anderen Nutzern oder Systemen zur Verfügung zu stellen. Dabei unterstützen die bekanntesten E-Mail- und Kalenderprogramme, wie Microsoft Outlook und der Google Kalender, dieses Format (Sangmeister, 2016). Dies ergibt, dass ein geplanter serverseitiger Service, in Bezug auf die Dateistruktur die Kalenderdateien, den HTML Dateien der Webseite, vorziehen sollte.

[6] Abkürzung für Extensible Markup Language

Die Aktualität der Stundenplan-Daten der Webseite ist stets hoch und Änderungen, welche in das Stundenplan-System der EAH Jena eingearbeitet worden sind, stehen sofort zur Verfügung. Hingegen werden die Stundenplan-Kalenderdateien nur einmal täglich aktualisiert.[7] Bei Betrachtung diesem Kriterium hat somit die Stundenplan Webseit einen klaren Vorteil, gegenüber den bereitgestellten Kalenderdateien.

Der Aufbau beider Daten der Quellen ist grundlegend verschieden. Die HTML geführte Form der Stundenplan-Webseite zielt auf die einfache Lesbarkeit eines menschlichen Nutzers ab und definiert somit keine Datenwert nach ihren Datentyp, der Art der Daten oder ihren Zweck. Besonders die Nutzung von eigenen Wochenzahlen, welche mit Semesterstart bei eins beginnen und keine direkte Referenz zu ihrem Startdatum aufweisen, verkompliziert das Nutzen dieser Datenquelle. Darüber hinaus werden inhaltlich gleichartige Wochen zusammengefasst, was die eindeutige Identifizierung von einzelnen Event deutlich erschwert. Im Gegensatz dazu, nutzten die Kalenderdateien einen definierten Aufbau, welche durch das genutzte Format vorgegeben wird. Dabei teilt sich eine Kalenderdatei der EAH Jena in zwei Abschnitte ein. Der erste Abschnitt der Datei beinhaltet informelle Informationen, wie den Herausgeber dieser Datei, aber auch datenrelevante Informationen, z. B. die genutzten Zeitzone der dargestellten zeitlichen Daten. Darüber hinaus ist jedes Event in seiner kompletten Form einzeln abgebildet und jeder Datenwert hat eine verständliche Datenbezeichnung.

```
36 BEGIN:VEVENT
37 ORGANIZER:MAILTO:stundenplanung@eah-jena.de
38 DTSTART:20190520T080000
39 DTEND:20190520T110000
40 UID:SPLUS183D37s-1
41 DTSTAMP:20190528T061700Z
42 SUMMARY:WIEC(BA)Web Shop Proj./P/01.1
43 LOCATION:03.00.18
44 DESCRIPTION:Dozent: Werner\, A.\n
45  Raum: 03.00.18\n
46  Veranstaltung: WIEC(BA)Web Shop Proj./P/01.1
47 CLASS:PUBLIC
48 END:VEVENT
49 BEGIN:VEVENT
```

Listing 2 - Aufbau einer Kalenderdatei

[7] Stand August 2019

Listing 2 zeigt den Aufbau eines einzelnen Events des Stundenplanes in einer verfügbaren Kalenderdatei. Hierbei bilden die Zeilen 36 bis 48 ein komplettes Event ab. Alle weiteren Events sind mit gleichem Aufbau untereinander angeordnet. Darüber hinaus hat jeder Wert eine fest zugewiesene Datenbestimmung. Somit ist klar erkenntlich, dass der Wert 05.03.28 in Zeile 43 mit der Bezeichnung LOCATION den Raum dieses Events abbilden soll. Beim Betrachten des Themas Aufbau sollten die Kalenderdateien bevorzugt werden.

Ein gewisser Zusammenhang zwischen einzelnen Events ist bei der Nutzung der Stundenplan-Daten der Webseite nur unzureichend gegeben, da der Zusammenhang nur über das Parsen und Auswerten des Event-Titels erfolgen kann.

Auch eine direkte Identifizierung eines Events ist nicht möglich, da keine eindeutigen Identifizierungsnummern gesetzt oder übergeben werden. Der Datenumfang ist im eigentlichen Sinne ausreichend, da die nötigsten Informationen, wie die Start und Endzeit, der Raum, der Titel und die Lehrkraft, angegeben werden. Dagegen besitzt jedes Event in den Kalenderdateien eine eindeutige Identifizierungsnummern. Diese ist in Listing 2 in Zeile 40 UID ersichtlich. Darüber hinaus lässt sich anhand dieser Identifizierungsnummer ein Zusammenhang zu allen weiteren Events eines Modules herstellen, da alle zusammengehörigen Events die gleichen Identifizierungsnummern vor dem Bindestrich haben. Inhaltlich sind hierbei alle nötigen Daten vorhanden.

Zusammenfassend kann ausgesagt werden, dass für den geplanten serverseitigen Service besonders auf Basis der Kriterien Verfügbarkeit, Lesbarkeit, Struktur, Aufbau, Zusammenhang und Inhalt eindeutig die verfügbaren Kalenderdateien genutzt werden sollten. Einzig die nur ungenügende Aktualität der Daten sollte jederzeit bedacht werden. Ein übergeordnetes Ziel sollte es sein, die Aktualisierungsfrequenz dieser Dateien erhöhen zu lassen.

4 Konzeption der Systemarchitektur

In diesem Kapitel sind die zum späteren Verständnis dieser Arbeit benötigten Technologien und Begriffe erläutert. Es werden die grundlegenden Technologien, die Arbeitsweisen und gewisse Funktionsweisen, welche beim Umsetzen des geplanten serverseitigen Service genutzt werden sollten, vorgestellt. Dabei werden einerseits architekturbezogene und technische Aspekte als auch sicherheitsrelevante Gesichtspunkte untersucht und Entscheidungen getroffen wie diese Thematiken während der Umsetzung bedacht und eingesetzt werden müssen.

4.1 LAMP-Architektur

Zur Umsetzung eines serverseitigen Service kann auf verschiedene frei verfügbare Programmiersprachen und Systeme zurückgegriffen werden. Im Rahmen dieses Projektes, besonders mit Bezug darauf, dass Studenten in Zukunft dieses Projekt weiterentwickeln können sollen, ist eine sogenannte LAMP-Paket[8] mit der Programmiersprache PHP zu bevorzugen. Im Folgenden wird dieses LAMP-Paket kurz erläutert.

Ein LAMP-Paket ist eine Zusammenstellung aus mehreren Einzelkomponenten und erleichtert die Installation und Ersteinrichtung eines Servers, da die benötigten Einzelkomponenten nicht einzelnen installiert werden müssen.

Dabei besteht diese Gruppierung ausschließlich auch freier Software und enthält folgende Komponenten: (Luber, 2019)

- Vorkonfiguriertes Linux als Betriebssystem
- Apache als Webserver
- MariaDB oder MySQL als Datenbankserver
- PHP als Programmiersprache

Die folgende Abbildung stellt den Zusammenhang der einzelnen Komponenten dar:

[8] Das Akronym LAMP steht für **L**inux, **A**pache, **M**ariaDB, **P**HP

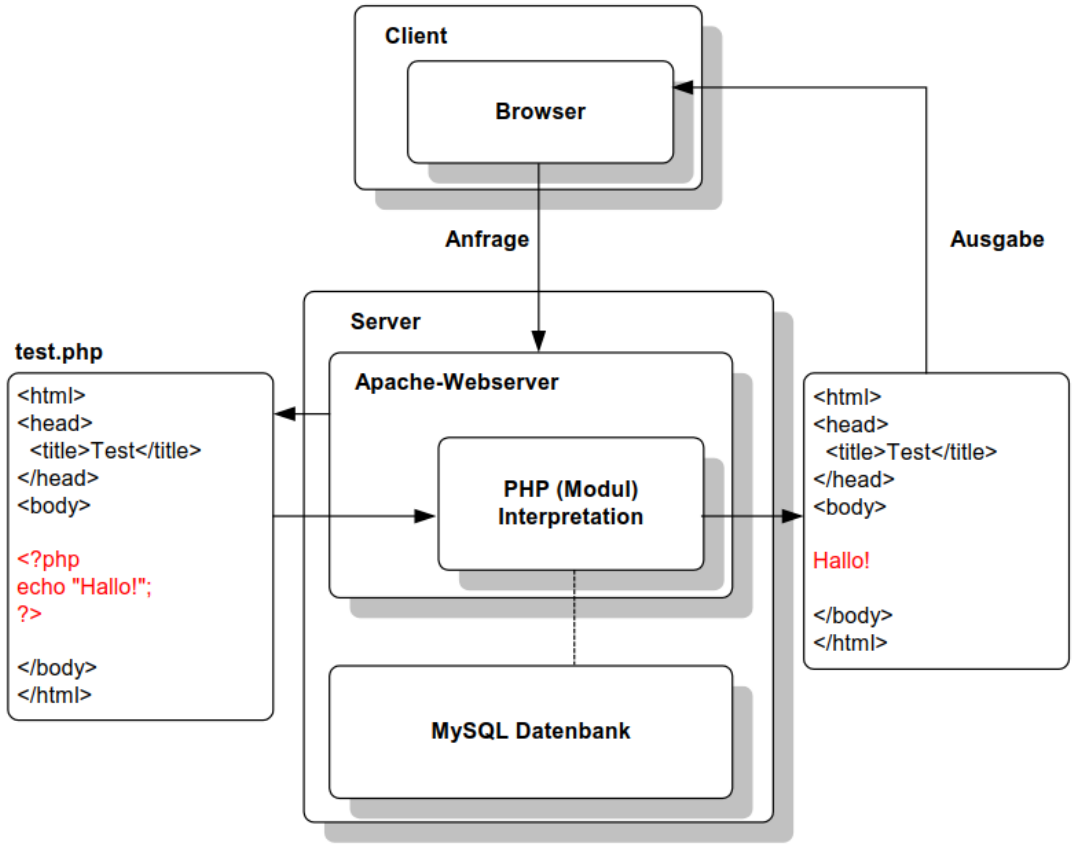

Abbildung 2: Zusammenspiel der Serverkomponenten in einer LAMP-Architektur

Der Server, welcher als Betriebssystem Linux einsetzt, besteht aus den beiden Komponenten Apache-Webserver und einem Datenbanksystem. In dem Webserver wird die Programmiersprache PHP als Modul implementiert, welches nur vom Webserver genutzt werden kann. Darüber hinaus sollte auch ein PHP in einer CLI Version installiert werden, da diese Kommandozeilen Version das Ausführen von PHP-Skripten vom System selbst ermöglicht. Die in der Abbildung als MySQL Datenbank dargestellte Komponente, ist ein komplettes Datenbank-Management-System, welche die verschiedenen Datenbanken in sich verwaltet. Des Weiteren ermöglicht der Datenbank-Server mithilfe der Programmiersprache SQL den Zugriff auf die einzelnen Datenbanken. Als Datenbank-Management-System kann und sollte MariaDB-DBMS[9] verwendet werden.

Die serverseitige und skriptbasierte Programmiersprache PHP kann entweder in HTML eingebettet oder eigenen ständig ausgeführt zu werden, um datenbankgestützte Webseiten oder serverinterne Prozesse zu realisieren. Serverseitig bedeutet in diesem Zusammenhang, dass der PHP Quellcode vom Webserver oder dem Server selbst verarbeitet wird.

[9] DBMS entspricht der Abkürzung für Datenbank-Management-System

Ein PHP Skript wird gestartet, wenn ein clientseitiger oder serverseitiger Anwender eine Anfrage an den Server oder das System stellt. Die Clientseite kann dabei der Server selbst, ein externes IT-System wie eine App oder ein menschlicher Nutzer, mithilfe eines Browsers, sein. Wenn der PHP-Programmcode in HTML eingebettet ist, erfolgt der Aufruf über eine gültige .html URL. Wenn der PHP-Programmcode nicht in HTML Code eingebettet ist, erfolgt der Aufruf über eine .php URL. Darüber hinaus ist es möglich, diesen PHP-Programmcode serverseitig über die durch das Betriebssystem bereitgestellte Konsole direkt auszuführen. Der PHP-Interpreter interpretiert die einzelnen Befehle des Programmcodes und führt diesen aus. Das Resultat, oder die gewünschte Ausgabe, der Ausführung findet seinen Platz entweder als HTML-Ausgabe anstelle des Quellcodes in der HTML Datei, oder als einzige Ausgabe, wenn eine PHP Datei ohne HTML Bezug, angesprochen worden ist und wird an den anfragenden Client gesendet. Es wird niemals der PHP-Programmcode, welche ordnungsmäßig implementiert worden ist, als Ausgabe gesendet, sondern immer nur dessen Ergebnis.

Um die in Programmablauf anfallenden Daten zu speichern, sollte das in dem LAMP-Paket inkludierte Datenbanksystem MariaDB genutzt werden, welches in der Abbildung 2 als MySQl Datenbank dargestellt ist. Dieses quelloffene System erlaubt es Datenbanken zu erstellen und stellt die Sprache SQL bereit, um mit dem Datenbank-Server zu kommunizieren.

4.2 Docker

Um die Anforderung zu erfüllen, den umgesetzten Service auf einem System der FH Erfurt laufen zu lassen, muss der Service in einem sogenannten Docker Container umgesetzt werden. In Rahmen dieser Arbeit wird das System Docker, mit allen seinen umfangreichen Eigenschaften, im Folgenden kurz beschrieben.

Docker ist eine Open-Source Software zur Virtualisierung von Betriebssystemen, um darin Anwendungen auszuführen. Docker arbeitet mit sogenannten Containern, in denen die Systeme laufen, um sie voneinander zu trennen. Diese Container enthalten alle notwendigen Abhängigkeiten, um die vorgesehene Anwendung darin auszuführen. (Docker Inc, 2019)

In der Abbildung 3 sind die wichtigsten Komponenten einer Docker-Installation dargestellt: (Docker Inc, 2019)

Docker Daemon – Erstellung, Ausführung und Überwachung der Container

Docker Client – Kommunikation via HTTP Rest

Docker Registry – Ablage und Verteilung der Images

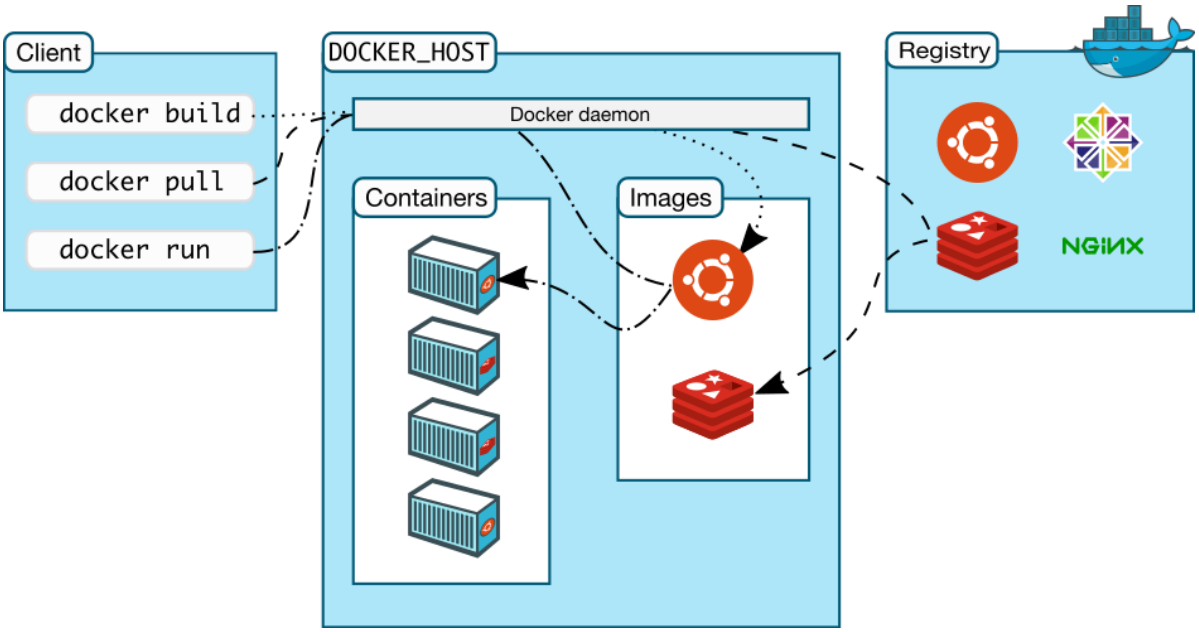

Abbildung 3: Architektur von Docker | Quelle: (Docker Inc, 2019)

Ein Container beinhaltet ein Image. Dieses Image wird mit Hilfe einer Docker-Datei gebildet. In jeder Zeile dieser Datei steht genau ein Befehl mit einem Parameter. In Listing 3 ist der Inhalt eine Docker-Datei beispielhaft dargestellt:

```
1  RUN apt-get update && apt-get -y install cron
2  ADD crontab /etc/cron.d/hello-cron
3  RUN chmod 0644 /etc/cron.d/hello-cron
4  RUN crontab /etc/cron.d/hello-cron
5  ADD crontab /etc/cron.d/task-cron
6  RUN chmod 0644 /etc/cron.d/task-cron
7  RUN touch /var/log/cron.log
8  COPY crontab /etc/crontab
9  RUN touch /var/log/cron.log
10 RUN (crontab -l ; echo "* * * * * echo "Hello world" >> /var/log/cron.log") | crontab
11 CMD ["/run.sh"]
```

Listing 3: Dockerfile zum Bilden eines Images

In diesem Beispiel-Code wird das Programm Cron installiert und mithilfe verschiedener Befehle konfiguriert.

4.3 Technische Aspekte

In diesem Kapitel werden die verschiedenen technischen Komponenten vorgestellt und kurz erläutert.

Da der geplante Service sicher mit einer Smartphone-App kommunizieren soll, muss hierfür ein sicheres Protokoll zur Datenübertragung genutzt werden. Hierfür werden die Protokolle HTTP und HTTPS betrachtet, um eine Aussage darüber zu treffen, welches der beiden Übertragungsprotokolle grundsätzlich verwendet werden sollte.

4.3.1 HTTP

Das Hypertext Transfer Protocol ist ein von der IETF[10] im RFC 2616 (Fielding, et al., 1999) spezifiziertes Protokoll. HTTP ist ein zustandsloses Client-Server-Protokoll welches ermöglicht, z. B. bei Belieben ein Link im Browser zu öffnen, um eine Verbindung zu einem Server herzustellen. Der Austausch einer sog. Request/Response-Session ist unabhängig von der gesamten vorgehenden und folgenden HTTP-Kommunikation und läuft stets unverschlüsselt ab. Dabei beschränkt sich dieses Protokoll nicht nur auf das Übertragen von einfachen Webseiten, sondern erlaubt den Austausch beliebiger Daten.

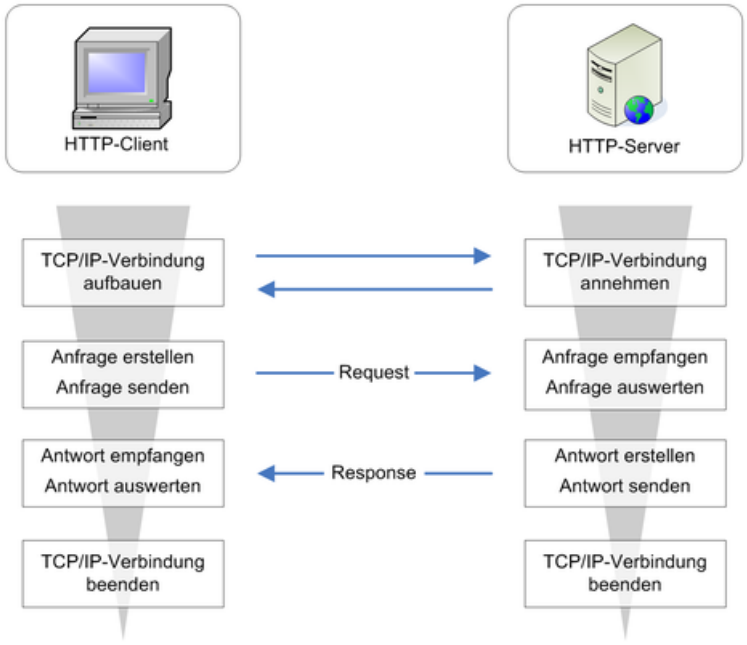

Abbildung 4: Beispiel einer Request/Response-Session |Quelle: ((vectorsoft), 2012)

[10] IETF Abkürzung für Internet Engineering Task Force

Die Funktionsweise einer HTTP-Anfrage ist in Abbildung 4 dargestellt und wird am Beispiel erläutert, dass ein Nutzer eine HTML Seite mithilfe des Browsers aufruft:

Im ersten Schritt wird eine TCP-Verbindung auf seitens des Clients aufgebaut. Dies geschieht, indem der Client eine URL[11] mithilfe eines Browsers aufruft.

Der Browser fragt nun einen DNS-Server an, wie die IP-Adresse des aufrufenden Servers ist. Der DNS-Server ist ein Dienst, welcher Namensauflösungen durchführt und ist vergleichbar mit einem großen Verzeichnis. Der DNS-Server nimmt Anfragen für URLs entgegen und antwortet mit der jeweiligen IP des Servers, da über die URL direkt keine Verbindung zum Server aufgebaut werden kann. Anschließend wird eine TCP-Verbindung zum Server aufgebaut. Standardmäßig wird für eine HTTP-Verbindung der Port 80 genutzt.

Im zweiten Schritt fordert der Browser des Clients die gewünschte Ressource vom Server an. Dies kann eine HTML-Webseite oder eine beliebige andere Datei sein. Wenn eine Ressource angefordert wird, wird dies mithilfe der HTTP Methode GET durchgeführt.

HTTP definiert acht Methoden. Die wichtigsten vier davon sind: (Fielding, et al., 1999)

Methode	Beschreibung
GET	Fordert die Repräsentation einer Ressource an
PUT	Anlage einer neuen Ressource bzw. Ersetzung einer bestehenden
POST	Übermittlung von Daten
DELETE	Löschen einer Ressource

Tabelle 1: Übersicht http-Methoden

Im Schritt drei, versucht der Server die angeforderte Ressource dem Client bereitzustellen. Bevor der Server die Datei „losschickt“, sendet er einen Statuscode, um den Client das Antwortergebnis vorab mitzuteilen, und überträgt dann die angeforderte Ressource. Für HTTP wurden viele Statuscodes definiert. Diese gruppieren sich in die Gruppen: 1xx für allgemeine Informationen, 2xx für erfolgreiche Anfragen, 3xx für Umleitungen, 4xx für clientseitige Fehler, 5xx für serverseitige Fehler.

[11] Abkürzung für Uniform Resource Locator

Die wichtigsten Statuscodes im Einzelnen sind:

Statuscode	Beschreibung
200	Die Anfrage kann erfolgreich beantwortet werden.
400	Die Anfrage war fehlerhaft.
404	Die geforderte Ressource kann vom Server nicht gefunden werden.
503	Der angefragte Server steht nicht zur Verfügung

Tabelle 2 HTTP Statuscodes

Im vierten Schritt versuchen der Server und der Client die Verbindung voneinander zu trenne. Dies geschieht, wenn alle angeforderten Daten übertragen worden sind, welche für die angeforderte HTML Webseite benötigt werden.

Besonders bei Nutzer- und Formulareingaben sollte eine einfache HTTP Verbindung ohne Verschlüsslung, nicht mehr verwendet werden. Da die Daten bei Nutzung von HTTP im sog. Klartext übertragen werden. Dies ist besonders risikobehaftet, wenn Nutzer sensible Informationen, wie Passwörter und Bankdaten, an den Server übermitteln möchte.

4.3.2 HTTPS

HTTPS[12] steht für HyperText Transfer Protocol Secure und ist eine Erweiterung zu HTTP. Mithilfe von HTTPS werden Daten im World Wide Web verschlüsselt übertragen. Somit wird stets eine verschlüsselte und abhörsichere Verbindung zwischen Client und dem Server hergestellt. Dies ist besonders erforderlich, um sensible Daten des Nutzers zu übertragen. Ob eine aktuelle Verbindung zu einem Server via HTTPS verschlüsselt ist, erkennt man am grünen Schloss neben der URL im Browser, welches in Abbildung 5 dargestellt ist.

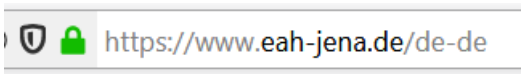

Abbildung 5 Sichere HTTPS Verbindung im Browser | Quelle: eigene Abbildung

[12] RFC 2818 HTTPS (Hypertext Transfer Protocol Secure)

Als Betreiber einer Webseite oder Service muss sichergestellt werden, dass der eigene Dienst HTTPS als Standardprotokoll verwendet.

Als Verschlüsslungsart wird TLS[13] eingesetzt. Syntaktisch ist HTTPS identisch mit HTTP[14], jedoch wird HTTP, um die Komponente einer zusätzlichen Verschlüsselung via SSL/TLS, erweitert.

Abbildung 6: HTTP vs. HTTPS | Quelle (Cloudflare, 2019)

Im Vergleich zur bereits erwähnten HTTP Verbindung, findet bei HTTPS als Erstes eine geschützte Identifikation und Authentifizierung statt. Anschließend wird mithilfe asymmetrischer Verschlüsselungsverfahren ein Schlüssel ausgetauscht und verwendet(Cloudflare, 2019).

Grundsätzlich kann ausgesagt werden, dass der geplante serverseitige Service, das Übertragungsprotokoll HTTPS als definierten Standard verwenden sollte, um besonders die Daten des Nutzers der App nur verschlüsselt über das Internet zu versenden.

Im Weiteren wird der netzwerkbasierte Architekturstil Rest beschrieben, welcher es erlaubt eine Art Schnittstelle zu erschaffen, welche externe Systeme, wie die geplante EAH Jena App, nutzen können, um mit dem serverseitigen Service in klar definierter Form zu kommunizieren.

[13] Abkürzung für: Transportschichtsicherheit

[14] Siehe Abschnitt 4.3.1

4.3.3 Rest

REST[15] ist ein Architekturansatz, welcher im Jahr 2000 von Roy Thomas Fielding ausgearbeitet worden ist (Fielding, 2000) und beschreibt wie Systeme miteinander über das Internet oder einem anderen beliebigen Netzwerk kommunizieren könne.

REST dient dabei der Schaffung einer einheitlichen Schnittstelle, welche die Kommunikation zwischen mehreren maschinellen Systemen vereinfachen soll. Hierbei stellt der Server eine Dienst oder Service bereit, welche jeder autorisierte Client abfragen kann.

REST beruht auf sechs unterschiedlichen Prinzipien, welche das zentrale Unterscheidungsmerkmal zu anderen netzwerkbasierten Architekturstilen sind (Wolf, 2015):

1) Eindeutig identifizierbare Ressourcen
   Eine Ressource ist eine Information, welche abgefragt werden kann und mithilfe eindeutigen IDs identifizierbar ist.
2) Verknüpfung / Hypermedia
   Der Client fragt ausschließlich über festgelegt URLs den Server an. Die Antworten des Servers sollten immer in einem Standardformat, wie JSON, erfolgen.
3) Standardmethoden
   Es sollen ausschließlich HTTP Standardmethoden zur Kommunikation verwendet werden.
4) Unterschiedliche Repräsentationen
   Die Client benötigt keine zusätzlichen Informationen des Servers, um mit diesem zu kommunizieren.
5) Statuslose Kommunikation
   Der Client und der Server müssen sich niemals wiedererkenne. Jede Abfrage ist in sich geschlossen und baut nicht auf eine vorherige Abfrage auf.
6) Caching
   Caching, das zwischenspeichern von Abfragen, kann oder sollte, wenn möglich, genutzt werden.

[15] REST ist die Abkürzung für REpresentational State Transfer

Auf Basis von HTTP sind alle Prinzipien direkt umsetzbar, da sie im bereits im fest definierten Transferprotokoll HTTP enthalten sind. In der Umsetzung, des geplanten serverseitigen Service, sollte eine Rest Schnittstelle geschaffen werden, welche die App der EAH Jena nutzen kann, um mit dem Service zu kommunizieren.

Das Antwort-Dateiformat der Rest Schnittstelle sollte JSON sein, welche im Folgenden beschrieben wird.

4.3.4 JSON

JSON ist ein Format zum Strukturieren von Daten, welches sowohl für den Menschen, als auch für einen Computer lesbar ist. Ecma International beschreibt JSON im Standard als schlankes, sprach-unabhängiges Datenaustauschformat (ECMA, 2013). Im Vergleich zum Datenformat XML, welches ebenso sehr häufig als Format für strukturierte Daten genutzt wird, besitzt JSON eine einfachere Syntax, welche auch von menschlichen Anwendern sehr einfach zu lesen ist. JSON baut in seiner Syntax auf zwei Strukturen auf:

- Schlüssel / Wert - Einem Schlüssel wird einem Wert zugeordnet. Der Schlüssel darf in einer Liste dabei nur einmal auftreten.
- Liste – Eine Liste ist eine Aneinanderreihung von Element. Die Struktur wird in vielen Programmiersprachen Array oder List genannt.

Darüber hinaus begünstig diese Syntax die Geschwindigkeit der Verarbeitung der strukturierten Daten in verschiedenen Programmiersprachen. Das Format JSON ist rein textbasiert und dabei unabhängig von allen Programmiersprache.

Somit sollten heutzutage alle strukturierten Daten, welche ein Service erhält und selbst absendet, das Format JSON nutzen, um den Datenaustausch aller Prozesse im Internet zu einfachen.

Der folgende Code stellt ein Beispiel für eine JSON-Datei dar:

```
1 {"name" : "Meier",  // Schlüssel , Wert
   "alter" : 42,      // Schlüssel , Wert
2 "freunde" : [       // Liste start
  { "name" : "Moritz", "text" : "Guten Morgen!" }, // Listeninhalt
  { "name" : "Kevin", "text" : "Guten Abend" }     // aus Schlüssel, Wert
  ],}                 // Liste end
```

Listing 4: Beispiel JSON-Datei

Hierbei werden die Datenwerte Maier und 42 mit jeweilig passenden Schlüsseln name und alter benannt, um einfach und klar identifiziert zu werden. Neben der einfachen Auflistung von Schlüsseln und Werten ist auch die Verschachtelung von Datensätzen möglich.

Eine solche Verschaltung wird in Zeile zwei genutzt und angewandt.

Mit dieser Erkenntnis sollte das Dateiformat JSON, bei jeder Ausgabe des serverseitigen Service, verwendet werden.

5 Analyse von Angriffsmöglichkeiten

Einem Angreifer stehen verschiedene Methoden für einen Angriff auf den anvisierten Service zur Verfügung. In dem folgenden Unterkapitel werden verschiedene Angriff Methoden betrachtet, welche bei der Umsetzung des Service bedacht und wenn nötig und möglich verhindert werden sollten.

5.1 Distributed Denial-of-Service Attacke

Ein besonders häufig auftretender Angriff, auf ein im Internet verfügbarer Server, ist eine Distributed Denial-of-Service Attacke – kurz DDoS-Angriff. Dieser Angriff zielt darauf ab den Webserver abstützen zu lassen, indem die Infrastruktur des Systeme selbst überlastet wird. Ein solcher Angriff ist mithilfe zweier unterschiedlicher Verfahren möglich.

In sehr häufigen Fällen werden gewisse Sicherheitslücken von eingesetzter Software oder Programmierfehler des Entwicklers gefunden und für einen DDoS-Angriff ausgenutzt. Beim ausnutzen solcher Lücken oder Fehler kann das absenden von wenigen Anfragen den Server zu einer Überlastung der System- Infrastruktur führen. (Terrorismusbekämpfung, 2017, p. 10)

Diesen Fall des Angriffs lässt sich leicht entgegenwirken. Die oft ausgenutzten Sicherheitslücken werden häufig erst nach ihrer öffentlichen Bekanntgabe von Kriminellen ausgenutzt und befinden sich meist nur in sehr alten Versionen der Software. Falls ein Fehler in einer aktuellen Version gefunden wird, wird dieser meist sehr schnell durch die jeweiligen Entwicklerteams repariert. Hierfür muss die eingesetzte Software aber stets in ihrer aktuellsten Version genutzt werden.

Eine weitere Möglichkeit eines DDoS-Angriffs beruht auf der reinen Masse der Anfragerechner. Dafür werden durch Viren befallen Rechner genutzt, um ein sogenanntes Bot-Netz zu erschaffen. Dieses Bot-Netz kann von einem sog. Bot-Master genutzt werden, um per Befehl einen Angriff auf einen bestimmten Server zu starten. Dabei rufen alle im Bot-Netz befindlichen Rechner gleichzeitig eine URL im Browser auf oder stellen eine gewisse Anfrage an den Ziel-Server. Die bereits erwähnten Sicherheitslücken spielen bei dieser Art des DDoS-Angriffs keine Rolle, da die reine Höhe der Anzahl der Anfragen eine Überlast des Servers hervorruft. Jede angenommene Anfrage benötigt eine gewisse Rechenleistung und Speicherbedarf, um vom Server abgearbeitet zu werden. Ist die beim Angriff auftreten Last zu hoch, kann keine weitere Anfrage entgegnen genommen und verarbeitet werden. (Terrorismusbekämpfung, 2017, p. 8)

Variante eins eines DDoS-Angriffs sollte bereits in der Umsetzung des geplanten Service bedacht werden, indem nur stets die aktuellste Version jeder Software genutzt wird.

Variante zwei dagegen kann nicht oder kaum in der Umsetzungsphase bedacht werden, da dieser Angriff sehr schwerwiegend ist und darauf abzielt den bereitgestellten Service und dessen Funktionen in ihrer normalen Form zu nutzten. Um einen reibungslosen Ablauf des Service zu ermöglichen, sollte der produktiv eingesetzte Server leistungsmäßig stark genug sein, um alle tagtäglich auftretenden Anfragen von realen Nutzern zu gewährleisten.

5.2 SQL-Injection

Im Folgenden werden spezielle Angriffe auf ein Datenbank-System beschrieben und erläutert. Abschließend werden verschiedene Empfehlungen abgegeben, um diese Angriffe, beim geplanten Service abzuwehren oder zu unterbinden.

Nicht nur in Programmiersprachen, sondern auch im Datenbank-Server selbst können architekturbedingte Risiken und Probleme auftreten. Auch wenn die aktuellen DBMS kaum Schwachstellen aufweisen, ist es trotzdem möglich, dass Kriminelle die Datenbank und dessen Anfragen für Angriffe ausnutzen. Oftmals entstehen die größten Probleme durch falsch konfigurierte Datenbank-Server, mit der Folge, dass Angreifer eine Verbindung zum Datenbank-Server aufbauen und in diesen eindringen können. (Sharma, 2005, p. 3) Besonders häufig wird bei Installation eines Datenbank-Servers der Benutzer root ohne Passwort erstellt. Wird dieser Benutzer nicht geändert oder gelöscht kann ein Angreifer direkt oder über die Webanwendung des Servers Zugriff auf das System erhalten, soweit dies nicht durch letzter Instanz durch das Betriebssystem unterbunden wird. Darüber hinaus haben Datenbank-Accounts oft zu viele Rechte, welche die jeweiligen nutzenden Personen oder Systeme gar nicht alle benötigen. Der größte Fehler ist es hierbei jedem Nutzer der Datenbank alle Rechte vollumfänglich einzuräumen. So kann mittels SQL-Injection die Datenbank unautorisiert abgefragt, Daten manipuliert oder die Struktur der einzelnen Tabellen geändert werden.

Eine SQL-Injection ist eine Schadcode-Einspeisung, wobei ein Angreifer versucht manipulierte SQL Befehle von der Anwendung oder dem Datenbank-Server selbst ausführen zu lassen (Sharma, 2005, p. 3). Dieser Angriffstyp gehört zu den häufigsten und gefährlichsten Angriffen auf Server, welcher Daten – oder schlimmer – personenbezogene Daten verarbeiten. Solchen Anwendungen nutzen meist Benutzereingaben für Werte der Datenbank-Abfragen. Übergibt ein Angreifer anstelle von erwartenden Werten manipulierte Datenbank-Statements, handelt es sich dabei um eine versuchte SQL-Injection.

Im Folgenden werden die Grundlagen beschrieben, um zu verstehen, wie ein Angreifer eigene Datenbank-Statements in einen serverseitigen Datenbank-Aufruf einschleusen kann.

Einzelne Datenbank-Felder werden verschiedene Datentypen zugeordnet. Die Typen String (z.B. varchar), Zahl (z.B. int) und Datum (z.B. datetime) sind für SQL-Injections am bedeutsamsten. (Esser, 2008, p. 123)

Wird ein String oder Date als Wert an eine Datenbank übermittelt, muss dieser mit zwei Anführungszeichen " ("{WERT}") oder Hochkommas ' ('{WERT}') umschlossen sein. Dagegen kann eine Zahl ohne Umschließung in einem SQL Statement verwendet werden.

```
1 SELECT name FROM studenten WHERE studiengang = "E-Commerce";
```

Listing 5: Beispiel einer SELECT-Abfrage

Diese SQL Anfrage gibt alle Namen der Studenten aus, welche den Studiengang „E-Commerce" studieren. Hierbei muss der String E-Commerce durch zwei Hochkommas oder Anführungszeichen im Statement angegeben werden. Es hat eine Datenbank Fehlerantwort zur Folge, wenn dies nicht bedacht wird.

```
1 SELECT name FROM studenten WHERE matrikelnummer = 638475;
```

Listing 6: angepasste SELECT-Abfrage

Diese SQL Anfrage gibt den Namen des Studenten aus, welcher die Matrikelnummer 638475 hat. Hierbei ist es unwichtig, ob die Zahl mit Hochkomma oder Anführungszeichen umschlossen ist.

```
1 SELECT name FROM studenten WHERE matrikelnummer = $_GET['mat_nummer'];
```

Listing 7: Übergabe einer Variable einer SELECT-Abfrage

Im Beispiel dieser Anfrage wird die Variable mat_nummer, welche mithilfe eines HTTP GET Parameter übergeben worden ist, ungeprüft in das SQL-Statement eingebunden.

Findet ein Angreifer solch eine Schwachstelle der Anwendung, kann er diesen nutzen, um eigene schädliche Befehle einzuschleusen. Finden steht hierbei für raten, da auch ein Angreifer die serverinternen eingesetzten SQL-Statements nur erraten kann. Darüber hinaus sind oft sehr simple SQL-Statements das Ziel einer SQL- Injection, da bei komplizierten SQL-Anfragen viele Versuche der SQL-Injection benötigt werden.

5.2.1 WHERE

Eine WHERE-Injection ist eine Injection, wobei der Angreifer das setzten einer Bedingungen (WHERE) ausnutzt, um dies zu verdeutlichen, dient folgendes Beispiel:

URL	www.stundenplanservice.de/beispiel.php?mat_nummer=638475
SQL 1	SELECT name FROM studenten WHERE matrikelnummer = $_GET['mat_nummer'];
SQL 2	SELECT name FROM studenten WHERE matrikelnummer = 638475;

Listing 8: Beispiel einer WHERE-Injection

Das beispiel.php Skript erwartet, als GET-Parameter mat_nummer -Variable aus der URL, eine Matrikelnummer eines Studenten als Zahl. Mithilfe einer Übergabe von Sonderzeichen oder anderen Werten, könnte ein Angreifer eine Schwachstelle dieser SQL-Anfrage erraten. Wird dabei anstelle einer gültigen Matrikelnummer ein Mix aus Nummer und weiten SQL-Statements übergeben, könnte der Angreifer alle Namen aller Studenten erfragen:

URL	www.stundenplanservice.de/beispiel.php?mat_nummer=638475 OR true
SQL 1	SELECT name FROM studenten WHERE matrikelnummer = $_GET['mat_nummer'];
SQL 2	SELECT name FROM studenten WHERE matrikelnummer = 638475 OR true;

Listing 9: angepasste WHERE-Injection

Durch die Erweiterung der GET Anfrage um den Wert „OR true“, (Sharma, 2005, p. 6) werden alle Namen der Datenbank-Tabelle studenten ausgegeben, da diese Bedingung immer zutrifft. Somit muss bei dieser Beispiel-Injection nur die WHERE Bedingung manipuliert werden, um unautorisiert an viele oder alle Daten der Datenbank zu gelangen.

5.2.2 INSERT, UPDATE und DELETE

Nicht nur bei datenbankabfragenden SQL-Anweisungen ist die Gefahr einer Injection groß, sondern auch bei den Anweisungen INSERT, UPDATE und DELETE. Werden verarbeitenden SQL Anfragen manipuliert, sind die Folgen meist schwerwiegend, da hierbei Daten eingefügt, geändert oder gelöscht werden. Dies gelingt, da der ausführende Datenbank-Account die Rechte besitzen muss, alle Datensätze der jeweiligen Tabellen zu bearbeiten. Die Injection für UPDATE und DELETE unterscheidet sich dabei von der bereits vorgestellte WHERE Injection, da in diesen SQL Anfragen auch WHERE Bedingungen genutzt werden.

INSERT-Anfragen können Syntaxbedingt keine WHERE Klausel beinhalten (Oracle Corporation, 2019) und bieten dennoch eine mögliche Angriffsmethode.

```
1 | INSER INTO studenten VALUES (638475, „Moritz“, „E-Commerce“, “{Adresse}”)
```

Listing 10: Beispiel einer INSERT-Anfrage

Diese INSERT INTO Anweisung erstellt in der Tabelle studenten einen neuen Datensatz mit den angegebene Werten. Werden diese Werte ungeprüft, z. B. aus einem Formular, in die Anfrage übernommen, könnte ein Angreifer ohne großen Aufwand weitere Anweisungen in dem eigentlichen Statement hinein manipulieren.

```
1 | INSER INTO studenten VALUES (638475, „Moritz“, „E-Commerce“, (SELECT * FROM
  | studenten WHERE name = “Kevin” ));
```

Listing 11: angepasste INSERT-Injection

Nicht nur Zahlen und String können in Insert-Anweisungen genutzt werden, sondern auch SELECT-Anweisungen. Das daraus folgende Ergebnis wird dann in die INSERT-Anweisung aufgenommen und in den neuen Datensatz übernommen. Wenn anschließend der Nutzer Moritz Zugriff auf seine Daten erhält, sieht er in seiner Adress-Spalte die Adresse des Nutzers mit dem Name Kevin.

5.2.3 Abwehr von SQL-Injections

Die in den vorherigen Abschnitten erläuterten Angriffe auf einen Datenbank-Server lassen sich mit einfachen Mitteln abwehren. Eine wichtige Einstellung, welche vorgenommen werden sollte, sind die Zugriffsrechte auf den Server nur über den serverinternen Localhost[16] zu ermöglichen. Darüber hinaus sollte nicht der Nutzer root oder admin und ein einfaches oder kein Passwort verwendet werden. Um SQL-Injections effektiv verhindern zu können, sollten nur Prepared Statements eingesetzt werden. Bei Prepared Statements werden Benutzereingaben nicht direkt an den Datenbank-Server übergeben, sondern die Anfrage wird mit Platzhaltern auf dem diesem vorbereitet und anschließend mit Werten gefüllt. PHP selbst bietet das Modul PDO an, welches im kompletten Service bei Datenbank-Anfragen genutzt werden sollte.

[16] Localhost ist stellvertretend für die interne IP des Servers selbst

5.3 Absicherung der Datenbank durch Verschlüsselung

Um eine höhere Sicherheitsstufe zu erlangen, sollten bestimmte Datenbankinhalte – also alle oder einzelne Spalten einiger Tabellen – durch die Datenbank selbst (z. B. MariaDB) verschlüsselt werden. Anderweitige Verschlüsslungen sind dabei ebenso denkbar. Das Verschlüsseln einer Datenbank ist ein weiter Sicherheitsmechanismus zum Schutz des Servers. Somit kann kein Angreifer die Daten weiterverarbeiten, falls dieser Zugriff auf den Datenbank-Server erhält. Darüber hinaus finden sich auch in Datenbank-Dump-Dateien[17] nicht die realen, sondern nur die verschlüsselten Werte.

Besonders nachteilig an dieser Methode ist, dass die Lesbarkeit von anfallenden SQL-Statements deutlich erschwert wird. Darüber hinaus erhöhen diese Verschlüsselungsmethoden den Arbeitsaufwand der Datenbank enorm. Dies hat zur Folge, dass die Zugriffszeiten auf den Datenbank-Server erhöht werden.

Zur Demonstration wird eine Datenbank Tabelle erstellt.

```
1  create table Stundenplan(room VARCHAR(256), module VARCHAR(256))
```

Listing 12: Beispiel zur Anlage einer verschlüsselten Tabelle

Ein Datensatz wird mittels INSERT INTO eingefügt. Dabei wird der Wert ‚E-Commerce' durch eine AES Verschlüsslung und dem Sicherheitsstring „LUYLA64u27PON9DM' verschlüsselt.

```
1  INSERT INTO Stundenplan (room, module) VALUES('3.33.33',
   AES_ENCRYPT(E-Commerce, "LUYLA64u27PON9DM"));
```

Listing 13: Beispiel eines verschlüsselten INSERT-Statements

Eine einfache Auslesung des Moduls-Namens ist ohne den genutzten Sicherheitsstring nicht mehr möglich:

SQL	SELECT name, modul FROM Stundenplan;
Spalten	room | module
Ergebnis	3.33.33 | 059828E1D108D7EC1B29Ae8D504FA0935

Listing 14: Beispiel eines verschlüsselten SELECT-Statements

[17] Abbildung von ganzen Tabellen oder Datenbanken in Textdateien.

Für die eigentliche Ver- und Entschlüsselung bietet MariaDB die Funktionen AES_ENCRYPT und AES_DECRYPT an (MariaDB, 2019).

AES_ENCRYPT() und AES_DECRYPT() ermöglichen die Ver- und Entschlüsselung von Daten mit dem offiziellen AES-Algorithmus (Advanced Encryption Standard) (MariaDB, 2019).

Besonders personenbezogene Daten vom Nutzer, welche im Kapitel 7 näher betrachtet werden, sollten nicht unverschlüsselt in einer Datenbank gespeichert werden. Dabei sollte bei der Umsetzung des serverseitigen Service, eine Datenbank interne Verschlüsslung oder andere Verschlüsselungsmethode unbedingt eingesetzt werden.

Abschließend kann gesagt werden, dass bei der Umsetzung eines Service, welcher über das öffentliche Internet verfügbar und abrufbar sein soll, viele Punkte rund um das Thema Sicherheit bedacht werden müssen.

6 Push Benachrichtigung

Um die geplanten Push Benachrichtigungen umzusetzen wird ein Push Notification Service benötigt. In diesem Kapitel werden die Übertragungswege (Push- bzw. Pull-Prinzip) behandelt. Auf Basis dieser Prinzipien wird anschließend die Funktion eines Push Notification Service anhand des Google Firebase Push Notification Service erläutert.

6.1 Push vs. Pull Prinzip

Wie in der Abbildung 7 zu sehen ist, gibt es grundsätzlich zwei übliche Wege, wie Informationen in über das Internet übertragen werden, die Push- und die Pull-Variante.

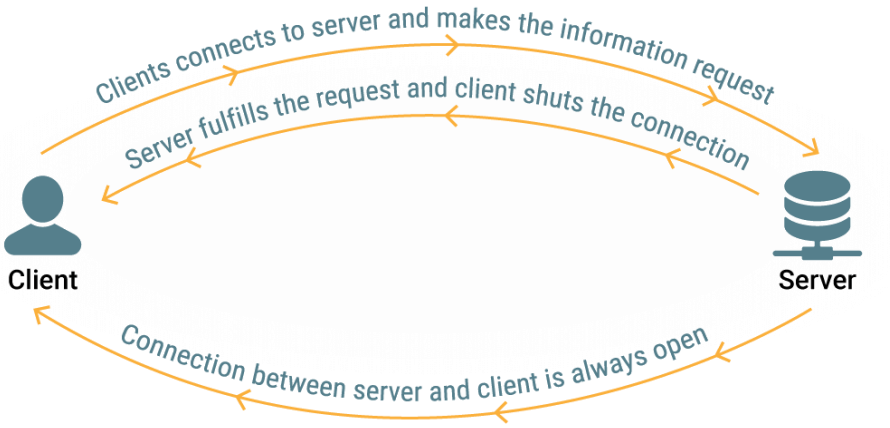

Abbildung 7: Informationsflüsse im Internet | Quelle: (Singh, 2019)

Werden Informationen vom Sender zum Client nach dem klassischen Sender-Empfänger-Modell gesendet, so spricht man von Push-Kommunikation (engl. „to push“ = drücken). Dies ist eine Form der Informationsübertragung, welche hauptsächlich vom Server ausgeht. Die primären Funktionen der Kommunikation sind hier die Informationsfunktion, Beeinflussungsfunktion und Bestätigungsfunktion. Pull-Kommunikation (engl. „to pull“ = ziehen) ist hingegen zweiseitig. Sie geht sowohl vom Client als auch vom Server aus(Singh, 2019).

Bei mobilen Geräten werden hauptsächlich Push-Benachrichtigung verwendet, die vom App-Server an die Geräteoberfläche des Clients gesendet wird, da so auch Nachrichten gesendet werden können, selbst wenn der Benutzer nicht angemeldet oder die App nicht offen ist.

6.2 Google Firebase Push Notification Service

Um Push-Benachrichtigungen empfangen zu können, muss die App bei einem Push-Benachrichtigungsdienstanbieter konfiguriert und registriert sein. In dieser Arbeit wird Google Firebase Push Notification Service als Benachrichtigungsdienstanbieter verwendet, da dieser ohne großen Aufwand in eine Android oder IOS App integriert werden kann. Darüber hinaus wurde dieser Benachrichtigungsdienstanbieter für die Umsetzung des ganzen Projektes gewählt.

Der Google Firebase Push Notification Service ermöglicht das Senden von Push-Benachrichtigungen an die Nutzer eines mobilen Device. Mit diesem Service verwaltet der Cloud Connection Server von Google, die dauerhaften Verbindungen zu dem mobilen Endgerät. Außerdem stellt dieser sicher, dass alle Push-Benachrichtigungen sicher und zuverlässig übermittelt werden.

Um eine Push Benachrichtigung, über den serverseitigen Service an einen bestimmten App Nutzer, senden zu können, muss zuvor die Android oder IOS App um den Firebase Cloud Messaging Dienst erweitert werden.

Die Implementierung des Firebase Cloud Messaging Dienst ist nicht Teil der Umsetzung dieses Bachelor Projekt und wird deswegen nur kurz am Beispiel einer Android App erläutert:

1. Als Erstes muss das bereits bestehende App Projekt in der Firebase Konsole[18] registriert und eingerichtet werden. Eine anschließend zum Download bereitstehende Konfiguration Datei muss in das App Projekt eingebunden werden. (Google, 2019)
2. Um Firebase Produkte zu aktivieren, muss ein sog. Google-Service Plug-Ins in der {Projekt}/build.gradle hinzugefügt werden:

```
1  classpath 'com.google.gms:google-services:4.3.2'  (Google, 2019)
```

3. Nun muss das Plug-In im App-Level aktiviert werden - {App}/build.gradle

```
1  apply plugin: 'com.google.gms.google-services' (Google, 2019)
```

   Mit dieser Implementierung des Google-Service Plug-Ins, können nun alle Firebase Produkte, in diesem Fall der benötigte Cloud Messaging Dienst, inkludiert und genutzt werden.

[18] https://console.firebase.google.com/

4. Es muss die Abhängigkeit des Cloud Messaging Dienstes in der build.gradle (meistens app/ build.gradle) hinzugefügt werden. Dies implementiert die Firebase SDK in die Android App und geschieht über folgende Codezeile:

```
1  „implementation'com.google.firebase:firebase-messaging:20.0.0' (Google, 2019)
```

5. Die Firebase SDK muss für die Android App aktiv genutzt werden. Dies geschieht über die Aktivierung der SDK in der AndroidManifest.xml mithilfe folgenden Codes:

```
1  <meta-dataandroid:name="com.google.firebase.messaging.
   default_notification_channel_id"
   android:value="@string/default_notification_channel_id" />
```

(Google, 2019)

6. Anschließend kann in der App ein Firebase Client erstellt und durch folgenden stark verkürzten Code der einzigartige Device Token (Variable String token), für dieses Smartphone und App, abgerufen werden:

```
1  FirebaseInstanceId.getInstance().getInstanceId()
   ….
2  String token = task.getResult().getToken();
   ….
```

(Google, 2019)

Dieser Device Token ist solange valid, wie er der Kombination aus Smartphone und App zugewiesen ist. Darüber hinaus verliert ein Token seine Funktion, wenn die App eigenständig den Device Token löscht oder der Nutzer die App oder dessen Daten löscht. (Google, 2019)

Mit diesen generierten Device Token erfolgt die Identifizierung eins Smartphones, an welches eine Push Benachrichtigung gesendet werden kann. Der Device Token ist ein einmalig vorkommender Schlüssel, der für die Authentifizierung zwischen dem mobilen Device und dem Push Notification Service verwendet wird und ist zwingend nötig, um mithilfe eines serverseitigen Service eine Push Benachrichtigung an ein bestimmtes Smartphone zu senden.

6.3 Senden einer Push Benachrichtigung

Das Senden einer Push Benachrichtigung kann, beim Nutzen des Google Firebase Push Notification Service, über viele Arten und Methoden ermöglicht werden. Im Umfang dieser Arbeit wird zur Methode Rest-HTTP Protokoll via cURL[19] Stellung bezogen. cURL ist eine Bibliothek, welche es ermöglicht, über URLs Daten und Abfragen zu senden. cURL unterstützt allerhand Transferprotokolle, darunter auch HTTP und HTTPs, welche im Kapitel 4 erläutert sind. Es kann angenommen werden, dass jeder Datentransfer im Internet mit cURL in Verbindung gebracht werden kann. (Stenberg, 2017, p. 13) Diese Bibliothek ist bereit im empfohlen LAMP-Paket enthalten und müsste nicht eigenständig installiert und eingerichtet werden.

Wie bereits im Abschnitt 4.3.3 beschriebenen, werden bei Rest Schnittstellen bestimmte Ressourcen immer über festgelegte URLs Endpunkte erreicht. Die Ressource - Push Benachrichtigung Senden - kann bei Google Firebase über folgenden Endpunkt erreicht werden:

1	https://fcm.googleapis.com/fcm/send

Um eine einfache Push Benachrichtigung in Auftrag geben zu können, müssen an diesen Endpunkt mindestens folgende Informationen gesendet werden:

- **Wer bin ich und darf ich Push Benachrichtigung in Auftrag geben?**
  Mithilfe eines Schlüssels kann die Anfrage verifiziert und zugeordnet werden. Dieser Schlüssel kann in den jeweiligen Projekteinstellungen in der Google Firebase Konsole eingerichtet und eingesehen werden.
- **Wer soll diese Benachrichtigung empfangen?**
  Es muss explizit angegeben werden, welche Smartphones diese Push Benachrichtigung erhalten sollen. Dies erfolgt durch die Übergabe aller Device Tokens.
- **Wie lautet die Benachrichtigung?**
  Eine Benachrichtigung sollte einen gewissen Inhalt besitzen. Somit sollte mindestens eine Nachricht aus Titel und Nachrichtentext übermittelt werden.

[19] Abkürzung für Client for URLs

```
1 curl “ https://fcm.googleapis.com/fcm/send“
2 -X POST –H
3 “Authorization: key= MY_AUTH_KEY”
4 -H “Content-Type: application/json” –d
5 ‘{"registration_ids":["DEVICE TOKEN"],
6 "notification": {"title": "ICH BIN DER TITEL"}, "body": "ICH BIN DER TEXT"}'
```

Listing 15 Beispiel Push Benachrichtigung via cURL

Listing 15 stellt einen cURL Aufruf dar, um eine Push Benachrichtigung bei Google Firebase in Auftrag zu geben. In Zeile eins wird die Bibliothek mittels – curl - aufgerufen und angegeben an welche URL diese Anfrage gesendet werden soll. POST wird in Zeile zwei als HTTP Anfragemethoden eingesetzt. In Zeile drei und vier werde bestimmte Header Informationen übermittelt. In diesem Fall wird der Schlüssel und die Information, dass der Anfragen Body im Format JSON übergeben wird, angegeben. Zeile fünf und sechs stellen den Anfragen Body dar. Hierbei werden im Punkt - registration_ids - alle Device Tokens eingesetzt, welche diese Benachrichtigung erhalten sollen. Darüber hinaus wird im Punkt - notification - die eigentliche Benachrichtigung mit Titel und Body angegeben.

Der Google Firebase Push Notification Service kann nach dieser technischen Betrachtung im Umfang des geplanten Projektes eingesetzt werden. Die Integration in eine Android oder IOS App ist, durch die gute Dokumentation seitens Google, umsetzbar. Darüber hinaus ist das Versenden von Push Benachrichtigungen über den geplanten serverseitigen Service problemlos realisierbar.

7 Datenschutz

Der Begriff Datenschutz in Kombination mit dem Ausdruck DSGVO war in den letzten Jahren in aller Munde. Das Thema Datenschutz ist in der heutigen Zeit sehr bedeutsam und sollte in jeder Planung eines neuen IT-Systems oder Service bedacht werden. Somit wird in diesem Abschnitt Stellung bezogen zu datenschutzrelevanten Themen. Anschließend wird erläutert ob der Device Token, um Push Benachrichtigungen zu versenden, eine personenbezogene Information ist und wie diese im System behandelt werden sollte.

Zuerst werden einige Begrifflichkeiten geklärt, um dieses Thema anschließend näher zu betrachten:

Datenschutz als Begriff

Der Datenschutz wird oft mit den Worten „Schutz der Daten“ beschrieben, welches aber keineswegs stimmt. Der Datenschutz richtet sich ausschließlich auf die Privatsphäre und das Persönlichkeitsrecht von natürlichen Personen. Dabei schützt der Datenschutz das individuelle Recht jeder einzelnen Person.

Person

Personen sind im Sinne des Datenschutzes nur natürliche Menschen, da nur diese Personen ein Recht auf Datenschutz eingeräumt wird.

Personenbezogene Daten

Wenn bestimmte Informationen, sich direkt oder indirekt auf natürliche Personen beziehen, sind dies personenbezogene Daten. Dabei ist es unerheblich, ob die Person direkt über ihren Namen oder indirekt durch eine gewisse Kennung, wie Telefonnummer, Adresse oder Matrikelnummer, identifiziert werden kann.

Besondere Arten personenbezogener Daten

Besondere Arten von personenbezogenen Daten sind Daten, welche die betroffenen Personen besonders stark benachteiligen könnte. Zu diesen besonderen Arten gehören Daten zu rassische und ethnische Herkunft, politische Meinung, religiöse oder philosophische Überzeugung, Gewerkschaftszugehörigkeit, Gesundheit oder Sexualleben.[20] (Handelskammer Hamburg, 2019)

[20] Art. 9 Nr. 1 DSGVO

Verarbeitung

Jeder Umgang oder Nutzen von personenbezogenen Daten ist eine Verarbeitung im Sinne des Datenschutzes. Dabei ist es nicht relevant ob die Daten mithilfe von IT-Systemen verarbeitet werden oder diese Tätigkeiten manuell durchgeführt werden (Meraneos, 2017). Ein Beispiel für eine manuelle Tätigkeit ist das Aufschreiben von personenbezogenen Daten auf einen Notizblock.

7.1 Datenschutzgrundverordnung

Im Folgenden wird die häufig „gefürchtete“ Datenschutzgrundverordnung kurz vorgestellt.

Die Datenschutzgrundverordnung, kurz DSGVO, ist eine Verordnung der Europäischen Union, welche am 25 Mai 2018 in Kraft getreten ist. Die DSGVO soll die gesetzlichen Bedingungen der Verarbeitung von personenbezogenen Daten innerhalb der EU vereinheitlichen. Dabei wurde diese Verordnung unmittelbar geltendes EU-Recht und muss nicht in das nationale Recht der einzelnen Länder übernommen werden. (Union, 2019) Den einzelnen Ländern ist es hierbei erlaubt, die verordnenden Gesetzten national weiter zu verschärfen. Dagegen ist eine Abschwächung der DSGVO, durch nationale Reglungen, nicht gestattet. Somit stellt die DSGVO die Grundlage des Rechtsverständnisses des Datenschutzes in der EU dar.

7.2 Grundsätze

Die DSGVO beschreibt im Artikel 5 allgemeine Grundsätze, wie personenbezogene Daten verwendet werden dürfen. Diese Grundsätze werden anschließend in weiteren Artikeln der DSGVO konkretisiert. Im Folgenden werden einige wichtige Grundsätze erläutert und analysiert, ob und wie diese bei der Umsetzung des geplanten serverseitigen Service bedacht werden müssen. Dabei wird bei dieser Analyse davon ausgegangen, dass personenbezogene Daten eventuell vom geplanten Service verarbeitet werden.

7.2.1 Rechtmäßigkeit [21]

Die Verarbeitung von personenbezogenen Daten ist grundsätzlich verboten. Es ist ein Verbot mit einem sog. Erlaubnisvorbehalt. Dies bedeutet, dass eine Verarbeitung von personenbezogenen Daten dennoch möglich ist, wenn eine „wichtige Rechtsgrundlage“ (Meraneos, 2017, p. 12) besteht.

[21] Art 6 DSGVO

Dabei sind wichtige Rechtsgrundlagen unter anderem z. B. das öffentliches Interesse, die Einhaltung von Verträgen, lebenswichtige Interesse der betroffen Person oder Dritter, aber auch die Einwilligung der betroffenen Person.

Damit der geplante Service diesen Grundsatz erfüllt, sollte der Nutzer eine selbstständige Willenserklärung abgeben. Da der geplante Service nur indirekt, über die App, mit dem Nutzer in Kontakt steht, muss die eventuell erforderliche Willenserklärung vor der App-Installation oder vor dem Einrichten des geplanten individuellen Stundenplanes vom Nutzer eingefordert werden.

7.2.2 Fairness [22]

Der Grundsatz Fairness beschreibt das Faire behandelt von betroffenen Personen. Das Verhalten des Verarbeiters soll dem eines anständigen Menschen entsprechen. Eine Person sollte von Anfang an wissen, wie seine Daten verarbeitet werden und worauf er sich einlässt. Dieser Grundsatz wird im deutschen Gesetz als „Treu und Glauben“ bezeichnet.

Nach diesem Grundsatz darf der geplante Service mögliche personenbezogene Daten nicht nutzen, um z. B. eine Statistik über die Studenten zu erstellen und diese zu veröffentlichen, da der Nutzer, nach diesem Grundsatz, seine Willenserklärung nur für einen Service erteilt hat, welcher geplante Push Benachrichtigungen umsetzt.

7.2.3 Transparenz [23]

Wenn personenbezogene Daten verarbeitet werden, muss für die betroffene Person klar ersichtlich sein, was mit seinen Daten passiert. Darüber hinaus muss transparent und verständlich dargestellt werden, wie genau und im welchem Umfang die personenbezogenen Daten verarbeitet werden (Meraneos, 2017, p. 20). Dabei bezieht sich dieser Grundsatz nicht nur auf die Zeit vor der Datenerhebung, sondern auf die komplette Zeit der Datenverarbeitung und auch darüber hinaus.

Die geplante App muss somit jederzeit dem Nutzer darüber informieren können, welche Daten vom Nutzer erhoben und wie und warum diese Dateien verarbeitet werden.

[22] Art. 5 DSGVO

[23] Erwägungsgrund 58 der DSGVO

7.2.4 Zweckbindung[24]

Nach dem Grundsatz der Zweckbindung dürfen personenbezogene Daten nur für einen rechtmäßigen Zweck verarbeitet werden (Meraneos, 2017, p. 22). Wenn die betroffenen Personen darüber klar informiert werden und zustimmen, können Daten aus einer Datenerhebung auch für mehrere Zwecke genutzt werden. Darüber hinaus räumt die DSGVO ein, dass personenbezogene Daten auch weiter verarbeitet werden dürfen, wenn eine Rechtsvorschrift vorliegt. Ein Beispiel hierfür ist das erforderliche Aufbewahren von gewissen personenbezogenen Daten für das Finanzamt.

Eine Bereitstellung von einem individuellen Stundenplan und daraus resultierenden Push Benachrichtigungen ist ein rechtmäßiger Zweck einer möglichen Datenverarbeitung. Somit verstößt der geplante Service nicht gegen die DSGVO.

7.2.5 Datenminimierung[25]

Die eingeforderten Daten, welche verarbeitet werden sollen, dürfen nur dem zweckentsprechend und angemessen sein. Es sollen keine Daten erhoben werden, welche nicht zwingend notwendig sind, um die ausgewiesene Dienstleistung zu erbringen. Das Ziel ist somit, „so wenig personenbezogene Daten wie möglich zu verarbeiten“. (Prof. Dr. Sibylle Gierschmann, 2018) .

Somit sollte der geplante Service nur die Daten vom Nutzer erheben und verarbeiten, welche notwendig sind, um den angesprochen Service bereitzustellen. Nicht notwendige Informationen sollten strikt nicht erhoben oder gespeichert werden. Dieser Grundsatz kommt nicht nur dem Datenschutz zugute, sondern minimiert auch die Last des Systems.

7.2.6 Datenrichtigkeit[26]

Nach dem Grundsatz der Datenrichtigkeit hat der Betroffene das Recht, dass seine Daten korrekt und „auf dem neuesten Stand“ sind. (Meraneos, 2017, p. 24) Der Verantwortliche hat hierbei die Aufgabe, fehlerhafte Daten unverzüglich zu löschen oder zu korrigieren.

---

[24] Art. 5 Abs. 1 DSGVO

[25] Art 5 Abs. 1/c DSGVO

[26] Art 5 DSGVO

Wenn die geplante App oder der Service gewisse personenbezogene Daten der Nutzer verarbeitet, muss dabei im Service eine Möglichkeit geschaffen werden, da fehlerhafte Daten vom Nutzer selbst, oder vom Verarbeiter korrigiert werden können.

7.2.7 Speicherbegrenzung [27]

Personenbezogene Daten dürfen nur solange gespeichert werden, wie es für die Verarbeitungen erforderlich ist. (Meraneos, 2017, p. 25)

Im Falle der App verfällt nach jedem Semester die Erforderlichkeit die erhobenen personenbezogenen Daten, des vergangenen Semesters, weiter zu speichern.

Somit sollte schon serviceintern eine Funktion eingebaut werden, welche ein Semesterende erkennt und nicht mehr benötigte Daten löscht.

7.2.8 Recht auf Löschung [28]

Die DSGVO verpflichtet Verantwortliche, personenbezogene Daten auf Verlangen des Betroffenen zu löschen. Darüber hinaus beinhaltet dieser Grundsatz auch das proaktive Löschen von personenbezogenen Daten als Pflicht für den Verantwortlichen. Dabei werden in der DSGVO gewisse Kriterien der Löschung beschrieben. Diese Kriterien lassen sich in die Gruppen „ Löschen auf Verlangen“ und „selbstständiges Löschen“ einordnen.

Die App, in Kommunikation mit dem Service, sollte eine Funktion bereitstellen, womit ein Nutzer all seine erhobenen Daten löschen lassen kann.

7.2.9 Datensicherheit[29]

Im Grundsatz der Datensicherheit wird die allgemeine Sicherheit der Daten beschrieben. Eine angemessene Sicherheit muss hierbei jederzeit gewährleistet werden, wenn personenbezogene Daten verarbeitet werden. Dabei müssen “technische und organisatorische Schutzmaßnahmen“ (Meraneos, 2017, p. 36) getroffen werden, damit kein unbefugter Zugriff, ungeplante Verbreitung, Löschung oder Manipulation der Daten eintreten kann. Als Mindestanforderung werden verschiedene Maßnahmen in der DSGVO genannt, wie die Verschlüsslung von Daten, wenn diese gespeichert werden.

---

[27] Art. 5 Abs. 1/e DSGVO

[28] Art. 17 DSGVO

[29] Art. 32 DSGVO

7.3 Einschätzung des geplanten Service

Der geplante serverseitige Service sollte die absolute Datenminimierung anstreben, um unnötige Datenschutz-Konflikte zu vermeiden. Dies bedeutet, dass keine personenbezogenen Daten vom Nutzer an den serverseitigen Service gesendet, oder von diesem empfangen, gespeichert und ausgewertet werden sollte. Darüber hinaus sollte er keine Daten speichern oder auswerten, welche nicht dem eigentlichen Ziel des Service entsprechen. Der benötigten Device Token, um Push Benachrichtigungen zu versenden, ist eine anonymisierte Geräte-ID. Mithilfe dieser ID lassen sich keine personenbezogenen Daten herleiten oder bei Google Firebase oder dem Smartphone selbst abfragen. Auch wenn dieser Device Token keine personenbezogene Information ist, sollte er im Rahmen des Projektes wie eine behandelt werden. Somit sollte diese nur streng vertraulich im System des serverseitigen Service genutzt werden. Besonders bei der Speicherung dieses Device Token, sollte dieser nur verschlüsselt in einer Datenbank abgelegt werden, um mögliche Risiken weiter zu reduzieren. Auch die Information, über die vom Nutzer abonnierten Module, sollten niemals mit personenbezogenen Nutzerdaten in Verbindung gebracht werden. Diese Informationen sollten ausschließlich dem anonymisierten Device Token zugeordnet werden. Der geplante Service sollte einen Schnittstellen Endpunkt anbieten, damit der Nutzer, über die App, jederzeit sich vom Push Benachrichtigung Service abmelden und seine Informationen über abonnierte Module und den Device Token selbst löschen kann.

Alle Informationen aller Stundenpläne der EAH Jena sind nicht datenschutzrelevant und benötigen somit keiner weiteren Betrachtung.

8 Umsetzung

In diesem Kapitel wird die Umsetzung des analysierten Service beschrieben. Dabei werden nur interessante und bedeutsame Bereiche der Umsetzung erläutert. Der Quellcode des erstellten serverseitigen Service liegt dieser Arbeit bei und wird hierbei punktuell in diesem Kapitel aufgezeigt.

Die Umsetzung lässt sich in mehrere einzelne Arbeitspakete aufteilen, welche meist parallel umgesetzt worden sind und im Folgenden beschrieben werden.

8.1 Einstellung

Das Projekt enthält gewisse Variablen, welche wiederholt benötigt und genutzt werden. Hierfür wurde eine Konfiguration erstellt, worin alle spezifizierbaren Variablen des ganzen Services definiert werden. Zu finden ist diese Konfigurationsdatei unter /settings/settings.php. In dieser Datei werden folgende Variablen definiert und müssen vor Servicestart dem jeweiligen System und Umgebung angepasst werden:

- Datenbank Informationen
- Google Firebase Schlüssel
- Link zur genutzten OPML Datei und Domain des SPlus Server
- Zwei Geheim Schlüssel, zum Verschlüsseln der Device Tokens
- Fehlerbehandlung für PDO und Slim
- Zeitzone des Projektes

8.2 Datenbank

In diesem Abschnitt wird der Datenbankaufbau beschrieben.

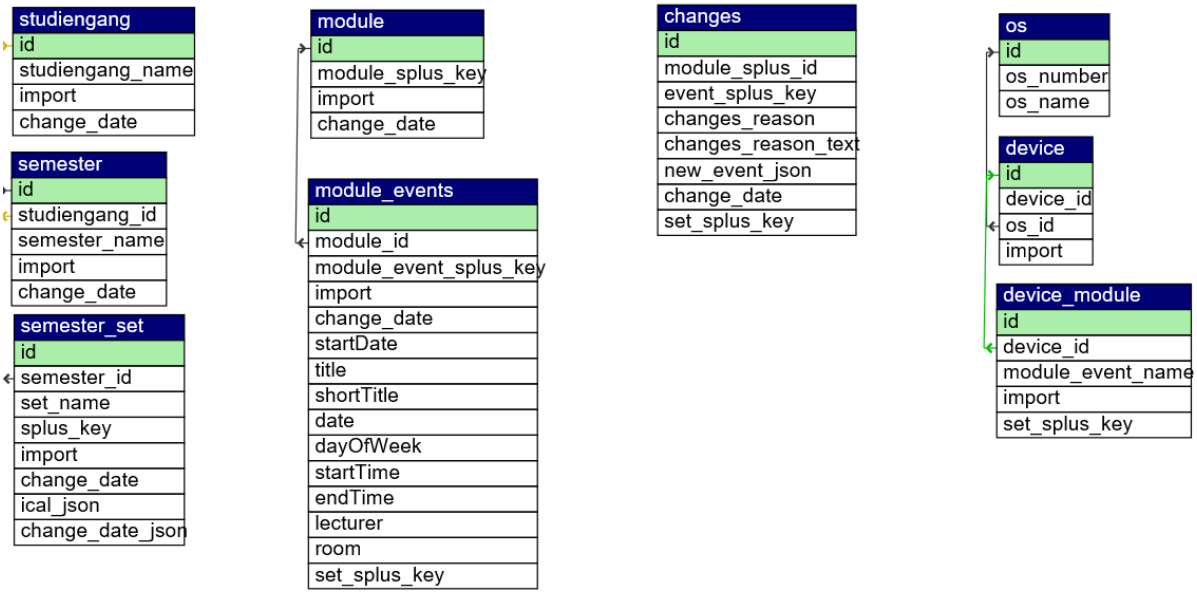

Abbildung 8 Datenbankaufbau

Die Abbildung 8 zeigt den umgesetzten Datenbankaufbau des Service. Durch den gewissen Umfang des Projektes wurden mehrere Datenbank Tabellen eingesetzt, welche sich in folgende vier Gruppen einteilen lassen:

- **Studiengänge, Semester, Sets**
  Diese Gruppe besteht aus drei einzelnen Datenbanktabellen und beinhalten jeweils alle Studiengänge, Semester von Studiengängen und alle Sets der Semester.
- **Module, Events**
  In diesen beiden Gruppen von Tabellen werden alle Module und alle Events mit ihren verfügbaren Informationen aktuell gehalten.
- **Änderungen**
  In der Tabelle changes werden alle erkannten Eventänderungen hinterlegt.
- **Abonnements | os, device, device_module**
  Diese Tabellengruppe umfasst alle Informationen zu Abonnements, also welches Device welche Module abonniert hat und über Änderungen dieser informiert werden möchte.

Diese Datenbankstruktur ist nötig, um alle im Service anfallenden Daten strukturiert abspeichern zu können. Dabei wurde stets darauf geachtet, die Datenbank Tabellen und ihre Spalten zweckmäßig nach ihrem Inhalt zu benennen.

Darüber hinaus wurde für alle Spalten ein für sie geeigneter Typ gewählt. So werden Zahlen als Integer, Wörter und kleine Texte als Varchar und alle Datumsangaben als Datetime gespeichert.

Mit dieser Datenbankstruktur lässt sich der Aufbau aller Studiengänge, Semester der Studiengänge, Sets, Module und Events ideal in der Datenbank abbilden und pflegen. Darüber hinaus lassen sich alle Abonnements von Modulen durch Devices gesondert verwalten.

8.3 Interner Prozess Ablauf

Der interne Prozessablauf ist der wesentlichste Teil der Umsetzung des Service. Dieser ist im Quellcode-Ordner builder auffindbar und umfasst die komplette Prozedur, um Änderungen in den Stundenplänen der EAH Jena zu finden und Push Benachrichtigungen zu versenden. Im Folgenden werden Besonderheiten dieses Ablaufes beschrieben.

8.3.1 Studiengänge, Semester und Sets

Das aktuell halten aller Studiengänge, Semester und Sets ist elementar für den Ablauf des Service, da besonders bei Semesterstart sich der Aufbau der Set Struktur je Semester häufiger ändern kann. Besonders die Datenstruktur der Sets muss im Service stets aktuell sein, da mithilfe deren SPlus Schlüssel die Kalenderdateien je Set vom Server der EAH Jena abgefragt werden. Wie in der Konzeption analysiert, wird hierfür eine gewisse OPML Datei genutzt. Diese liegt im XML Dateiformat vor und muss mittel folgender kleinen Konvertierung in ein PHP konformes Array übersetzt werden:

```
1 $omp = file_contents_exist($link);
2 $xml   = simplexml_load_string($omp, "SimpleXMLElement");
3 $json  = json_encode($xml);
4 $array = json_decode($json, TRUE);
```

Die OPML XML Datei wird hierbei mithilfe der Funktion file_contents_exits heruntergeladen und liegt dann im String Format in der Variablen $omp. Dieser String wird anschließend mit simplexml_load_string in ein PHP Objekt und anschließend durch json_encode in ein JSON-konformen String konvertiert, welcher dann durch json_decode in ein einfach weiter zu verwendendes PHP Array gewandelt wird. Dieses PHP Array wird geparst und der daraus resultierende Datensatz wird in der Datenbank eingepflegt.
Somit stehen dem weiteren Service stets alle Studiengänge, Semester und die wichtigen Sets mit ihren SPlus Schlüsseln zur Verfügung.

8.3.2 Einlesen der Kalenderdateien

Das Einlesen der Kalenderdateien stellte eine besondere Herausforderung bei der Umsetzung des Service dar. Das besondere an diesen Kalenderdateien ist, wie bereits erwähnt, ihr besonderes Dateiformat - iCal. Mit Bordmittel der Programmiersprache PHP lässt sich dieses Dateiformt nicht, oder nur schwer, einlesen. Hierfür wurde eine frei verfügbare iCal PHP Bibliothek[30] eingesetzt. Diese Bibliothek ebnete den Weg die Kalenderdateien in ein PHP Array zu konvertieren. Diese Bibliothek musste im Rahmen der EAH Jena Kalenderdateien nur um ein paar entscheidende Zeilen Code erweitert werden. Das größte Problem bestand darin, dass die Kalenderdateien in ihrer Start und Endzeitangabe eines Events keine Zeitzone übermitteln, dieses aber vom Parser der eingebunden Bibliothek, unbedingt benötigt worden ist, um mittels regulären Ausdruck die Start und Endzeit zu ermitteln.
Beispielzeile aus Kalenderdatei:

```
1 DTEND:20190506T110000
```

Bibliothek – Parser Codezeile, welche zwar das Datum, aber nicht die Zeit ermitteln kann:

```
1 if (preg_match('`^DTEND(?:;.+)?:([0-9]+(T[0-9]+Z)?)`m', $content, $m)) {
            $this->_timeEnd = strtotime($m[1]);
            $this->dateEnd  = date('Y-m-d H:i:s', $this->_timeEnd);}
```

Listing 16 Beispiel vom unangepassten regulären Ausdruck

Hinzugefügte Zeile, um die Zeit heraus lesen zu könne, welche einen veränderten regulären Ausdruck nutzt und dessen Ergebnis mithilfe der Funktion substr_replace und substr bearbeitet:

```
1 if (preg_match('^DTEND(?:;.+)?:([0-9]+(T)?)?([0-9]+()?)^', $content, $m))
  {$this->timeEnd = substr( substr_replace($m[3], ":", 2, 0),0,5);}
```

Listing 17 Beispiel vom angepassten regulären Ausdruck

Die jeweilig Set spezifische geparste Kalenderdatei wird nun in JSON konvertiert und in der Datenbank Tabelle semester_set in der Spalte ical_json abgelegt.

[30] URL: https://gist.github.com/seebz/c00a38d9520e035a6a8c

8.3.3 Aktualisierung und Änderungserkennung der Events

Die Einarbeitung der Events und Erkennung aller angefallen Änderungen ist die Hauptaufgabe dieses Prozesses. Hierfür werden alle bereits geparsten Kalenderdateien aus der Spalte ical_json der Tabelle semester_set nacheinander nach folgendem Workflow abgearbeitet:

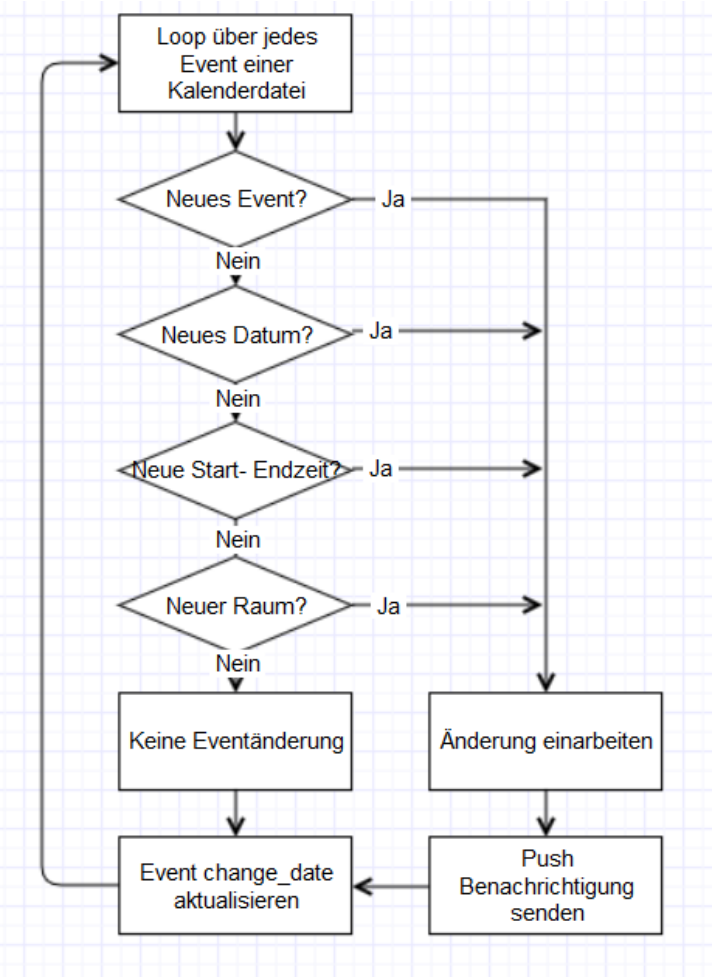

Abbildung 9 Workflow Eventänderung [31]

Jedes verfügbare Event einer Kalenderdatei durchläuft einen gewissen Workflow, welche alle Änderungen eines Events erkennen soll. Eine Änderung eines Events kann eine neue Hinzufügung, eine Änderung des Datums, Start und Endzeit oder Raum oder Löschung sein. Wenn eine dieser Änderung an einem Event erkannt wird, wird die Änderung in die module_events Tabelle eingearbeitet und alle Änderungsinformationen in der changes Tabelle abgespeichert. Besonders entscheiden ist hierbei die Speicherung des Änderungsgrunds, da dieser von der neuen EAH Jena App benötigt wird, wenn diese die Änderung abfragt.

[31] Eigene Abbildung

8.3.4 Senden von Push Benachrichtigungen

Wenn eine Änderung, Hinzufügung oder Löschung erkannt worden ist, wird jeder Nutzer, welcher dieses geänderte Modul abonniert hat, mittels Push Benachrichtigung, über diese Änderung informiert. Hierfür werden alle Nutzer Device Token aus der device Tabelle geholte, welche dieses Modul in genau diesem überprüften Set abonniert hat. Dabei wird das abonnierte Modul nicht über eine festgesetzte ID oder Schlüssel identifiziert, sondern über den Titel. Wenn z. B. ein Nutzer ein Modul – Mathe/V – mit einem eindeutigen SPlus Schlüssel aus einem bestimmten Set abonniert hat, erhält er eine Push Benachrichtigung für alle Eventänderungen aus diesem bestimmten Set, welche den String – Mathe/V – im Titel des Events haben. Dieses Verfahren des Titelabgleiches ist nötig, da eine Eventhinzufügung unter Umständen eine SPlus Schlüssel Änderung mit sich ziehen kann. Diese Abfrage an die Datenbank erfolgt über folgende SQL Anfrage:

```
1  SELECT device.device_id,device.os_id FROM device INNER JOIN device_module
   ON device.id = device_module.device_id WHERE device_module.set_splus_key
   = {Set SPuls Key} AND LOCATE(device_module.module_event_name, {Event
   Titel})>0
```

Listing 18 SELECT Anfrage mithilfe von LOCATE

Die Rückgabe dieser Anfrage erhält dann alle Device Tokens in Kombination mit der Betriebssystemangabe, welche von einer bestimmten Eventänderung betroffen sind.
Zur Umsetzung der Firebase Push Benachrichtigungen wurde eine dafür zugeschnittene und geeignete freie Bibliothek, namens php-fcm, eingesetzt.[32] Diese Bibliothek ermöglicht ein einfaches Senden von Push Benachrichtigungen an ein oder mehrere Device Token.

```
1  require 'vendor/autoload.php';
2  $firebase_client = new \Fcm\FcmClient(apiToken, senderId);
3  $notification = $firebase_client->pushNotification({Title},{Nachricht},
   {Device Token Array});
4  $response = $firebase_client->send($notification);
```

Listing 19 Senden einer Push B. mithilfe der php-FCM Bibliothek

Diese Bibliothek wird in Zeile eins über den Composer in das laufende PHP Skript eingebunden. In Zeile zwei wird mit dem Aufruf – new \Fcm\FcmClient – eine Firebase Objekt - $firebase:client - erstellt. Dieser Aufruf benötigt einen gewissen apiToken und eine senderId, welche beide in den Firebase Konsolen Einstellung des Projektes zu finden sind.

[32] URL: https://github.com/EdwinHoksberg/php-fcm

Die eigentliche Push Benachrichtigung wird in Zeile drei intern über das erstellte Firebase Objekt instanziiert. Hierfür wird der Titel und Nachricht der geplanten Push Benachrichtigung und alle Device Token, welche diese Nachricht erhalten sollen, übergeben. Zeile vier stellt hierbei das Absenden der Push Benachrichtigung dar.

8.4 Umsetzung der Schnittstelle

Die geplante neue EAH Jena App soll mit diesem Service kommunizieren, um den Ablauf des individuellen Stundenplanes zu verwirkliche. Hierfür wurde eine Rest Schnittstelle umgesetzt, über welche jegliche Kommunikation mit dem Service läuft. Diese Schnittstelle ist im Quellcode Ordner rest_api zu finden und enthält den Rest Schnittstellen Service an sich und alle definierten Endpunkte, über welche mit dem Service kommuniziert werden kann. Für die grundlegende Struktur der Schnittstelle wurde das frei nutzbare Framework Slim eingesetzt:[33] Das Framework bietet nach dessen Einbindung ein komplettes eigenständiges Projekt, welches eine einfache beispielhafte Rest Schnittstelle abbildet. Diese wurde dann auf die Bedürfnisse der geplanten ServiceSchnittstelle angepasst.

8.4.1 Anpassung der Abhängigkeiten

Die bereits vorhandenen Abhängigkeiten, welche in der Datei /rest_api/src/dependencies.php zu finden sind, worden um zwei weitere benötigte Abhängigkeiten erweitert.

```
1  $container['db'] = function ($c) {
       return new Db(DBHost, DBPort, DBName, DBUser, DBPassword);};
2  $container['encryption_class'] = function ($c) {
    return new Encryption();};
```

Listing 20 Abhängigkeit Datenbank und Verschlüsslung

Zeile eins erweitert das Objekt $container um die Funktion db, welche eine sichere Datenbankverbindung bereitstellt. Dagegen wird in Zeile zwei eine Klasse eingebunden, um Strings zu verschlüsseln.

[33] URL: http://www.slimframework.com/

8.4.2 Endpunkte

Während der Umsetzung dieses Projektes worden einige Schnittstellen Endpunkte eingerichtet, welche nach Belieben von der EAH Jena App oder anderen Interessierten genutzt werden können. Im Folgenden werden zwei Endpunkte der Schnittstelle erläutert, welche grundlegend notwendig sind, um die Anforderung des individuellen Stundenplanes in Verbindung mit Push Benachrichtigungen zu erfüllen.

8.4.3 Schnittstelle /changes

Die Schnittstelle /changes ermöglicht das Abonnieren von Modulen und Antwortet mit einer auf die Anfrage angepassten Änderungs-Historie, welche von der App genutzt werden kann, um ihre interne Stundenplan-Struktur zu aktualisieren.
Angefragt werden kann dieser End Point in folgender Form:

```
1  curl -X POST \
2    'https:// {DOMAIN}\rest_api\public/changes' \
3    -H 'Content-Type: application/json' \ // optional
4    -d '{BODY der Anfrage}'
```

Listing 21 Beispiel /change

Generell muss die HTTP Anfragemethoden POST verwendet werden, da dieser Endpunkt nur auf diese Methode reagiert. Wie in Zeile zwei zu sehen, muss die komplette URL bis zum Endpunkt genutzt werden. Alle Daten, welche an den Endpunkt übergeben werden sollen, müssen im JSON String Format im Body der Anfrage eingetragen sein. Diese JSON muss folgenden grundlegenden Aufbau einhalten:

```
1  {          "os_id":1,
2              "device_id":"DEVICE TOKEN",
3              "refresh_timestamp":99,
4              "module_list":[{
                   "module_title":"Data Min./S",   "set_title":"SPLUSECBB84"}]}
```

Listing 22 JSON Beispiel für einen Body - /changes

- **os_id**
  Ist das verwendete Betriebssystem des Smartphone. Die eins steht hierbei für Android.
- **device_id**
  Ist der Device Token des Smartphones. Dieser wird genutzt um das Device zu identifizieren und Push Benachrichtigungen an diesen zu senden.
- **refresh_timestamp**
  Ist eine übermittelte Unix Zeitangabe, um die zu erhaltende Änderungshistorie einzugrenzen
- **module_list**
  In dieser Variablen müssen listenartig alle abonnierten Module des Nutzers angegeben sein. Benötigt wird der Titel des Moduls und der Set Titel/ Set SPlus Schüssel.

Beim Empfangen einer solchen Anfrage prüft der Endpunkt den kompletten Anfragen-Body nach seinem Inhalt. Wenn hierbei fehlerhafte oder fehlende Informationen erkannt werden, wird die Anfrage nicht weiter bearbeitet und der anfragende Client erhält eine Fehlerantwort mit dem HTTP Statuscode 400.

Um die Sicherheit zu erhöhen, wird der empfangen Device Token des Nutzer PHP seitig verschlüsselt. Um diese Verschlüsslung im kompletten Projekt vorzunehmen, wurde eine PHP Klasse namens Encryption geschaffen.[34] Diese Klasse verschlüsselt und entschlüsselt die Device Token mit der Verschlüsselungsmethode aes-256-cbc und mithilfe von zwei selbst wählbaren Sicherheitsschlüsseln.

```
1  $this->encryption_class->encryptString({Device Token]);
2  $this->encryption_class->decryptString({Device Token]);
```

Listing 23 Verschlüsslung, Entschlüsselung

Zeile eins zeigt den Funktionsaufruf, um den Device Token zu verschlüsseln, und Zeile zwei stellt die Nutzung der Entschlüsselung dar.

Anschließend wird dieses Device in der Datenbank Tabelle hinzugefügt oder aktualisiert und dessen abonnierte Module in der Tabelle device_module eingetragen. Dieser Schritt ist nötig, um den bereits beschrieben Prozess des Sendens der Push Benachrichtigungen umzusetzen.

[34] Siehe Quellcode: /general_functions/encryption.php

Anschließend wird folgende SQL Anfrage an die Datenbank gestellt, um die angefragte individuelle Änderung-Historie zu erhalten:

```
1  SELECT changes.event_splus_key,
   changes.changes_reason,changes.changes_reason_text,changes.new_event_json,
   changes.change_date,changes.set_splus_key,
          module.module_splus_key
    FROM changes
    INNER JOIN
      (SELECT device_module.set_splus_key
       FROM device_module
       WHERE device_module.device_id = ? GROUP  BY
   device_module.set_splus_key) device_module ON device_module.set_splus_key
   = changes.set_splus_key
    INNER JOIN
      (SELECT module_events.module_event_splus_key,
              module_events.module_id
       FROM module_events) module_events ON changes.event_splus_key =
   module_events.module_event_splus_key
    INNER JOIN
      (SELECT module.id,
              module.module_splus_key
       FROM module) module ON module_events.module_id = module.id
    WHERE Unix_timestamp(changes.change_date) > ? GROUP  BY changes.id
      ORDER  BY changes.change_date ASC
```

Diese umfangreiche SQL Anfragen kombiniert durch mehrerer INNER JOINS mehrere Datenbank Tabellen miteinander. Dabei wird stets darauf geachtet, dass die Tabellen vor der Einbindung durch JOIN in sich selber vorselektiert werden, um die Last dieser Anfrage massiv zu verringern. Darüber hinaus werden nur diese Tabellen -Spalten abgefragt, welche unbedingt benötigt werden. Das Ergebnis dieser Anfrage sind alle Eventänderungen, welche für das anfragende Device relevant und für die App notwendig sind.

Dieses Datenbankergebnis wird im JSON Format an den anfragenden Client zurückgesendet.

Dieser Endpunkt ist ebenfalls mit der HTTP Anfragemethoden GET erreichbar. Diese kann genutzt werden, um einzig die aktuelle individuelle Änderung-Historie abzufragen, ohne eine Änderung an den abonnierten Modulen vorzunehmen.

8.4.4 Schnittstelle /delete

Ein weiterer kleiner aber wichtiger Endpunkt ist /delete. Dieser ermöglicht, nach Übergabe des Device Tokens, dieses Device und alle seine abonnierten Module aus der Datenbank zu entfernen. Somit bietet dieser Endpunkt eine Möglichkeit, ein Device komplett entfernen zu lassen.

8.5 Probleme und Bewertung

8.5.1 Stundenplanaufbau

Das Ziel eines datenverarbeitenden Service ist es strukturierte, zusammenhängende und klar identifizierbare Daten zu verarbeiten. Der Stundenplan-Aufbau der EAH Jena entpuppte sich währende Umsetzung des Projektes als komplexer wie in der Konzeption analysiert. Durch gewisse Eventhinzufügungen während des Semesters, seitens der EAH Jena, kann es vorkommen, dass zusammenhängende Events nicht mehr über ihren festen SPlus ID Schlüssel identifiziert werden konnten. Hierbei wurde sehr viel Zeit während der Umsetzung genutzt, um doch eine ID oder Schlüssel basierte Lösung zu finden. Eine zwar unzufriedene, aber nutzbare Lösung war es, keine Zusammenhänge über die Event Schlüssel herzustellen, sondern über einen String Vergleich der Event Titel.

8.5.2 Fehlende Stundenplan Daten

Das Bachelorprojekt und dessen Umsetzung fiel zeitlich an das Semesterende und in die Semesterferien der EAH Jena. Dies hatte zur Folge, dass am Ende des laufenden Semesters kaum bis keine Änderungen an den Stundenplan-Daten der EAH Jena aufgetreten sind. Darüber hinaus wurde vor Umsetzungsstart nicht bedacht, dass der Stundenplan und all seine Daten in den Semesterferien offline genommen werden.

Dieses Problem konnte gelöst werden, indem vor Abschaltung des Stundenplan-Servers ein komplettes Back-up aller Daten angefertigt worden ist. Diese wurden auf einen über das Internet verfügbaren FTP Server geladen, um das Abfragen der Stundenplandaten über das Internet zu simulieren.

8.5.3 Webserver

Es stand während der Umsetzung des Service nicht das spätere Produktivsystem zur Verfügung, da dessen Einrichtung erst nach der Umsetzung des Service und der vollständigen Implementierung des Service in die App geplant war.

Die Lösung dieses Problem war das selbstständige Aufsetzen und Einrichten eines lokalen Webservers, welcher in seiner Architektur und Art dem des späteren Webservers entsprach.

8.6 Testen des Service

Für umfangreiche Programmierprojekte ist es stets nötig, eine gewisse Teststrategie zu besitzen und einzuhalten. Des Weiteren sollte nach jedem Umsetzungsschritt ein Funktionstest durchgeführt werden, um auftreten Fehler schnellstmöglich lokalisieren zu können. Darüber hinaus sollte stets überprüft werden, ob der umgesetzte Arbeitsprozess, z. B. die Fertigstellung einer neuen Funktion, den vorher gesetzten Anforderungen entspricht und allen Erwartungen erfüllt.

Um jederzeit während der Umsetzung umfängliche Tests durchzuführen zu können, wurde ein privater Webserver dem späteren Produktivsystem nachempfunden. Der lokale Entwicklungsserver konnte hierfür nicht genutzt werden, da hierbei das Anfragen eines Service über das Internet, nicht getestet werden konnten.

Als anfragender Client wurde das Programm Postman[35] verwendet. Dies erleichtert das Erstellen und Senden von Testanfragen an den Service. Des Weiteren stellt Postman alle vorhanden Informationen wie Antwort, Header Informationen, Antwortzeit und weitere Informationen, der Anfrage bereit. Andere anfragende Client mussten nicht genutzt werden, da die Schnittstelle das Service der Rest Definition entspricht und somit jeder Client gleichwertig betrachtet wird.

Um Push Benachrichtigungen zu testen wurde auf die Firebase CLI [36] zurückgegriffen. Diese CLI ermöglicht es, einen Service zu starten, welche Firebase Push Benachrichtigungen empfangen kann.

[35] URL: https://www.getpostman.com/

[36] URL: https://firebase.google.com/docs/cli

Die Durchführung aller Tests wurden vom Entwickler des Service selbst übernommen. Dabei lassen sich die ausgeführten Tests in drei Kategorien einteilen.

1. Testen des internen Prozesses, welcher die Stundenplan Daten in seiner Datenbank aktualisiert
2. Testen der Rest Schnittstelle
3. Testen der Push Benachrichtigungen

Vor Abgabe dieses Projektes wurde der Service selbst umfangreich getestet und konnte jegliches Szenario abbilden, welches in den Anforderungen definiert ist.

Getestet werden konnte nicht das Zusammenspiel zwischen geplanter EAH Jena App und dem Service auf einem Produktivsystem.

Darüber hinaus konnte der finale Service nicht mit live Stundenplan-Daten und Änderungen getestet werden, da diese im Zeitraum des Testens nicht zur Verfügung standen.

8.7 Bewertung der Umsetzung

Im Großen und Ganzen kann ausgesagt werden, dass der umgesetzte Service den geforderten Anforderungen entspricht und alle geplanten Szenarien funktionstüchtig abbilden kann. Es wurde, wie in den Anforderungen festgehalten, ein Service umgesetzt, welcher die Programmiersprache PHP nutzt, um zum späteren Zeitpunkt von Studenten der EAH Jena, auch im Rahmen von Projekten, gepflegt und weiterentwickelt werden kann. Es wurde eine gute strukturierte Datenbank erschaffen, welche alle Daten übersichtlich und sinnvoll aufnehmen und abbilden kann. Des Weiteren wurde ein Prozess entwickelt, um Stundenplan-Änderung zuverlässig erkennen und auswerten zu können und um Push Benachrichtigungen an die jeweiligen Nutzer der App zu senden. Darüber hinaus wurde eine standardisierte Schnittstelle geschaffen, um eine Kommunikation zwischen App und Service zu ermöglichen.

Die Implementierung des umgesetzten Service in ein Produktivsystem konnte leider im Rahmen des Projektzeitraums nicht umgesetzt werden, da ein Abschlusstest in Kombination mit der neuen EAH Jena App und Live-Daten des EAH Jena Stundenplan-System noch nicht durchgeführt werden konnte.

9 Ausblick und Fazit

9.1 Ausblick

Mit Abschluss und Abgabe dieses Bachelorprojektes ist die Entwicklung des serverseitigen Services, welcher die EAH Jena App um individuelle Stundenpläne und Push Benachrichtigungen erweitert, keineswegs vollumfänglich beendet. Erst mit der noch geplanten Implementierung in die EAH Jena App und dem Einrichten dieses Services auf einem Produktivsystem wird ein gewisser Projektstatus erreicht.

Anschließend ist geplanten den umgesetzten Service für andere Hochschulen zur Verfügung zu stellen, damit diese ihre eventuell bereits vorhandenen Apps um Push Benachrichtigungen erweitern können.

Darüber hinaus ist die Weiterentwicklung der neuen EAH Jena App noch voll in Gange und soll neben der Erweiterung um individuelle Stundenpläne auch eine generelle Überarbeitung erhalten.

Des Weiteren ist aktuell an der EAH Jena geplant einen standardisierten Nachrichtenkanal für alle Fachbereiche zu planen und umzusetzen. Über diese Nachrichtenkanäle sollen die einzelnen Fachbereiche ihre aufkommenden Nachrichten und Informationen verteilen. Diese Nachrichtenkanäle könnten im Anschluss in die EAH Jena App und dem Service integriert werden. Es wäre denkbar, dass der Nutzer der App sich einen individuellen Nachrichtenfeed zusammenstellen könnten und via Push Benachrichtigung über neue Nachrichten oder Informationen informiert wird.

Auch wäre es denkbar die App in Kombination mit dem serverseitigen Service, um einen gewissen Nachrichten Chat für Module zu erweitern. Hierbei könnten interessierte Studenten sich über das jeweilige Modul austauschen, Probleme lösen oder Fragen zur nächsten Klausur klären.

Diese und viele weitere Konzepte und Funktionen sind denkbar, um die App in ihrem Funktionsumfang zu erweitern. Es ist an dieser Stelle nicht abschätzbar, welche dieser Ideen den Weg in die App oder den Service finden.

9.2 Fazit

Das Ziel dieser Bachelorarbeit war eine Konzeption und Umsetzung eines Service, welcher es ermöglichen sollte, dass Nutzer der EAH Jena App sich einen individuellen Stundenplan zusammenstellen können und über Änderungen dieses Planes via Push Benachrichtigung informiert werden. Darüber hinaus sollte die App bei der Aktualität der individuellen Stundenpläne unterstützt werden. Mit Abschluss dieses Projektes wurden alle definierten Anforderungen und Kernziele erfüllt und eine Grundlage geschaffen die Entwicklung des Service und der EAH Jena App voranzutreiben. Des Weiteren wurden alle Sekundärziele, besonders die datenschutzorientierten Sekundärziele, in der Planung ausgearbeitet und während der Entwicklung des Service bedacht und eingebunden. Allumfassend kann ausgesagt werden, dass dieses Projekt, mit all seinen Eigenheiten während der Umsetzung, als gelungen angesehen werden kann, da das Senden von individuellen Push Benachrichtigungen vollständig umgesetzt worden ist.

Die Ausarbeitung und Umsetzung dieses Projektes hat mir persönlich viel Spaß gemacht und ich konnte meine Kenntnisse im Bereich Webentwicklung weiter vertiefen. Im ganzen Projektzeitraum konnte ich dabei mein schon vorhandenes Wissen einbringen und anwenden. Meinen Kenntnisstand konnte hierbei dennoch um viele aktuelle und interessante Themen erweitert werden. Besonders die Wissensbereiche Datenschutz und Push Benachrichtigungen für Smartphones waren sehr informativ.

Insgesamt bin ich mit dem Gesamtergebnis und meiner erbrachten Leistung zufrieden. Das gewonnen Wissen und die neuen Erfahrungen werden mir sicher im Laufe meiner weiteren Studien- und Arbeitszeit vom Vorteil sein. Eine spätere Beschäftigung im Bereich der Webentwicklung kann ich mir sehr gut vorstellen.

Literaturverzeichnis

(vectorsoft), F., 2012. *HTTP – Kommunikation im Web.* [Online]
Available at: https://www.vectorsoft.de/blog/2012/09/http-kommunikation-im-web/
[Zugriff am 20 09 2019].

Cloudflare, 2019. *HTTP vs. HTTPS: What are the differences?.* [Online]
Available at: https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/
[Zugriff am 05 09 2019].

Docker Inc, 2019. *Docker overview.* [Online]
Available at: https://docs.docker.com/engine/docker-overview/
[Zugriff am 29 08 2019].

ECMA, 2013. *The JSON Data Interchange Format Standard.* [Online]
Available at: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf
[Zugriff am 09 08 2019].

Esser, C. K. /. S., 2008. *PHP-Sicherheit - PHP/MySQL-Webanwendungen sicher programmieren.* 3., überarbeitete Auflage Hrsg. s.l.:dpunkt.verlag.

Fielding, R. et al., 1999. *Hypertext Transfer Protocol -- HTTP/1.1.* [Online]
Available at: https://www.ietf.org/rfc/rfc2616.txt
[Zugriff am 29 07 2019].

Fielding, R. T., 2000. *https://www.ics.uci.edu.* [Online]
Available at: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf
[Zugriff am 20 08 2019].

Google, 2019. *Add Firebase to your Android project.* [Online]
Available at: https://firebase.google.com/docs/android/setup
[Zugriff am 15 09 2019].

Haas, M., 2018. *bitkom.org.* [Online]
Available at: https://www.bitkom.org/sites/default/files/file/import/Bitkom-Pressekonferenz-Smartphone-Markt-22-02-2018-Praesentation-final.pdf

Handelskammer Hamburg, 2019. *Die Datenschutzgrundverordnung (DS-GVO).* [Online]
Available at: https://www.hk24.de/produktmarken/beratung-service/recht_und_steuern/wirtschaftsrecht/medien_it_recht/datenschutzgrundverordnung/3740520?index=12&printsrc=button
[Zugriff am 02 08 2019].

Johner, P. D. C., 2013. *Funktionale Anforderungen versus nicht-funktionale Anforderungen.* [Online]
Available at: https://www.johner-institut.de/blog/iec-62304-medizinische-software/funktionale-und-nicht-funktionale-anforderungen/
[Zugriff am 01 08 2019].

Luber, S., 2019. *Was bedeutet LAMP?.* [Online]
Available at: https://www.cloudcomputing-insider.de/was-bedeutet-lamp-a-808284/
[Zugriff am 30 07 2019].

MariaDB, 2019. *AES_ENCRYPT.* [Online]
Available at: https://mariadb.com/kb/en/library/aes_encrypt/
[Zugriff am 04 08 2019].

Meraneos, C., 2017. *DATENSCHUTZ-GRUNDVERORDNUNGDS-GVO.* [Online]
Available at: https://www.stiftungswoche.online/der-inhalt/uploads/2018/07/OSW_Datenschutzgrundverordnung_20180927.pdf
[Zugriff am 10 08 2019].

Oracle Corporation, 2019. *INSERT Syntax.* [Online]
Available at: https://dev.mysql.com/doc/refman/8.0/en/insert.html
[Zugriff am 03 09 2019].

Prof. Dr. Sibylle Gierschmann, L. (. U. F. f. U.-. u. M. D. (., 2018. *Kommentar Datenschutz-Grundverordnung.* s.l.:s.n.

Sangmeister, B., 2016. *smitscon.de/.* [Online]
Available at: http://www.smitscon.de/ics-dateien-erstellen/
[Zugriff am 20 08 2019].

Sharma, P., 2005. *cert-in..* [Online]
Available at: https://www.cert-in.org.in/Downloader?pageid=7&type=2&fileName=ciwp-2005-06.pdf
[Zugriff am 10 08 2019].

Singh, A., 2019. *The Ultimate Guide to Push Notifications.* [Online]
Available at: https://webengage.com/blog/what-is-push-notification/
[Zugriff am 03 09 2019].

Stenberg, D., 2017. *The cURL project.* [Online]
Available at: https://plennuscursos.com.br/storage/educationalmaterials/__acd372841289b14dade72301f2b57ba64c8506ed__/everything-curl.pdf
[Zugriff am 01 09 2019].

Terrorismusbekämpfung, B. f. V. u., 2017. *Distributed Denial of Service.* [Online]
Available at: https://www.bvt.gv.at/401/files/CSCSchriftenreihe-DDoS_HintergruendepraeventiveMassnahmenundMitigationsmassnahmen_final_2017-12.pdf
[Zugriff am 20 08 2019].

Union, E., 2019. *DSGVO.* [Online]
Available at: https://dsgvo-gesetz.de/
[Zugriff am 27 07 2019].

Wolf, S. T. /. M. E. /. S. S. /. O., 2015. *https://www.dpunkt.de.* [Online]
Available at: https://www.dpunkt.de/common/leseproben//11629/4_Einfuehrung%20in%20REST.pdf
[Zugriff am 10 09 2019].

Wichtiger Hinweis für alle online verfügbaren Quellen:

Die komplette PDF oder eine Kopie der HTML Seite befindet sich im CD-Anhang.

Anlagen

Beiliegendes Medium enthält:

- Bachelorarbeit im PDF Format
- Nachweis aller Onlinequellen
- Quellcode des umgesetzten Projektes

Ehrenwörtliche Erklärung und Einverständniserklärung

Ich versichere, dass ich die vorliegende Arbeit selbständig und ohne unerlaubte Hilfe Dritter verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Stellen, die inhaltlich oder wörtlich aus Veröffentlichungen stammen, sind kenntlich gemacht. Diese Arbeit lag in der gleichen oder ähnlichen Weise noch keiner Prüfungsbehörde vor und wurde bisher noch nicht veröffentlicht.

Hiermit erkläre ich mich mit der Einsichtnahme in meine Abschlussarbeit im Archiv der Bibliothek der EAH Jena ~~einverstanden~~ / nicht einverstanden (Unrichtiges bitte streichen).

Ort, Datum Unterschrift