



## Diplomarbeit

Höhere Technische Bundeslehranstalt Leonding  
Informationstechnologie mit Schwerpunkt Medientechnik

# Footnote

Footnote - Die AR-Notizapp

Eingereicht von: **Isabella Hundstorfer, 5AHTIM**

**Christopher Gusenbauer, 5AHITM**

Datum: **5. April 2019**

Betreuer: **Prof. Franz Rager, BSc MA**

## **Declaration of Academic Honesty**

Hereby, I declare that I have composed the presented paper independently on my own and without any other resources than the ones indicated. All thoughts taken directly or indirectly from external sources are properly denoted as such.

This paper has neither been previously submitted to another authority nor has it been published yet.

Leonding, 5. April 2019

Isabella Hundstorfer, Christopher Gusenbauer

## **Eidesstattliche Erklärung**

Hiermit erkläre ich an Eides statt, dass ich die vorgelegte Diplomarbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Gedanken, die aus fremden Quellen direkt oder indirekt übernommen wurden, sind als solche gekennzeichnet.

Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Leonding, am 5. April 2019

Isabella Hundstorfer, Christopher Gusenbauer

## Zusammenfassung

Augmented Reality, Neurale Netzwerke, Smart Devices, Smart Home, Industrie 4.0 und Internet of Things sind Themenfelder, die zurzeit enorm in Bewegung sind. Unternehmen weltweit streben die Integration von Objekterkennung in ihre Produkte an. Dennoch ist es für Entwickler noch schwierig derartige Technologien flexibel einzusetzen, weil die Verwendung der Frameworks häufig nicht intuitiv ist und diese teils schlecht dokumentiert sind. Es existieren verschiedene Einschränkungen, was die Speicherdauer bzw. Funktionsweise und Offenheit der Frameworks betrifft. Des Weiteren sind beinahe alle am Markt existierenden Lösungen nicht um eigene Objekte erweiterbar, sie erkennen lediglich vordefinierte Gegenstände. Das Ziel von Footnote war es, technisch nicht versierten Anwendern die Möglichkeit bereitzustellen, selbst Informationen zu realen Objekten zu speichern.

Ein erstes Teilziel war es, Industriemaschinen zu erkennen und Informationen wie Notizen oder Bedienungsanleitungen mit ihnen digital zu verknüpfen. Weiters können Benutzer selbst neue reale Objekte hinzufügen, wobei der Fokus darauf gelegt wurde, dass hierfür keinerlei Erfahrung in der Softwareentwicklung nötig ist. Zusätzlich dazu wird Text, falls einer auf den Gegenständen sichtbar ist, erkannt und abgespeichert. Die Anwendung wurde in Form einer selbst hostbaren Client-Server Architektur realisiert.

Für die Objekterkennung wurde aufgrund einer besseren Erkennungsqualität und guter Kompatibilität mit Client/Server Strukturen das „Vuforia“ Framework gewählt, da alternative Frameworks diese nicht im gleichen Ausmaß aufwiesen. Der Client basiert zur plattformübergreifenden Programmierung auf „Ionic“, die Objekterkennung mit Vuforia wurde nativ in „Android“ programmiert. Serverseitig wurde aufgrund einer Vielzahl an gut benutzbaren Frameworks und verlässlicher Qualität in „Java Enterprise“ entwickelt. Für die Texterkennung wird die Engine „Tesseract“ benutzt. In der praktischen Anwendung ist der Client ähnlich einer Notizapp wie beispielsweise „Evernote“ oder „Onenote“ mit oben besprochener Funktionalität.

Der Benutzer kann nun mit dem fertigen Produkt - der „Footnote App“ - und seinem mobilen Endgerät reale Objekte, wie z.B. Gemälde, Anlagenteile, Geräte, Sehenswürdigkeiten und vieles mehr fotografieren und zu diesen realen Dingen digitale Informationen, wie Künstlerdaten, Entstehungsgeschichte, Wartungs- und Schaltpläne, Bedienungsanleitungen, usw. abspeichern. Diese können in Form von Text, Bildern oder Dateien gespeichert werden. Die App kann diese dann mithilfe der Kamera des Geräts wiedererkennen und oben erwähnte Informationen virtuell anzeigen. Das Erkennen von Gegenständen der Realität wird bereits in vielen Programmen umgesetzt, Footnote geht aber einen Schritt weiter und ermöglicht es dem Benutzer, selbst Objekte anzulegen, die vom System wiedererkannt werden sollen.

## **Abstract**

Augmented Reality, Neural Networks, Smart Devices, Smart Home, Industry 4.0 and the Internet of Things are a collection of topics that are currently widely discussed. Companies worldwide aim to integrate object recognition into their products. However, some of these technologies are quite difficult to use for developers, because dealing with frameworks is not very intuitive and they are sometimes lacklusterly documented. There are several restrictions in terms of how long objects can be saved, or how open these frameworks are. Furthermore, most of the existing solutions are not extensible by self-made objects, they only detect predefined items. The goal of Footnote was to provide a possibility for technically inept users to save and relate information to real objects (things).

A first subgoal for the app was recognizing industrial machines and linking them together with digital information like notes or user manuals. Users can also create new real objects themselves – here the focus was put on minimizing a need for technical experience. Additionally, if there is text visible on items, it gets detected and saved.

The application was realized in a Client-Server architecture. The “Vuforia” Framework was used for object recognition, because of superior detection quality and good compatibility with Client-Server architecture, which alternatives did not provide to the same extent. The client is based on “Ionic” for cross-platform programming, the object recognition with Vuforia was programmed natively in “Android”. Java Enterprise was used server-sided, because it is a reliable language with a large array of libraries. Furthermore, “Tesseract” is used for text-detection. In reality, the client is similar to a note-taking app like “Evernote” or “Onenote” with the additional functionality discussed above.

With the finished product, the Footnote App, the User can take pictures of real objects like paintings, landmarks, industrial components, devices and more and link these to digital information like the artist, their creation history, user manuals, maintenance plans or circuit diagrams. These can be displayed in the shape of text, pictures or files. The app can then detect objects with the help of the device’s camera and display the previously mentioned information. The recognition of objects is already used in lots of currently running systems around the world, but Footnote takes one step further and gives every user the chance to link objects of the real world to digital information that can be recognized again.

## **Genderklausel**

Aus Gründen der Lesbarkeit und eines einfacheren Schreibprozesses wird in dieser Diplomarbeit darauf verzichtet, geschlechtsspezifische Formulierungen zu verwenden. So weit personenbezogene Bezeichnungen nur in männlicher Form angeführt sind, beziehen sie sich auf Männer und Frauen in gleicher Weise.

## **Danksagung**

An dieser Stelle möchten wir uns bei all denjenigen bedanken, die uns während der Auffertigung dieser Diplomarbeit unterstützt und motiviert haben. Zuerst gebührt unser Dank Herrn Professor Rager, der die Diplomarbeit betreut und begutachtet hat. Für die hilfreichen Anregungen und die konstruktive Kritik bei der Erstellung dieser Arbeit möchten wir uns recht herzlich bedanken. Auch bei allen anderen Professoren, die uns immer wieder bei Programmierfragen zur Seite gestanden sind, möchten wir uns bedanken.

Außerdem möchte wir uns bei Alexander Steinmaurer für das Korrekturlesen unserer Diplomarbeit bedanken. Genauso möchten wir uns bei unseren Klassenkollegen und Freunden bedanken, welche uns fachlich und mit aufmunternden Worten eine Hilfe waren. Abschließend möchten wir uns bei unseren Eltern bedanken, die immer ein offenes Ohr für uns hatten und eine große Stütze waren.

# Inhaltsverzeichnis

<b>1 Einleitung</b>	<b>6</b>
1.1 Aufgabenstellung und Zielsetzung . . . . .	6
1.2 Aufgabenverteilung . . . . .	6
1.3 Marktanalyse existierender Lösungen . . . . .	7
1.3.1 Evernote . . . . .	7
1.3.2 Onenote . . . . .	7
1.3.3 Turtl . . . . .	8
1.3.4 Standard Notes . . . . .	8
1.4 Abgrenzungskriterien . . . . .	9
1.5 Grundbegriffe . . . . .	9
1.6 Autoren der Diplomarbeit . . . . .	10
1.6.1 Isabella Hundstorfer . . . . .	11
1.6.2 Christopher Gusenbauer . . . . .	12
<b>2 Pflichtenheft</b>	<b>13</b>
2.1 Zielbestimmungen . . . . .	13
2.1.1 Musskriterien . . . . .	13
2.1.2 Wunschkriterien . . . . .	13
2.1.3 Abgrenzungskriterien . . . . .	13
2.2 Produkteinsatz . . . . .	14
2.2.1 Anwendungsbereiche . . . . .	14
2.2.2 Zielgruppen . . . . .	14
2.3 Produktumgebung . . . . .	14
2.3.1 benötigte Software . . . . .	14
2.3.2 benötigte Hardware . . . . .	14
2.3.3 Qualitätszielbestimmungen . . . . .	15
2.4 Entwicklungsumgebung . . . . .	15
2.4.1 Software . . . . .	15
2.4.2 Hardware . . . . .	15
<b>3 Augmented Reality</b>	<b>16</b>
3.1 Definition . . . . .	16
3.2 Verwendungsbereiche . . . . .	16

3.2.1	Medizin . . . . .	16
3.2.2	Industrielle Anwendungen . . . . .	17
3.2.3	Verknüpfung von Informationen mit Objekten . . . . .	17
3.2.4	Unterhaltung . . . . .	17
3.2.5	Militär . . . . .	18
3.2.6	Architektur . . . . .	18
3.2.7	Werbung . . . . .	18
3.2.8	Bildung . . . . .	18
3.3	Technische Voraussetzungen . . . . .	19
3.4	Geschichte und Entwicklung . . . . .	19
3.5	Funktionsweise . . . . .	19
3.5.1	Marker-Based Augmented Reality . . . . .	19
3.5.2	Markerless Augmented Reality . . . . .	20
3.5.3	Projection-based Augmented Reality . . . . .	20
3.5.4	Superimposition-based Augmented Reality . . . . .	21
<b>4</b>	<b>AR-Frameworks</b>	<b>22</b>
4.1	Frameworkübersicht . . . . .	22
4.2	Vuforia . . . . .	23
4.2.1	Vorteile . . . . .	23
4.2.2	Nachteile . . . . .	24
4.3	Wikitude . . . . .	25
4.3.1	Vorteile . . . . .	25
4.3.2	Nachteile . . . . .	25
<b>5</b>	<b>Client</b>	<b>27</b>
5.1	Überblick . . . . .	27
5.2	Desktop-Client . . . . .	27
5.2.1	Login . . . . .	27
5.2.2	Registrieren . . . . .	28
5.2.3	Home . . . . .	29
5.2.4	Objekt erstellen . . . . .	30
5.2.5	Benutzer-Gruppen . . . . .	30
5.2.6	Logout . . . . .	31
5.3	Android-Client . . . . .	32
5.3.1	Login . . . . .	32
5.3.2	Registrieren . . . . .	33
5.3.3	Home . . . . .	33
5.3.4	Benutzer-Gruppen . . . . .	34
5.3.5	Umgebungsansicht . . . . .	36
5.3.6	Vuforia . . . . .	36
5.3.7	Logout . . . . .	37
5.4	Ionic . . . . .	38
5.4.1	Alternativen zu Ionic . . . . .	38

5.4.2	Generelle Informationen zu Ionic . . . . .	40
5.5	Verwendung von Ionic . . . . .	42
5.5.1	Ionic installieren CLI . . . . .	42
5.5.2	Erstellen eines eigenen Icons und Splash-Screens . . . . .	43
5.5.3	HTTP-Provider . . . . .	43
5.5.4	Standort erkennen . . . . .	43
5.5.5	Dateiupload . . . . .	44
5.5.6	Bilder vom Server anzeigen . . . . .	45
5.5.7	WYSIWYG-Editor . . . . .	45
5.5.8	App-Launcher . . . . .	47
5.6	Android . . . . .	48
5.6.1	Android-Studio . . . . .	48
5.6.2	Remote Debuggen-Android . . . . .	49
5.6.3	Zugriff auf lokalen Server mit dem Android-Smartphone . . . . .	50
5.7	Verwendung von Wikitude . . . . .	52
5.7.1	Lizenz . . . . .	52
5.7.2	JWT . . . . .	52
5.8	Verwendung von Vuforia . . . . .	53
5.8.1	Einrichten der Android-Entwicklungsumgebung . . . . .	53
5.8.2	Erkennen eigener Objekte . . . . .	53
5.8.3	JWT . . . . .	54
<b>6</b>	<b>Server</b> . . . . .	<b>55</b>
6.1	Anforderungen und deren Umsetzung . . . . .	55
6.2	Das Java EE Projekt . . . . .	56
6.2.1	Entitäten . . . . .	56
6.2.2	Services und Repositories . . . . .	57
6.2.3	Sicherheit . . . . .	57
6.2.4	Kommunikation mit Vuforia . . . . .	57
6.2.5	Sonstige Logik . . . . .	59
6.3	Systemumgebung . . . . .	60
6.3.1	Java . . . . .	60
6.3.2	Apache NetBeans . . . . .	62
6.4	Application Server im Vergleich . . . . .	62
6.4.1	Wildfly/Jboss EAP . . . . .	63
6.4.2	Glassfish . . . . .	64
6.4.3	Tomcat . . . . .	65
6.4.4	Payara Server . . . . .	65
6.4.5	Rechercheergebnisse . . . . .	66
6.5	JW-Tokens . . . . .	66
6.5.1	Was sind JW-Tokens . . . . .	66
6.5.2	Anwendungsfälle von JW-Tokens . . . . .	66
6.5.3	Strukturierung . . . . .	67

6.5.4	Praxisbeispiel Tokengenerierung Java . . . . .	67
6.6	Tesseract . . . . .	68
6.6.1	Tess4J . . . . .	69
6.7	Logging . . . . .	69
6.8	Fehlerbehandlung . . . . .	70
6.8.1	Exception Mapper . . . . .	70
6.8.2	Jax-RS WebapplicationException . . . . .	70
6.9	Authentifizierung, Autorisierung & Benutzer . . . . .	70
6.9.1	Passwort-Hashing . . . . .	70
6.9.2	Registrierungsprozess . . . . .	72
6.9.3	ContainerRequestFilter, ContainerResponsefilter . . . . .	73
6.9.4	Berechtigungen . . . . .	74
6.10	FileHandlerServlet . . . . .	76
6.10.1	FileUpload . . . . .	76
6.10.2	FileDownload . . . . .	78
6.11	Vuforia Codebeispiele . . . . .	79
6.11.1	Target hochladen . . . . .	79
6.11.2	Target löschen . . . . .	80
6.12	OpenAPI(Swagger) . . . . .	81
6.13	Designphilosophie am Server . . . . .	83
6.14	Ubuntu: Wildfly aufsetzen und Datenbankverbindung hinzufügen . . . . .	84
6.14.1	Voraussetzungen: . . . . .	84
6.14.2	Wildfly installieren . . . . .	85
6.14.3	Remotezugang . . . . .	87
6.14.4	Möglichkeit 1: NGINX Reverse Proxy . . . . .	88
6.14.5	Möglichkeit 2: Config ändern . . . . .	89
6.14.6	Datenbankverbindung konfigurieren . . . . .	89
6.14.7	Deployment . . . . .	91
<b>7</b>	<b>Logo, Webseite, Plakat</b>	<b>93</b>
7.1	Projektwebsite . . . . .	93
7.1.1	Motivation . . . . .	93
7.1.2	Umsetzung . . . . .	94
7.2	Logo . . . . .	95
7.3	Plakat . . . . .	96

# Kapitel 1

## Einleitung

### 1.1 Aufgabenstellung und Zielsetzung

Die Maschinenbaufirma Kremsmüller hat das Problem, dass Informationen über Ihre Maschinen meist nicht gesammelt und digital vor Ort (also direkt bei der jeweiligen Maschine) vorliegen. Die Maschinen sollen via AR erkannt werden und Informationen, wie z.B. Betriebsanleitungen, Parameterlisten, Wartungstermine, Anlagen- und Schaltpläne sollten direkt am Display des Smartphones eingeblendet werden.

Das Ziel des Projektes ist es eine App zu erstellen, die Notizen zu Objekten des echten Lebens mithilfe von erweiterter Realität direkt zuordnet und vor Ort abrufbar macht. Die App soll Dokumente, Termine, Notizen und Bilder zu echten Objekten zuordnen und diese Infos mit anderen teilen können. Mit Infos versehene Objekte, die in der Nähe sind, werden von der App auffindbar gemacht. Bsp: Man scannt einen Drucker, die App bietet jetzt ein manual.pdf, eine Notiz zum Patronentauschen und Wartungstermine für diesen Drucker an.

### 1.2 Aufgabenverteilung

Isabella Hundstorfer ist zuständig für den Client, das Auswählen eines geeigneten Augmented Reality Frameworks, die Datenvisualisierung, das Aufrufen des „Vuforia“ Frameworks zum Anzeigen bzw. Wiedererkennen von Objekten und das Schaffen von Schnittstellen zwischen „Ionic“, „Android“, „Vuforia“ und „Wikitude“ für eben jenen Zweck, des Weiteren übt Sie die Rolle der Projektleiterin aus.

Christopher Gusbauer ist für den Server, die Server-Client Kommunikation, die Datenbank, die Infrastruktur(„Production Environment“, Auswählen eines Application Servers) und Entwicklungsumgebung, die Sicherheit, Rollen und Benutzergruppen und die Kommunikation mit „Vuforia“ was das Erstellen, Hochladen und Löschen von Targets (Objekten) betrifft, sowie OCR (Text in Bildern erkennen & auslesen) und den Großteil der Projektwebsite verantwortlich.

## 1.3 Marktanalyse existierender Lösungen

### 1.3.1 Evernote



Abbildung 1.1: Logo Evernote

Der Platzhirsch Evernote dominiert den Markt der tagebuch-, bzw. notizähnlichen Informationsspeicherung. Sämtliche Standardfeatures sind verfügbar - Notizen sind in unterschiedlichsten Formen verwendbar: Text, Bilder, Audio und Videoclips. Diese werden in sogenannten Notizbüchern gespeichert - Hauptorganisationshilfe sind aber Tags, die Notizen mit beschreibenden Stichwörtern beschreiben und so sehr gut durchsuchbar machen. Checklisten, Texterkennung in Bildern & Browser-Addons, mit denen man Websites speichern kann, sowie ein „PDF-Editor“ und plattformübergreifende Synchronisation von Anwendungsdaten machen die App zum Marktführer.

Evernote hat aber auch markante Nachteile:

- Features, die über die Standardfunktionalitäten einer Notizapp hinausgehen, benötigen das Abo der relativ teuren Pro-Version des Services.
- Evernote kann „mitlesen“, was User schreiben.
- Es ist nicht möglich die App selbst zu hosten.
- Kommunikation mit den Entwicklern ist schwierig, eigene Modifizierung der Software ist für Firmen nicht möglich.

[6]

### 1.3.2 Onenote

Microsofts Antwort auf Evernote, Onenote, ist sehr gut in das Microsoft Universum und die Office-Suite integriert. Onenote lässt den Benutzer den Arbeitsbereich frei beschreiben, ebenfalls ist ein sehr guter „Stylus Support“, d. h. die Möglichkeit mit einem Stift z.B. auf einem Tablet zu schreiben, enthalten. Die Funktionen und Nachteile von Onenote decken sich stark mit denen von Evernote, wobei Evernote manche Funktionen im Repertoire hat, die Onenote nicht hat. Beispielsweise sind Onenotes Tags deutlich schlechter



Abbildung 1.2: Logo Onenote

umgesetzt (bis vor kurzem keine Möglichkeit eigene Tags zu erstellen, funktionale Unterschiede zw. Onlineversion und Desktopanwendung, Suchfunktion findet unbeabsichtigte Ergebnisse). Onenote ist allerdings kostenlos. [23]

### 1.3.3 Turtl



Abbildung 1.3: Logo Turtl

Turtl ist die offene Alternative zu den Apps der Giganten. Das heißt der Quellcode ist einsehbar und verschiedenste Entwickler können ihre Verbesserungsvorschläge in die Entwicklung mit einbringen. In Markdown kann man Notizen verfassen, die sehr sicher verschlüsselt und auch für die Entwickler des Projekts nicht sichtbar sind. Es gibt ein gutes Taggingsystem, des Weiteren kann man Turtl auf einem eigenen Server abgeschottet hosten.

Nachteil: Abstraktere Luxusfeatures wie einen PDF Viewer oder „Optical Character Recognition“ hat Turtl nicht, Evernote und Onenote z.B. schon. [28]

### 1.3.4 Standard Notes

Simpel, aber sicher: Standard Notes ist via AES-256 „End to end“ verschlüsselt. Dafür ist die Standardversion relativ simpel aufgebaut, hauptsächlich wird einfacher Text synchronisiert. In einer kostenpflichtigen Version sind aber „Rich Text“, „LaTeX“, „Markdown“ & „HTML“ Editoren verfügbar. [27]



Abbildung 1.4: Logo Standard Notes

## 1.4 Abgrenzungskriterien

Traditionelle Notiz- und Informationsspeicherungsapps funktionieren alle nach demselben Schema. Notizen, egal welcher Art werden in virtuelle Notizbücher eingeordnet. Diese bestehen meistens aus Text, Bildern, Videos und Checklisten, oft ist auch das Anhängen von Dokumenten möglich.

Doch sie bleiben alle in der digitalen Welt, sie knüpfen nicht an unsere Realität an. Das ändert Footnote. Wenn ein Mensch sich bewegt, sollen sich die Inhalte, die Footnote abspeichert hat, an seine Welt anpassen. Gegenstände oder Gebäude, an denen er vorbeigeht, sowie reale Objekte die er mit seiner Handykamera einscannt, zeigen ihm ihre versteckten Informationen - ganz dynamisch.

Die meisten verfügbaren Lösungen bieten die folgenden drei Eigenschaften nicht kombiniert: Sicherheit, Offenheit und Benutzerfreundlichkeit. Footnote kann selbst heruntergeladen werden und mit leichten Einschränkungen (Vuforia-Lizenz und Wikitude-Lizenz) selbst gehostet werden, der Quellcode ist offen & eine leichte Bedienbarkeit mit WYSIWYG Editor & Komfortfeatures sind vorhanden.

## 1.5 Grundbegriffe

**Annotation** - Programmiersprachenelement, dass die Einbindung von Metadaten in den Quelltext erlaubt

**API** - eine Programmierschnittstelle

**AR** - erweiterte Realität

**Base64** - Verfahren zur Kodierung von 8-Bit-Binärdaten in eine Zeichenfolge, die nur aus lesbaren ASCII-Zeichen besteht

**CORS** - Mechanismus, der Webbrowsersn oder auch anderen Webclients Cross-Origin-Requests ermöglicht

**CSS** - Cascading Style Sheets

**Exception** - ein Fehler in einem Programm

**Footnote** - Titel der Diplomarbeit und des Projekts

**Framework** - umfasst Bibliotheken, Komponenten und Laufzeitumgebungen und stellt

die Designgrundstruktur für die Entwicklung zur Verfügung

**GPS** - Globales Positionsbestimmungssystem

**HTML** - Hypertext Markup Language

**IDE** - integrierte Entwicklungsumgebung

**Java EE(Enterprise)** - Spezifikation einer Softwarearchitektur für die transaktionsbasierte Ausführung von in Java programmierten Anwendungen und insbesondere Web-Anwendungen

**JAX-RS** - Java API for RESTful Web Services

**JS** - JavaScript

**JSON** - einfach lesbares Datenformat für den Datenaustausch zwischen Anwendungen

**JWT** - auf JSON basierter genormter Access-Token, mit dem alle für die Authentifikation benötigten Informationen übertragen werden

**Logging** - Informationen über Programmaktivitäten ausgeben und speichern

**Maven** - Build-Management-Tool der Apache Software Foundation basierend auf Java

**OCR** - Texterkennung in z.B. Bildern via Software

**Package** - Ordnerähnliche Struktur zum Organisieren von Klassen in Programmen

**REST** - Standard für verteilte Systeme, speziell für Webdienste

**Ubuntu** - eine Linux Distribution

**VR** - virtuelle Realität

**Vuforia** - ein AR-Framework

**Wikitude** - ein AR-Framework

**WYSIWYG** - steht für What You See Is What You Get

**WYSIWYG-Editor** - Bearbeitungsprogramm für Grafik oder Text mit optischen Bedienelementen

## 1.6 Autoren der Diplomarbeit

Auf den beiden folgenden Seiten stellen sich die Autoren der Diplomarbeit vor.

### **1.6.1 Isabella Hundstorfer**



#### **Persönliche Daten:**

Name: Isabella Hundstorfer  
Geburtsdatum: 09. Juni 2000  
Adresse: Rappersdorf 24, 4621 Sipbachzell  
E-Mail: isabella.hundstorfer@gmail.com

#### **Schulausbildung:**

seit 09/2014 HTBLA Leonding, Medientechnik  
09/2010 – 07/2014 Hauptschule Sattledt  
09/2006 – 07/2010 Volksschule Sipbachzell

#### **Berufserfahrung:**

2018: 3-wöchiges Praktikum im IT-Support  
(Kremsmüller Industrieanlagenbau KG)  
4-wöchiges Auslandspraktikum im IT-Bereich  
(Universität von Kreta)

2017: 6-wöchiges Praktikum im Bereich Softwareentwicklung  
(Silbergrau Consulting & Software GmbH)

2016: 4-wöchiges Praktikum im Büro  
(Grundner Sondermaschinen GmbH)

#### **Qualifikationen**

2018: Anerkennung beim Prix Ars Electronica (Maladidea)  
2017: Bundessieg beim Jugendrotkreuz Erste Hilfe-Bewerb  
2016: Landessieg beim Wettbewerb Känguru der Mathematik  
HTL Leonding: Oracle Database Programming Certificate

## 1.6.2 Christopher Gusenbauer



### Persönliche Daten:

Name: Christopher Gusenbauer  
Geburtsdatum: 18. April 2000  
Adresse: Mühlbachstraße 65c, 4073 Wilhering  
E-Mail: chrisi.gusenbauer101@gmail.com

### Schulausbildung:

seit 09/2014 HTBLA Leonding, Medientechnik  
09/2010 – 07/2014 Stiftsgymnasium Wilhering  
09/2006 – 07/2010 Volksschule Wilhering

### Berufserfahrung:

2017: 4-wöchiges Praktikum im Marketing & SEO Bereich  
(Banner Batterien)  
2016: 4-wöchiges Praktikum im IT-Bereich  
(Gesundheitsinformatik Kepleruniklinikum)

### Qualifikationen

2018/19: Schulsprecher (HTBLA Linz-Leonding)  
2018: 2. Platz beim Fremdsprachenwettbewerb der BHS  
Oberösterreich in Englisch (für HTBLA Leonding)  
2018: Rhetorik und Projektmanagementtrainings im Umfeld der  
LSV/UHS  
HTL Leonding: Oracle Database Programming Certificate  
Persönlich: Google Analytics für Fortgeschrittene

# Kapitel 2

## Pflichtenheft

### 2.1 Zielbestimmungen

Das Ziel des Projektes ist es, eine Multiplattform Notiz-App zu erstellen, die Notizen und digitale Informationen zu Objekten des echten Lebens mithilfe von Objekterkennung und erweiterter Realität zuordnen kann. Diese App sollte großen Unternehmen mit großen Werkshallen und vielen Maschinen dabei helfen, Informationen zu ihren Maschinen leichter darstellen zu können.

#### 2.1.1 Musskriterien

Der Benutzer-Account

- Der Internet-Benutzer kann sich selbst am System registrieren.
- Der Benutzer kann sich am System anmelden und vom System abmelden.
- Der Benutzer kann Notizen erstellen und bearbeiten.
- Der Benutzer kann Notizen mit Objekten verknüpfen.
- Notizen können aus Text, Bildern, Dokumenten und Terminen bestehen.
- Erfasste Objekte im Umkreis werden dem Benutzer angezeigt.
- Mithilfe der Kamera bzw. GPS können Objekte erkannt und aufgerufen werden.

#### 2.1.2 Wunschkriterien

Text auf Bildern wird erkannt und in die Tags der Notizen gespeichert.

#### 2.1.3 Abgrenzungskriterien

Das System ist nicht, wie etwa Evernote als persönlicher Notizzettel gedacht, sondern als Software für Wissensaustausch, wobei die Notizen mit Objekten und nicht nur mit Benutzern verknüpft sind.

## **2.2 Produkteinsatz**

### **2.2.1 Anwendungsbereiche**

Die App soll Notizen und Informationen mit realen Objekten verknüpfen können. Die Informationen sollten mit anderen Benutzern geteilt werden können. Informationen aus Bildern etc. sollten in die Tags der Notizen einfließen und sie nach ihnen durchsuchbar machen. Die Anwendung ist für Android und Apple verwendbar.

### **2.2.2 Zielgruppen**

Unternehmen können z. B. Information über Maschinen, Wartungsanleitung für Drucker, oder etwa Termine für Wartungsarbeiten, und vieles mehr speichern .  
Privatpersonen können z. B. Betriebsanleitung zu Geräten speichern, um Sie bei Problemen schnell griffbereit zu haben.

## **2.3 Produktumgebung**

Das Speichern der Objekte und Notizen ist betriebssystemunabhängig. Das Erkennen der Objekte funktioniert nur mit Android.

### **2.3.1 benötigte Software**

- Client
  - Browser für Verwaltung
  - Android für Objekterkennung und Verwaltung
- Server
  - Java
  - JPA
  - Wildfly
  - MySQL
  - Linux

### **2.3.2 benötigte Hardware**

- Client
  - Internetfähiges Mobilgerät mit Kamera und GPS
- Server
  - Internetfähiger Server, der oben genannte Server-Software installiert und konfiguriert, sowie ausreichend Rechen- und Festplattenkapazität hat.

### 2.3.3 Qualitätszielbestimmungen

	sehr wichtig	wichtig	weniger wichtig	unwichtig
Robustheit		x		
Zuverlässigkeit		x		
Korrektheit		x		
Benutzerfreundlichkeit	x			
Effizienz		x		
Portierbarkeit			x	
Kompatibilität			x	
Design		x		
Sicherheit		x		

## 2.4 Entwicklungsumgebung

### 2.4.1 Software

- Plattform
  - MySQL - 5.7.24
  - Wildfly 14.0.1
  - Java-EE: jboss-javaee-7.0
  - Nginx 1.15.1
  - Windows 10
  - Ionic 3.9.2
  - Angular 6.1.3
  - Android 5.0.1
  - Vuforia 8.0
  - Wikitude SDK 8.1
- Tools
  - Netbeans
  - Visual Studio Code
  - Android Studio
  - Browser (Chrome 67.0.3396.87, Mozilla 61.0)

### 2.4.2 Hardware

- Laptop oder Computer

# Kapitel 3

## Augmented Reality

### 3.1 Definition

Augmented Reality (AR) oder auf Deutsch **erweiterte Realität** sollte nicht mit Virtual Reality (VR) verwechselt werden. Bei VR kann der Benutzer die Welt um sich herum nicht sehen. Er ist in einer künstlichen Umgebung. Bei Augmented Reality sieht der Benutzer die reale Welt um sich herum, wobei darüber virtuelle Objekte wie Bilder, Texte oder 3D-Objekte gelegt werden.

In der Wissenschaft wird AR mit folgenden Charakteristiken definiert:

- Kombination von realen und virtuellen Elementen
- Reale und virtuelle Objekte stehen dreidimensional in Bezug zueinander
- Interaktion in Echtzeit

[7]

### 3.2 Verwendungsbereiche

#### 3.2.1 Medizin

Ein medizinischer Eingriff erfordert sehr viel Fingerspitzengefühl. Die Chirurgen müssen exakte, millimetergenaue Arbeit an ihren Patienten leisten. Dafür braucht der Chirurg Röntgen-, Ultraschall-, Computer Tomografie- (CT) und Magnetresonanztomografieaufnahmen (MRI), welche auf einem Monitor angezeigt werden. Mit Augmented Reality kann dies bald der Vergangenheit angehören.

Der Chirurg könnte mithilfe einer Datenbrille einen Blick in den Körper des Patienten werfen. Vitalwerte, Röntgenbilder, Hinweise und Empfehlungen, die bisher auf separaten Monitoren angezeigt werden, könnte er dadurch direkt im Blickfeld haben.



Abbildung 3.1: Augmented Reality in der Medizin

Auch im Medizinstudium wird mit AR gearbeitet, um den Studierenden den Aufbau des Körpers zu erklären. [45]

### 3.2.2 Industrielle Anwendungen

Beim Reparieren von komplexen Maschinen kann AR eingesetzt werden. Anweisungen sind durch geeignete AR Überlagerungen leichter verständlich als mit Bildern und Texten. [7]

### 3.2.3 Verknüpfung von Informationen mit Objekten

AR kann benutzt werden, um Objekte mit Informationen zu verknüpfen. Beispielsweise können dadurch auf dem Display beim Durchgehen durch eine Bibliothek zu Büchern Name, Autor, Inhalt und viele weitere nützliche Informationen angezeigt werden. [7]

### 3.2.4 Unterhaltung



Abbildung 3.2: Pokemon Go

Schon jetzt wird AR in Museen und bei Kulturdenkmälern eingesetzt, um den Besuchern mehr Informationen zu geben. Durch AR ist es möglich, den Besuchern zerfallene

Bauwerke ohne teure Restaurierungen zu zeigen, wie sie einmal ausgesehen haben. Augmented Reality wird aber auch für Spiele verwendet. Das bekannteste Beispiel ist wohl „Pokemon Go“. [7]

### 3.2.5 Militär

Ursprünglich wurde AR vom Militär genutzt, mittlerweile wird AR auch in vielen anderen Bereichen genutzt. Den Piloten wurden im Helm mit Vektorgrafiken Informationen zur Navigation, wie zum Flug angezeigt. Durch ständige Weiterentwicklung könnten AR-Brillen schon bald zur Standardausrüstung von Bodentruppen gehören. Durch die Verwendung tragbarer Brillen und Headsets, könnten wichtige Daten auf das Schlachtfeld gelegt werden. Topographische Daten könnten zusammen mit Video-Feeds von entfernten Drohnen oder anderen Einsatzkräften an einen Trupp weitergeleitet werden. Trotzdem bekommt der Träger der AR-Brille sein Umfeld noch mit. [7]

### 3.2.6 Architektur

AR wird eingesetzt, um sich geplante Gebäude besser vorstellen zu können. Vor allem, aber auch um einen besseren Blick dafür zu bekommen, ob ein Gebäude zur Umgebung passt. [7]

### 3.2.7 Werbung

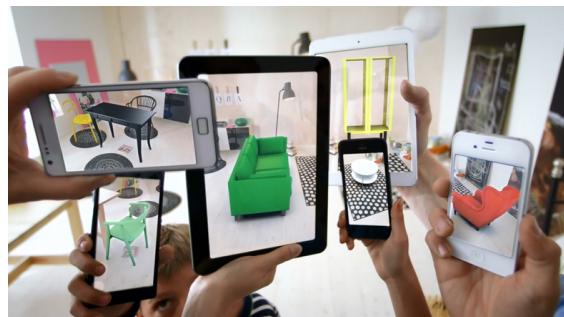


Abbildung 3.3: Augmented Reality IKEA-App

Um die Kunden von Produkten zu begeistern, wird teilweise auch schon Augmented Reality verwendet. Ein Beispiel hierfür bietet der von IKEA 2013 veröffentlichte Katalog mit dazugehörender AR-App. Durch die App und den Katalog ist es möglich, die Möbeln in den eigenen vier Wänden beliebig zu platzieren. [29]

### 3.2.8 Bildung

Durch AR ist es möglich komplexe Konzepte durch Animationen in der realen Welt darzustellen. Genauso ist es möglich, historische Ereignisse und Gebäude in die heutige Umgebung zu integrieren, dadurch wird Geschichte anschaulicher. [7]

### **3.3 Technische Voraussetzungen**

Um AR nutzen zu können wird ein Gerät mit Kamera und Display wie zum Beispiel ein Smartphone, ein Tablet oder auch einen Laptop benötigt. Natürlich können auch eigens dafür entwickelte Smart Glasses (AR-Brillen) verwendet werden. Das Gerät muss genügend Leistung aufbringen können, um die Bilder in Echtzeit analysieren zu können. [7]

### **3.4 Geschichte und Entwicklung**

Im Jahr 1901 schreibt L. Frank Baum (bekannt durch „Der Zauberer von Oz“) das erste mal über eine Brille, die Informationen zu den Charaktereigenschaften der Figuren einblendet. Das erste AR-System wurde 1968 von Ivan Sutherland an der Unitversität Utah vorgestellt. Der Begriff Augmented-Reality entstand erst in den 1990ern. 1997 wurde das erste mobile AR-System entwickelt. Es wurden GPS-Informationen genutzt, um Informationen über die Gebäude des Campuses über ein „Head-Worn-Display“ anzuzeigen. Um die nötige Rechenleistung zu erreichen, musste ein Laptop auf dem Rücken getragen werden. Wikitude, der erste Augmented Reality Browser für das Smartphone, wurde 2008 veröffentlicht. Ein Jahr später mit der Verbreitung des „iPhone 3GS“, wurde AR für den Standardanwender interessant. Das iPhone 3GS war das erste tragbare Gerät, das über ausreichend Leistung und die nötigen Sensoren verfügte. [20]

### **3.5 Funktionsweise**

Es gibt unterschiedliche Arten wie AR funktionieren kann. Es gibt welche mit, sowie welche ohne Marker.

#### **3.5.1 Marker-Based Augmented Reality**



Abbildung 3.4: Marker-Based Augmented Reality

Man könnte es auch Bilderkennung nennen, da zum Verknüpfen realer mit irrealen Elementen ein spezielles visuelles Objekt und eine Kamera erforderlich sind. Diese visuellen

Objekte können QR-Codes oder andere einfache Symbole sein. Das AR-Gerät berechnet in einigen Fällen auch die Position und Ausrichtung einer Markierung, um den Inhalt zu positionieren. So veranlasst eine Markierung digitale Animationen, die dem Benutzer angezeigt werden können, und Bilder in einem Magazin werden möglicherweise zu 3D-Modellen. [32]

### 3.5.2 Markerless Augmented Reality



Abbildung 3.5: Markerless Augmented Reality

Standortbasierte oder positionsbasierte Augmented Reality-Anwendungen verwenden GPS-Daten, den Kompass, das Gyroskop und den Beschleunigungssensor, um Daten, basierend auf der Position des Benutzers, bereitzustellen. Diese Daten bestimmen dann, welche AR-Inhalte in einem bestimmten Bereich angezeigt werden. [32]

### 3.5.3 Projection-based Augmented Reality



Abbildung 3.6: Projection-based Augmented Reality

3D-Projektionen von synthetischem Licht auf physische Oberflächen ermöglicht in manchen Fällen die Interaktion mit dem Objekt. Diese Art von AR nennt man auch Hologramme, die uns aus Sciencefiction-Filmen, wie „Star Wars“ bekannt sind. [32]

### 3.5.4 Superimposition-based Augmented Reality



Abbildung 3.7: Superimposition-based Augmented Reality

Durch superimposition-based AR wird die Kameraansicht vollständig oder teilweise durch eine erweiterte Ansicht ersetzt. Hierbei spielt die Objekterkennung eine wichtige Rolle. Ein Beispiel hierfür ist die IKEA-Katalog-App, mit der Benutzer virtuelle 3D-Abbildungen ihrer Möbel aus dem Möbelkatalog in ihre Räume platzieren können. [32]

# Kapitel 4

## AR-Frameworks

### 4.1 Frameworkübersicht

Mittlerweile sind eine Menge an AR-Frameworks am Markt. Es folgt eine nicht vollständige Liste von AR-Frameworks, die einer näheren Recherche unterzogen wurden. Viele der Frameworks waren ungeeignet, weil sie keine ausführliche Dokumentation haben, die Objekte nur für wenige Stunden speichern oder die Objekte nicht dynamisch erstellt werden können. In den folgenden Seiten wird auf die beiden in unserer Diplomarbeit verwendeten Frameworks genauer eingegangen.

- Wikitude
- Vuforia
- ARCore
- ARKit
- EasyAR
- Kudan
- Maxst
- DeepAR
- EasyAR
- ARToolKit
- VOID AR

## 4.2 Vuforia



Abbildung 4.1: Vuforia Logo

Vuforia ist ein Augmented Reality Software Development Kit (SDK) zum Erstellen von Augmented Reality-Anwendungen für Mobilgeräte. Vuforia verwendet die Computer-Vision-Technologie, um in Echtzeit ebene Bilder (Image Targets) und einfache 3D-Objekte zu erkennen und zu verfolgen. Diese Methode ermöglicht, es virtuelle Objekte in Echtzeit auf dem Bildschirm anzuzeigen.

Die „API“ kann mit „C++“, „Java“, „Objective-C++“ (Kombination aus C++ - und Objective-C-Syntax) und den „.NET-Sprachen“ über eine Erweiterung der „Unity-Spiel-Engine“ verwendet werden. Auf diese Weise unterstützt das „SDK“ sowohl native Entwicklung für „iOS“ und „Android“ als auch die Entwicklung von AR-Anwendungen in Unity.

Vuforia wurde im November 2015 von PTC Inc. übernommen. [31]

### 4.2.1 Vorteile

- Extrem viele Möglichkeiten: „Model Targets“, „Image Targets“, „Multi Targets“, „Cylinder Targets“, „Object Targets“, „VuMarks“, „Virtual Buttons“, „Cloud Recognition“

Objekt	Vuforia	Wikitude	VOID AR
Kupferschild	7	6	4
Gesünder leben	12	6.5	9
Tür 5	28	4	12
Tür 4	23	8	6
Eiseneimer	5	2	0
Volkshilfe	23	5.5	15
Buntglascontainer	3.8	5	2
Öffnungszeiten	25	13	9
Tür 2	0	4	0
Eisenbahnschriftzug	60	40	18
Resultate	186.8	94	75

Abbildung 4.2: Vergleich von Vuforia-Wikitude-Void AR aus der Diplomarbeit „Augmented Reality Machbarkeitsstudie Hofburg-App“ Seite 92

- Bessere Objekterkennung mit „Vuforia“ als mit „Wikitude“ - siehe in der Diplomarbeit „Augmented Reality Machbarkeitsstudie Hofburg-App“ auf der Seite 91f. „Vuforia“ kann Objekte erkennen auch wenn lediglich ein kleiner Ausschnitt des Objektes im Bildausschnitt zu sehen ist. Die Konkurrenz kann das nicht. [47]

#### 4.2.2 Nachteile

- Einarbeitung sehr zeitintensiv
- Dokumentation teilweise veraltet und Vieles ist schwierig zu finden

	Classic	Cloud	Pro
Price	\$499 one-time	\$99 / mo	Contact Us
License Eligibility	Apps built for companies with revenue under \$10 Million/year	Apps built for companies with revenue under \$10 Million/year	No revenue restriction
Content Placement			
On Surfaces	●	●	●
On Objects from Images	●	●	●
from VuMarks	100	100	Custom via API
from 3D Scans			●
from 3D Models			●
Advanced APIs			
Advanced Camera			●
External Camera			●
Cloud database			
Max Size (# of images)		100,000	Custom
Max Usage (Reco./mo)		10,000	Custom
Support			
Software Updates		●	●
Direct Support			●
Price	\$499 one-time	\$99 / mo	Contact Us
	Buy Classic	Buy Cloud	Contact Us

Abbildung 4.3: Vuforia Preise: Screenshot 6.3.2019

## 4.3 Wikitude



Abbildung 4.4: Wikitude Logo

Wikitude bietet seit 2008 Augmented-Reality-Technologie an. Das Unternehmen hat seinen Sitz in Salzburg. Das Kernprodukt des Unternehmens ist das „Wikitude-SDK“. Das SDK wurde erstmals im Oktober 2008 auf den Markt gebracht und umfasst Bilderkennung und -verfolgung, 3D-Modell-Rendering, Video-Overlay und standortbasiertes AR. 2017 hat Wikitude die „SLAM-Technologie“ (Simultaneous Localization And Mapping) eingeführt, die die Objekterkennung und -verfolgung sowie die markenlose Sofortverfolgung ermöglicht.

Das „SDK“ ist für „Android“, „Windows“ und „iOS“ verfügbar. [34]

### 4.3.1 Vorteile

- österreichisches Unternehmen: Viele Foreneinträge auf Deutsch
- Serverstandort kann ausgewählt werden
- Plattformunabhängig
- einfach zum Verwenden, funktioniert sogar mit Javascript
- kostenlose Schülerlizenzen

### 4.3.2 Nachteile

- Objekterkennung funktioniert nicht so gut wie bei Vuforia
- Aussehen etwas veraltet

 <p><b>SDK STARTUP</b></p> <p><b>free</b></p> <p><a href="#">View Product</a></p>	 <p><b>WIKITUDE DEMO</b></p> <p><b>499 €</b></p> <p><a href="#">View Product</a></p>	 <p><b>SDK PRO (ONE TIME)</b></p> <p><b>1990 €</b></p> <p><a href="#">View Product</a></p>	 <p><b>SDK PRO 3D (ONE TIME)</b></p> <p><b>2490 €</b></p> <p><a href="#">View Product</a></p>
<p><b>FEATURES</b></p> <ul style="list-style-type: none"> <li>✓ Geo</li> <li>✓ 2D Image Recognition</li> <li>✓ 3D Engine</li> <li>✓ Instant Tracking (SMART)</li> <li>✓ Object Recognition</li> <li>✓ Scene Recognition</li> <li>Cloud Recognition</li> <li>Enterprise Apps</li> <li>Smart Glasses</li> </ul>	<p><b>FEATURES</b></p> <ul style="list-style-type: none"> <li>✓ Geo</li> <li>✓ 2D Image Recognition</li> <li>✓ 3D Engine</li> <li>✓ Instant Tracking (SMART)</li> <li>✓ Object Recognition</li> <li>✓ Scene Recognition</li> <li>Cloud Recognition</li> <li>Enterprise Apps</li> <li>Smart Glasses</li> </ul>	<p><b>FEATURES</b></p> <ul style="list-style-type: none"> <li>✓ Geo</li> <li>✓ 2D Image Recognition</li> <li>✓ 3D Engine</li> <li>✓ Instant Tracking (SLAM)</li> <li>Object Recognition</li> <li>Scene Recognition</li> <li>Cloud Recognition</li> <li>Enterprise Apps</li> <li>Smart Glasses</li> </ul>	<p><b>FEATURES</b></p> <ul style="list-style-type: none"> <li>✓ Geo</li> <li>✓ 2D Image Recognition</li> <li>✓ 3D Engine</li> <li>✓ Instant Tracking (SLAM)</li> <li>✓ Object Recognition</li> <li>✓ Scene Recognition</li> <li>Cloud Recognition</li> <li>Enterprise Apps</li> <li>Smart Glasses</li> </ul>
<p><b>SERVICE</b></p> <ul style="list-style-type: none"> <li>✓ Extensions</li> <li>✓ 1 iOS + 1 Android + 1 Windows App</li> <li>✓ Studio Offline Export</li> <li>Studio Hosting</li> <li>Enterprise API/Script</li> <li>Self Hosting</li> </ul>	<p><b>SERVICE</b></p> <ul style="list-style-type: none"> <li>✓ Extensions</li> <li>✓ 1 iOS + 1 Android + 1 Windows App</li> <li>✓ Studio Offline Export</li> <li>Studio Hosting</li> <li>Enterprise API/Script</li> <li>Self Hosting</li> </ul>	<p><b>SERVICE</b></p> <ul style="list-style-type: none"> <li>✓ Extensions</li> <li>✓ 1 iOS + 1 Android + 1 Windows App</li> <li>✓ Studio Offline Export</li> <li>Studio Hosting</li> <li>Enterprise API/Script</li> <li>Self Hosting</li> </ul>	<p><b>SERVICE</b></p> <ul style="list-style-type: none"> <li>✓ Extensions</li> <li>✓ 1 iOS + 1 Android + 1 Windows App</li> <li>✓ Studio Offline Export</li> <li>Studio Hosting</li> <li>Enterprise API/Script</li> <li>Self Hosting</li> </ul>
<p><b>SUPPORT</b></p> <ul style="list-style-type: none"> <li>✓ Forum</li> <li>SDK Updates</li> <li>Email</li> <li>Priority Tickets</li> <li>Phone</li> <li>Custom SLA</li> <li>Developer Training</li> </ul>	<p><b>SUPPORT</b></p> <ul style="list-style-type: none"> <li>✓ Forum</li> <li>SDK Updates</li> <li>Email</li> <li>Priority Tickets</li> <li>Phone</li> <li>Custom SLA</li> <li>Developer Training</li> </ul>	<p><b>SUPPORT</b></p> <ul style="list-style-type: none"> <li>✓ Forum</li> <li>SDK Updates</li> <li>Email</li> <li>Priority Tickets</li> <li>Phone</li> <li>Custom SLA</li> <li>Developer Training</li> </ul>	<p><b>SUPPORT</b></p> <ul style="list-style-type: none"> <li>✓ Forum</li> <li>SDK Updates</li> <li>Email</li> <li>Priority Tickets</li> <li>Phone</li> <li>Custom SLA</li> <li>Developer Training</li> </ul>

Abbildung 4.5: Wikitude Preise: Screenshot 6.3.2019

# Kapitel 5

## Client

### 5.1 Überblick

Der Client unserer Diplomarbeit wurde größtenteils als hybride Applikation für Ionic implementiert. Nur die Objekterkennung wurde nativ in Android implementiert.

### 5.2 Desktop-Client

Als Entwicklungsumgebung für den Client wurde „Visual Studio Code“ und „Android-Studio“ verwendet. „Visual Studio Code“ ist eine kostenlose Entwicklungsumgebung. Es ist sehr übersichtlich und einfach gestaltet und bietet sehr viele Plugins und Erweiterungen, die dem Programmierer das Arbeiten vereinfachen. Es bietet eine integrierte Git-Versionskontrolle, dadurch wird das Arbeiten im Team erleichtert.

Der Desktop-Client, entweder als Webseite im Browser oder als „PWA“ (Progressive Web App), gibt dem Benutzer die Möglichkeit, alle für ihn sichtbaren Objekte anzuzeigen, zu bearbeiten, zu löschen, Notizen hinzuzufügen, zu durchsuchen, neue Objekte hinzuzufügen und Benutzergruppenabhängigkeiten anzuzeigen.

#### 5.2.1 Login

Ein Benutzer kann sich mit seiner E-Mail-Adresse und seinem Passwort einloggen. Die gesamte Kommunikation mit dem Server läuft über „https“. Das heißt, die Kommunikation zwischen Server und Client wird verschlüsselt. Wenn die E-Mail-Adresse und das Passwort gültig sind, wird der Benutzer eingeloggt.

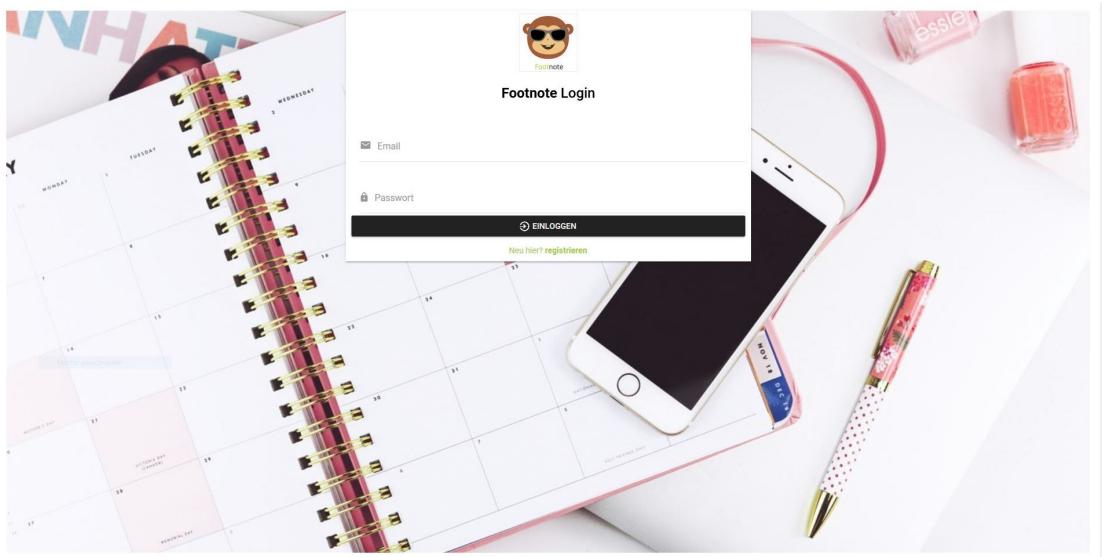


Abbildung 5.1: Screenshot: Desktop Loginansicht

### 5.2.2 Registrieren

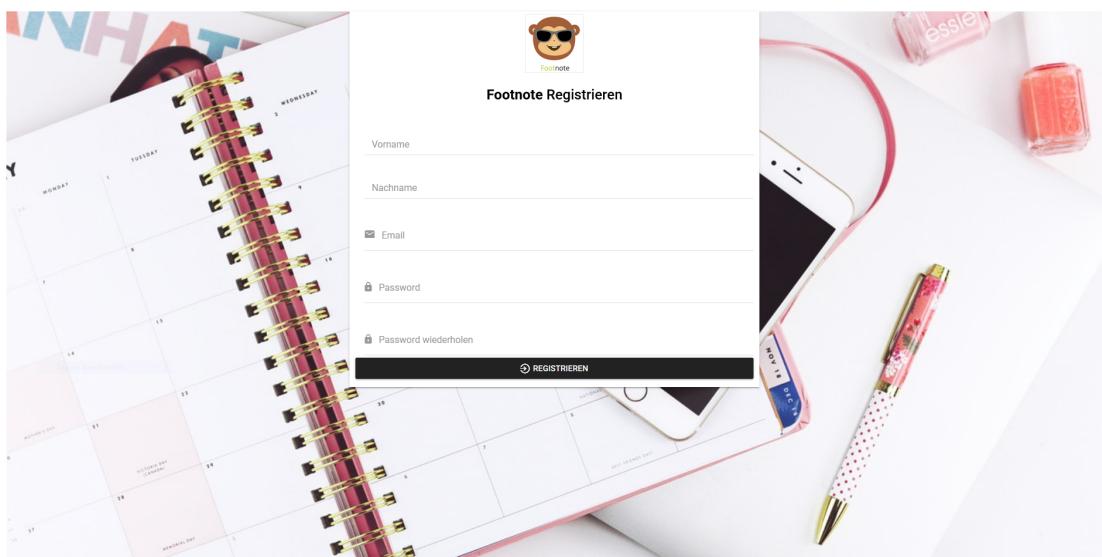


Abbildung 5.2: Screenshot: Desktop Registrierenansicht

Wenn sich ein neuer Benutzer registrieren möchte, muss er seinen Vornamen, seinen Nachnamen, seine E-Mail-Adresse und ein Passwort doppelt eingeben. Nach dem Klicken auf Registrieren, bekommt der Benutzer eine E-Mail mit einem Link. Nach Öffnen des Links ist die Registrierung abgeschlossen.

### 5.2.3 Home

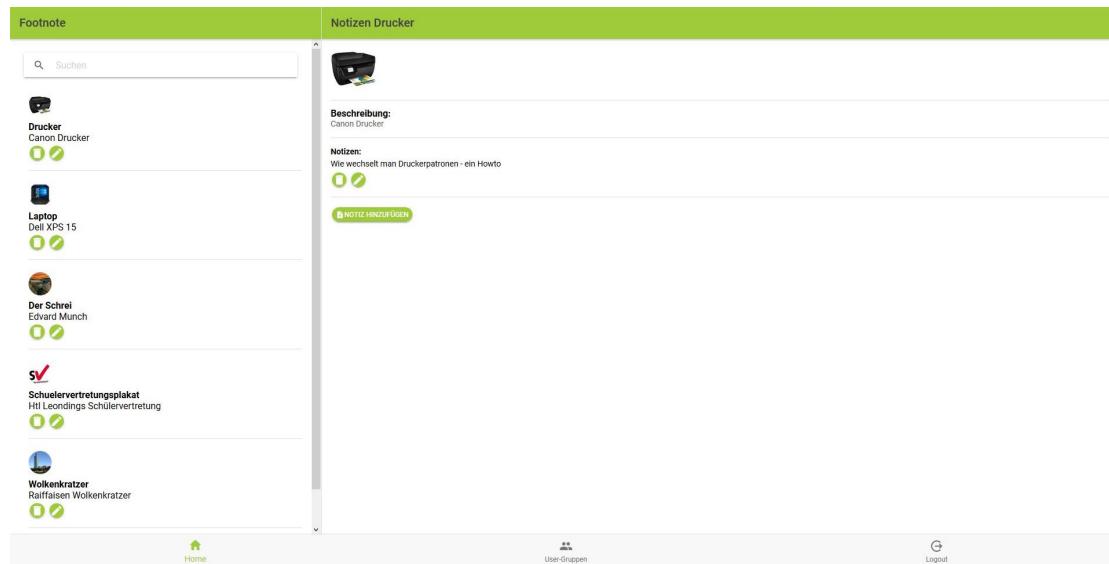


Abbildung 5.3: Screenshot: Desktop Homeansicht mit Details

Der erste Tab (Home) ist derjenige, der automatisch nach dem Login angezeigt wird.



Abbildung 5.4: Screenshot: Desktop Objekt bearbeiten

Oben befindet sich eine Suchleiste, mit der die Objekte durchsucht werden können. Darunter werden alle Objekte in einer Liste angezeigt. Jedes Objekt wird mit Bild, Titel und Beschreibung angezeigt. Die Objekte können mit dem „Löschen-Button“ gelöscht werden

und mit dem „Bearbeiten-Button“ bearbeitet werden. Beim Anklicken eines Objektes wird die Detailansicht geöffnet.

In der Detailansicht, die sich bei einer Bildschirmbreite von mindestens 768 px direkt rechts daneben öffnet, ist das Bild, der Titel und die Beschreibung des Objektes zu sehen. Zusätzlich sind auch alle zu diesem Objekt erstellten Notizen zu sehen. Die Notizen können bearbeitet und gelöscht werden. Es können auch neue Notizen erstellt werden.

#### 5.2.4 Objekt erstellen

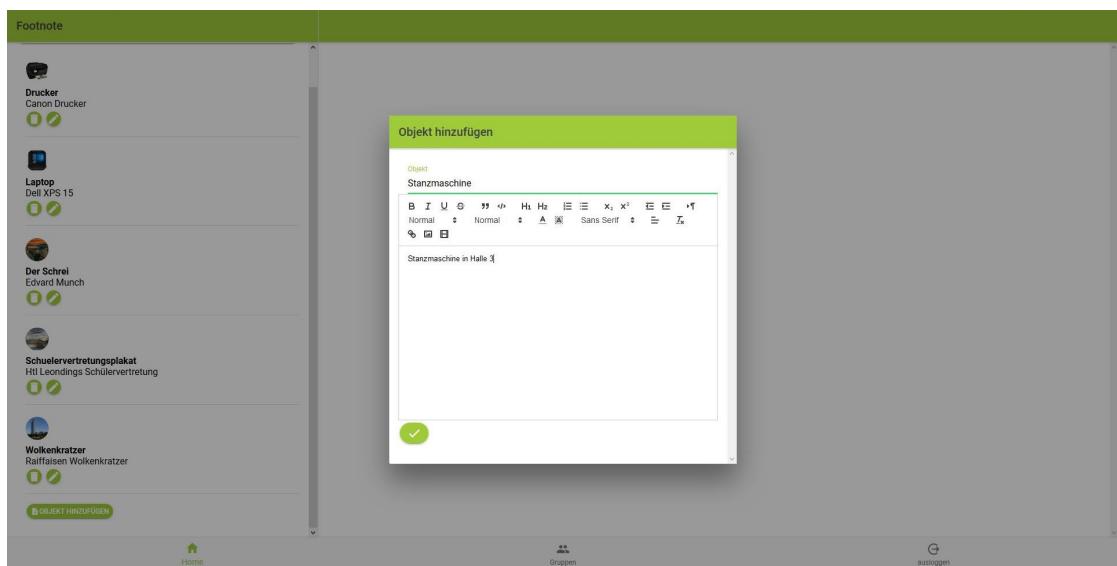


Abbildung 5.5: Screenshot: Desktop Objekt erstellen

Mithilfe des Buttons „Objekt erstellen“ am Ende der Objektliste können neue Objekte erstellt werden. Für ein neues Objekt muss zuerst der Name und eine Beschreibung des Objektes eingegeben werden. Die Koordinaten des neuen Objektes werden durch den aktuellen Standort, der über GPS ausgelesen wird, gesetzt. Danach muss der Benutzer noch ein Bild hochladen. Dieses Bild wird für die Objekterkennung mit „Vuforia“ verwendet.

#### 5.2.5 Benutzer-Gruppen

In der Benutzergruppenansicht kann eine Benutzergruppe ausgewählt werden, zu dieser Benutzergruppe kann es einen oder mehrere Feeds geben, die wiederum ausgewählt werden können.

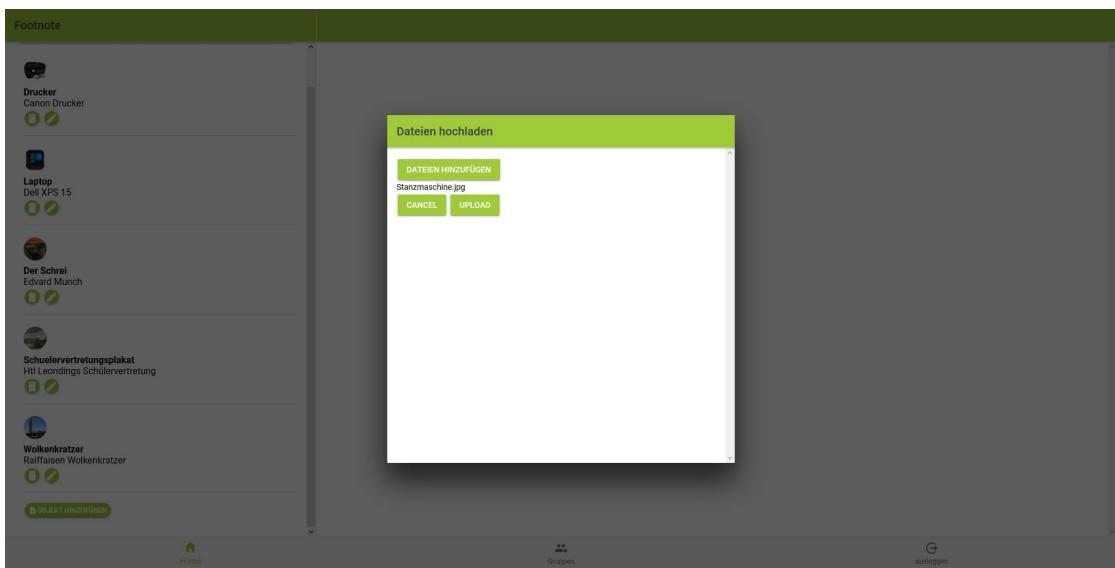


Abbildung 5.6: Screenshot: Layout Desktop Bild hochladen

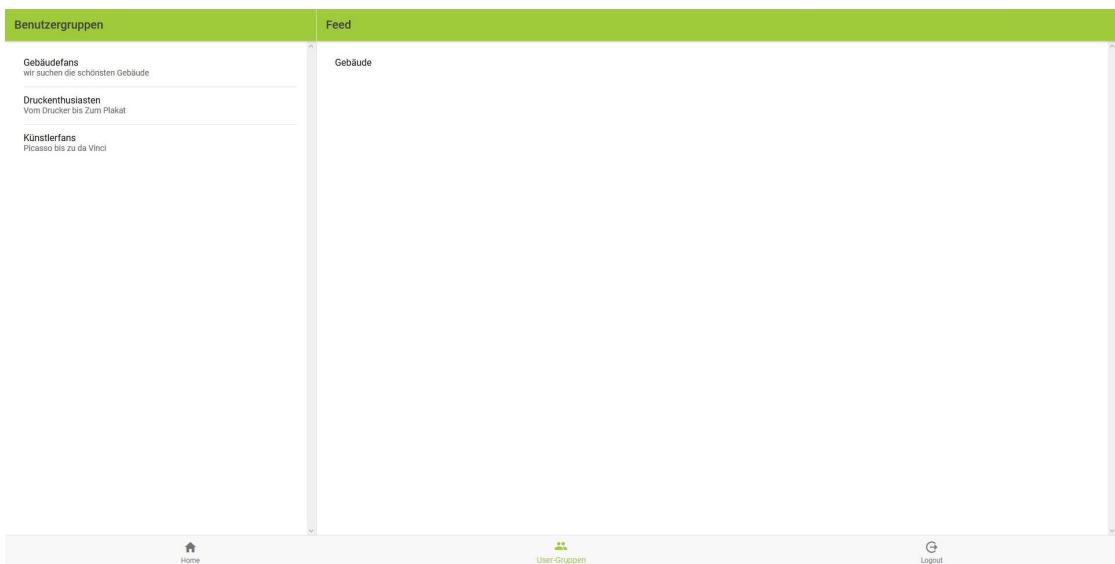


Abbildung 5.7: Screenshot: Desktop Feed

### 5.2.6 Logout

Durch den Klick auf den „Logout-Button“ wird der Benutzer ausgeloggt.

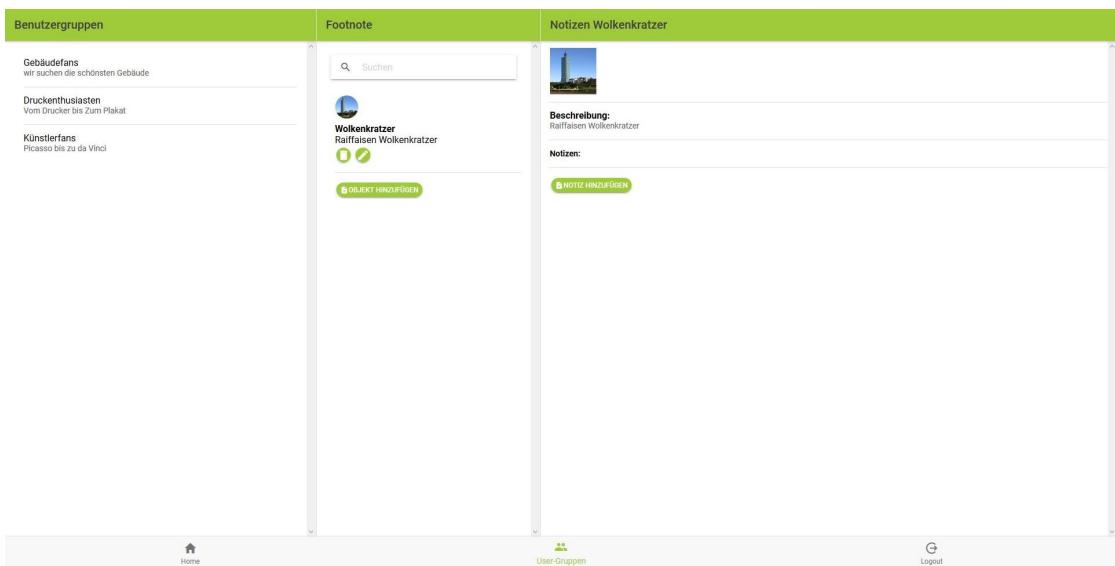


Abbildung 5.8: Screenshot: Desktop Benutzergruppen

### 5.3 Android-Client

Der Android-Client beinhaltet alle Bestandteile des Desktop-Clients und ein paar weitere Bestandteile. Das Design ist ein bisschen anders aufgebaut, wie auf den folgenden Seiten zu sehen ist.

#### 5.3.1 Login

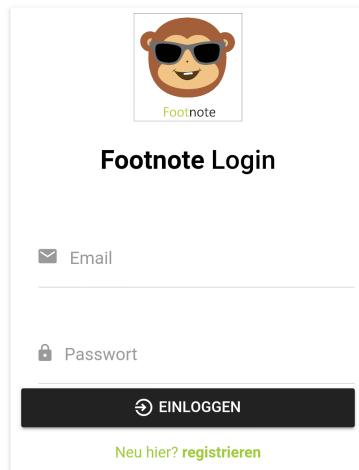


Abbildung 5.9: Screenshot: Smartphone Loginansicht

Der Login am Android-Client funktioniert gleich wie in der Desktop-Version.

### 5.3.2 Registrieren

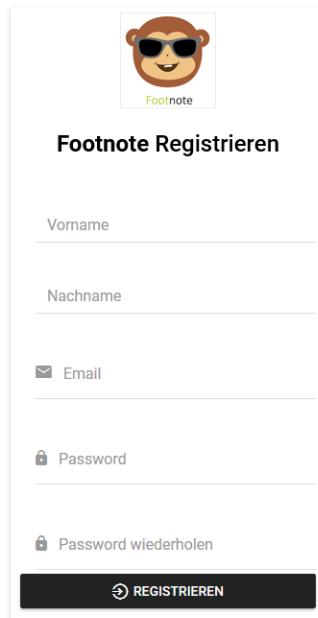


Abbildung 5.10: Screenshot: Smartphone Registrierenansicht

Das Registrieren am Android-Client funktioniert gleich wie in der Desktop-Version.

### 5.3.3 Home

Der erste Tab (Home) ist derjenige, der automatisch nach dem Login angezeigt wird. Oben befindet sich eine Suchleiste, mit der die Objekte durchsucht werden können.

Darunter werden alle Objekte in einer Liste angezeigt. Jedes Objekt wird mit Bild, Titel und Beschreibung angezeigt. Die Objekte können mit dem „Löschen-Button“ gelöscht werden und mit dem „Bearbeiten-Button“ bearbeitet werden. Beim Anklicken eines Objektes wird die Detailansicht geöffnet. Also soweit alles gleich wie in der Desktop-Version.

Doch bei einem Smartphone-Bildschirm, der kleiner als 768 px ist, öffnet sich die Detailansicht nicht daneben, sondern überlagert die Listenansicht. In der Detailansicht ist wieder das Bild, der Titel und die Beschreibung des Objektes zu sehen. Zusätzlich sind alle zu diesem Objekt erstellten Notizen zu sehen. Die Notizen können bearbeitet und gelöscht werden. Es können auch neue Notizen erstellt werden.

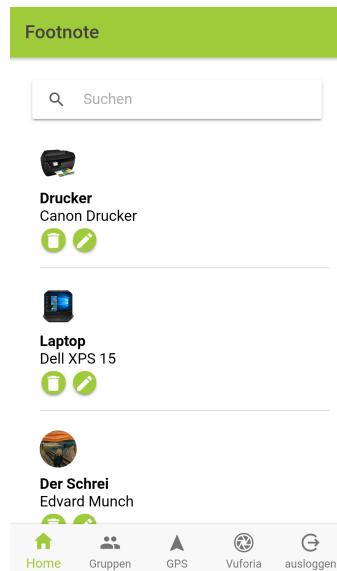


Abbildung 5.11: Screenshot: Smartphone Homeansicht

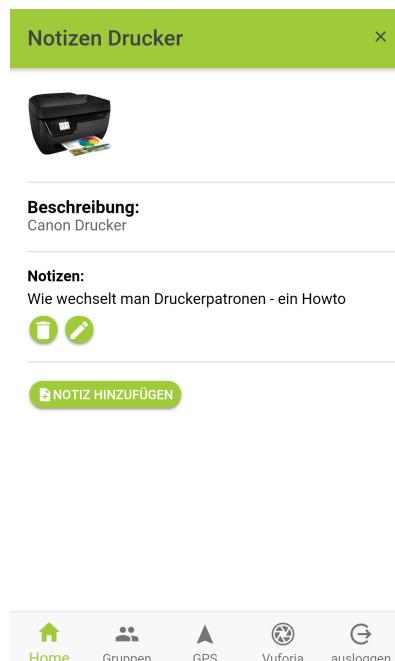


Abbildung 5.12: Screenshot: Smartphone Objekt-Detailansicht

#### 5.3.4 Benutzer-Gruppen

Die Benutzergruppenansicht fast gleich wie die Desktop-Version aufgebaut.

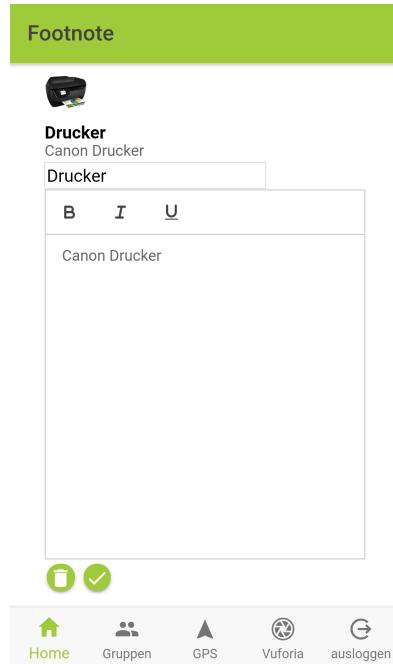


Abbildung 5.13: Screenshot: Smartphone Objekt bearbeiten

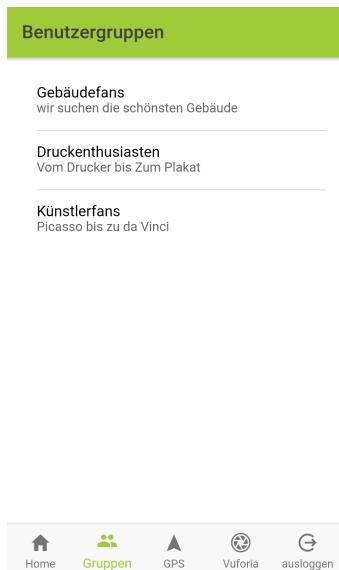


Abbildung 5.14: Screenshot: Smartphone Benutzergruppen

Wie in der Home-Ansicht sind die sonst nebeneinanderliegenden Ansichten übereinandergelegt. Zuerst muss die Benutzergruppe ausgewählt werden. Danach kann der Feed ausgewählt werden. Erst dann ist eine Liste aller in diesem Feed vorhandenen Objekte zu sehen.



Abbildung 5.15: Screenshot: Smartphone Feed

Eine Benutzergruppe können Personen mit gleichem Interesse sein, aber auch Personen einer Firma. Der Feed ist eine Feinunterteilung der einzelnen Benutzergruppen. Ein Feed kann bei einer Firma z.B. die Halle 1 und die Halle 2 sein. Genauso kann es aber auch in der Benutzergruppe Kunstfans mehrere Feeds für unterschiedliche Künstler geben.

### 5.3.5 Umgebungsansicht

In der Umgebungsansicht sind die Objekte in der realen Umgebung dort platziert zu sehen, wo sie laut den Koordinaten sind. Beim Klicken auf ein Objekt wird in eine Detailansicht gewechselt, in der wiederum der Titel, die Beschreibung, und die Notizen und zusätzlich die Entfernung zum Objekt angezeigt wird.

Die Umgebungsansicht ist nur in der „Android“- und „iOS“-Version vorhanden. Die Umgebungsansicht wurde mithilfe von „Wikitude“ umgesetzt.

### 5.3.6 Vuforia

Die Vuforia-Ansicht ist die wichtigste Ansicht der gesamten Anwendung. Der Benutzer kann die Objekte, die am Server gespeichert sind, mithilfe der Kamera erkennen, um diese mit den dazu passenden Informationen vom Server zu überlagern. Die Überlagerung beinhaltet das dazugehörige Bild vom Server, den Objektnamen und die Objektbeschreibung. Diese Ansicht verwendet das AR-Framework Vuforia. Diese Ansicht ist nativ in „Android-Studio“ programmiert und wird von „Ionic“ nur gestartet.

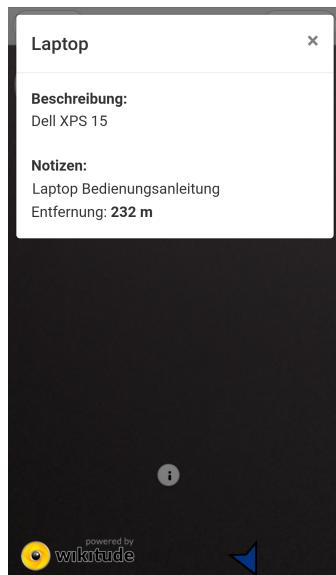


Abbildung 5.16: Screenshot: Smartphone Verwendung von Wikitude



Abbildung 5.17: Screenshot: Smartphone Verwendung von Vuforia

### 5.3.7 Logout

Durch das Klicken von „Logout“ wird der Benutzer ausgeloggt.

## 5.4 Ionic



Abbildung 5.18: Ionic-Logo

Ionic wird als Client-Framework verwendet, da es sehr einfach möglich ist, mithilfe von HTML, SCSS und TypeScript mobile Apps und Webseiten zu erstellen. Schon im letzten Schuljahr wurde ein größeres Projekt mit Angular umgesetzt und festgestellt, dass Angular zwar leicht zu verwenden ist, aber eine „PWA“ eine native App in manchen Bereichen nicht ersetzen kann. Ionic erstellt sozusagen aus einer Angular-Anwendung eine hybride App. Ein weiterer Vorteil von Ionic ist, dass sehr viele Plugins und UI-Komponenten zur Verfügung stehen.

### 5.4.1 Alternativen zu Ionic

#### 5.4.1.1 Angular

Angular ist ein TypeScript-basiertes JavaScript-Framework, das von Google entwickelt und erweitert wird. Angular wird von „Wix“, „Weather.com“, „Forbes“ und vielen anderen verwendet.

#### Vorteile

- Komponentenbasierte Architektur bietet höhere Codequalität.
- Google bietet einen Long-Time-Support für Angular an.
- Es sind sehr viele vorgefertigte Komponenten vorhanden.

#### Nachteile

- Die Migration von AngularJS auf neuere Angular Versionen ist sehr zeitaufwändig.
- Angular ist sehr umfassend und komplex.
- Keine hybride App sondern nur „PWA“ möglich.

[24]

#### **5.4.1.2 React Native**

„React Native“ ist ein Open-Source-Framework, das von Facebook entwickelt wurde. Es verwendet JavaScript und „JSX“ (JavaScript XML), um native Apps für iOS und Android zu erstellen. React wird von „Uber“, „Twitter“, „Netflix“, „Walmart“ und vielen anderen verwendet.

Mit den Layout-Komponenten können mit „React Native“ ansprechende UIs erstellt werden, die sich nicht von einer nativen App unterscheiden lassen.

#### **Vorteile**

- Gute Performance durch Verwendung von nativen Modulen.
- Geld und Zeit kann gespart werden, da nur eine App für mehrere Betriebssysteme geschrieben werden muss.

#### **Nachteile**

- Es gibt wenige vorgefertigte Komponenten, dadurch sind selbst kleine Anwendungen sehr zeitaufwändig.
- Die Speicherverwaltung ist in React Native ineffizient gelöst.

[26]

#### **5.4.1.3 Vue**

„Vue“ ist ein dynamisches System zum Erstellen von Benutzeroberflächen. Vue wird von „Alibaba“, „Nintendo“, „GitLab“ und vielen anderen verwendet.

#### **Vorteile**

- Vue ist sehr einfach lesbar, da die Vue-Komponenten eine Kombination aus HTML und JavaScript sind.
- Vue-Komponenten können leicht wiederverwendet werden.

#### **Nachteile**

- Die Mehrheit der Community spricht nicht Englisch, bei Problemen kann es daher länger dauern bis eine Lösung gefunden ist.
- Überflexibilität kann ein Projekt übermäßig komplizieren und zu mehr Fehlern und Unregelmäßigkeiten im Code führen, was wiederum das Projekt verzögern und die Entwicklungskosten erhöhen kann.

[25]

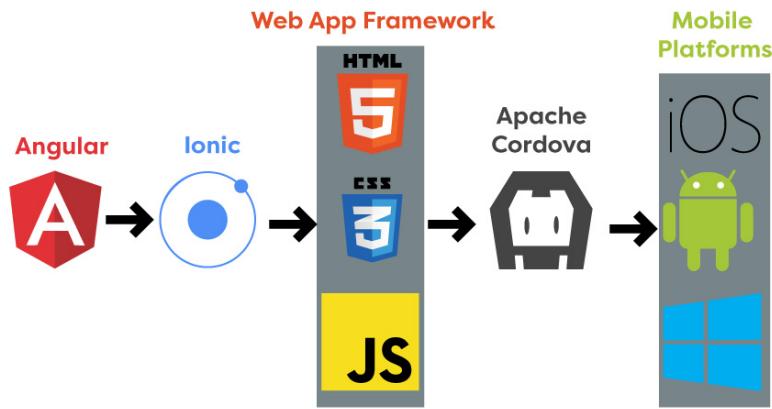


Abbildung 5.19: Ionic-Aufbau

#### 5.4.2 Generelle Informationen zu Ionic

Das Ionic Framework ist ein Open Source SDK (Software Development Kit), mit dem Entwickler performante, qualitativ hochwertige mobile Apps mit bekannten Webtechnologien (HTML, SCSS und TypeScript) erstellen können.

Ionic konzentriert sich hauptsächlich auf das Erscheinungsbild und die Interaktion der Benutzeroberfläche einer App. Es ist kein Ersatz für „Cordova“, sondern eine gute Erweiterung.

Angular ist ein zugrundeliegendes Framework von Ionic. Es ist für die Komponenten-API verantwortlich, die der Grundbaustein von Ionic ist. [42]

##### 5.4.2.1 CLI

Zusätzlich gibt es ein „Ionic-CLI“ (Command Line Interface). Das CLI ist ein Tool, das eine Reihe hilfreicher Befehle für Entwickler von Ionic bereitstellt. Neben der Installation und Aktualisierung von Ionic verfügt die CLI über einen integrierten Entwicklungsserver, Build- und Debugging-Tools und vieles mehr. [42]

##### 5.4.2.2 Komponenten

Komponenten in Ionic sind wiederverwendbare Elemente der Benutzeroberfläche, die als Bausteine für die mobile App dienen. Komponenten bestehen aus HTML, SCSS und TypeScript. Jede Ionic-Komponente passt sich an die Plattform an, auf der die App ausgeführt wird. [42]

#### 5.4.2.3 Design

Ionic verwendet standardmäßig ein helles Theme, aber es gibt auch ein dunkles Theme. Viele weitere Themes können im Internet gekauft werden. [42]

#### 5.4.2.4 Geschichte

Ionic wurde ursprünglich von „@benjsperry“, „@adamdbradley“ und „@maxlynch“ entwickelt. Eine Alpha-Version wurde im November 2013 veröffentlicht. Im März 2014 wurde eine 1.0-Betaversion und im Mai 2015 eine 1.0-Endversion veröffentlicht. [42]

#### 5.4.2.5 Code-Dokumentation

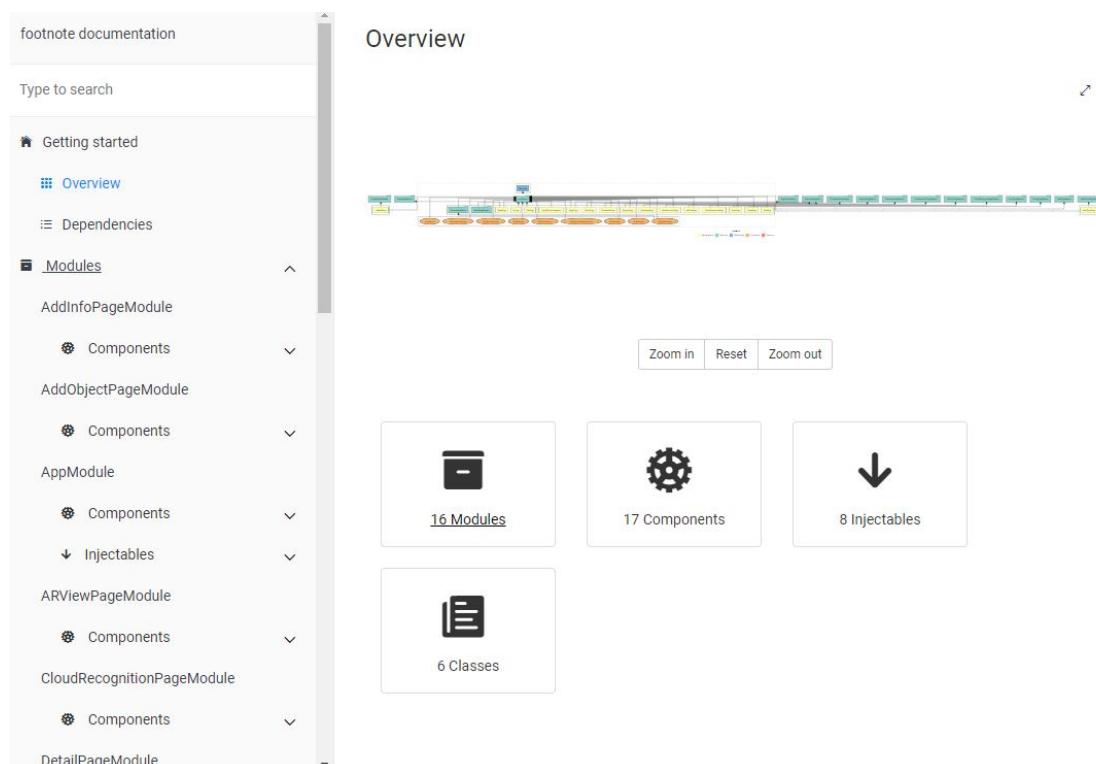


Abbildung 5.20: Screenshot: Compodoc

In Ionic gibt es die Möglichkeit Code mithilfe des „Compodoc-Plugins“ zu dokumentieren. Es schaut ähnlich wie eine „Javadoc“ aus. Dazu werden Kommentare in die TypeScript-Dateien geschrieben. Diese Kommentare werden erkannt und für die Dokumentation verwendet.

## 5.5 Verwendung von Ionic

### 5.5.1 Ionic installieren CLI

Um Ionic installieren zu können, muss zuvor Node.js installiert werden. Ionic kann mit folgenden Befehl installiert werden.

---

```
npm install -g ionic
```

---

Eine Ionic-App kann mit dem folgenden Befehl generiert werden.

---

```
ionic start myApp tabs
```

---

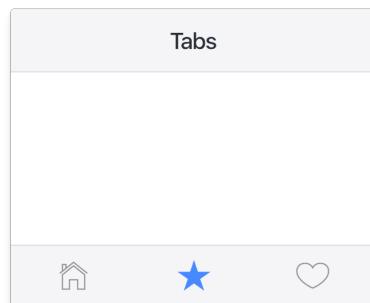


Abbildung 5.21: Ionic-Template Tabs

Dieser Befehl liefert eine App mit Tabs. Es gibt auch einen Befehl für eine leere App und eine App mit Seitenmenü.

---

```
ionic start myApp blank  
ionic start myApp sidemenu
```

---

Die App kann mit den folgenden Befehlen im Browser geöffnet werden: (Native Plugins funktionieren im Browser nicht.)

---

```
cd myApp  
ionic serve
```

---

Viele Ionic „Native-Plugins“ funktionieren nur auf einem Smartphone. Dazu kann die App mit dem folgenden Befehl auf dem Smartphone zum Laufen gebracht werden. Dazu muss das Smartphone über USB-Kabel mit dem Computer verbunden sein.

---

```
ionic cordova run android --device
```

---

[43]

### 5.5.2 Erstellen eines eigenen Icons und Splash-Screens

Wenn anstelle des Ionic-Logos ein eigenes Logo verwendet werden sollte, muss in den „resources“-Ordner das eigene Logo (Name: icon.png) und ein eigener Startbildschirm (Name: splash.png) gegeben werden. Das Logo sollte eine Abmessung von 1024x1024 px und der „splash“ eine Abmessung von 2732x2732 px haben. Die Auflösung sollte 150 dpi betragen.

Damit die beiden Bilder für alle Auflösungen zur Verfügung stehen, ist der folgende Befehl zu verwenden:

---

```
ionic cordova resources
```

---

Dadurch werden das Icon und der Splash in verschiedenen Auflösungen und Größen gespeichert.

### 5.5.3 HTTP-Provider

Am Android-Smartphone wurde der „CORS-Header“ durch Http-Aufrufe mit dem „HttpClientModule“ nicht erkannt. Mit dem „Ionic-Native-Http-Plugin“ funktionierte der „CORS-Header“ am Smartphone. Dieses Plugin funktioniert wiederum aber nicht auf Windows-Geräten. Deshalb wurde ein „HttpProvider“ erstellt, der bei „iOS“- oder „Android“-Geräten die Http-Aufrufe über das „Ionic-Native-Http-Plugin“ und sonst über das „HttpClientModule“ durchführt. [17] [16]

### 5.5.4 Standort erkennen

Damit Footnote die Objekte auch auf einer Karte anzeigen kann, müssen die Koordinaten gespeichert werden. In Ionic können die Koordinaten des Gerätes mit dem Ionic-Geolocation-Plugin ausgelesen werden. Dieses Plugin wird von allen für unser Projekt relevanten Plattformen (Android, iOS und Browser) unterstützt.

Das Plugin kann mit folgenden Befehlen installiert werden:

---

```
ionic cordova plugin add cordova-plugin-geolocation --variable
  GEOLOCATION_USAGE_DESCRIPTION="To locate you"
npm install --save @ionic-native/geolocation
```

---

Um das Plugin verwenden zu können, muss es in das app.module.ts eingetragen werden:

---

```
import { Geolocation } from '@ionic-native/geolocation';

@NgModule({
  ...
  providers: [
    Geolocation,
```

```
    ...
]
})
export class AppModule { }
```

---

Für iOS-Geräte muss folgende Konfiguration in das configuration.xml geschrieben werden:

```
<edit-config file="*-Info.plist" mode="merge"
  target="NSLocationWhenInUseUsageDescription">
  <string>We use your location for full functionality of certain app
  features.</string>
</edit-config>
```

---

Verwendung des Geolocation-Plugins in der Anwendung:

```
//Position auslesen
this.geolocation.getCurrentPosition().then((resp) => {
  var lat = resp.coords.latitude;
  var long = resp.coords.longitude;
}).catch((error) => {
  console.log('Error getting location', error);
});
```

---

[15]

### 5.5.5 Dateiupload

In Footnote muss ein Bild zu jedem Objekt hochgeladen werden, dieses Bild wird zum Erkennen des Objektes verwendet. Zusätzlich können noch Dateien zu Notizen hochgeladen werden. Diese Dateien können bei der Notiz angesehen werden.

Die Datei wird als DataURI auf den Server hochgeladen.

```
var reader = new FileReader();

reader.onload = (e) => {
  var dataURI = reader.result;
  console.log(dataURI);

  var url: string = this.http.url + "FileHandlerServlet?filename=" +
    file.name + "&" + typ + "=" + this.id;

  this.uploadFile(url, dataURI, params).subscribe(data => {
    this.http.presentToast("Bild auf Server hochgeladen");
  })
}
```

---

### 5.5.6 Bilder vom Server anzeigen

Um Bilder am Client anzuzeigen, hat sich das „IonicImageLoader“-Plugin als sehr hilfreich herausgestellt. Mit Hilfe dieses Plugins ist es möglich, Bilder im Hintergrund zu laden.

Dieses Plugin und das dazu benötigte „Ionic-File-Plugin“ können mit den folgenden Befehlen installiert werden:

---

```
npm install --save ionic-image-loader
npm i --save @ionic-native/file
ionic cordova plugin add cordova-plugin-file
```

---

Das Plugin muss ins app.module.ts eingetragen werden:

---

```
import { IonicImageLoader } from 'ionic-image-loader';

// import the module
@NgModule({
  ...
  imports: [
    ...
    IonicImageLoader.forRoot()
  ]
})
export class AppModule {}
```

---

Das Image kann in der HTML-Datei wie folgt verwendet werden:

---

```
<img-loader [src]="getImage(object.image)" useImg></img-loader>
```

---

Die dazugehörige Funktion in der ts-Datei:

---

```
getImage(image) {
  return "http://vm68.html-leonding.ac.at/footnote/UploadServlet?filename=" +
    image;
}
```

---

[14]

### 5.5.7 WYSIWYG-Editor

„WYSIWYG“ steht für „What You See Is What You Get“. Auf Deutsch übersetzt bedeutet das also „Was du siehst, ist [das], was du bekommst.“ Damit ist gemeint, dass der Text während dem Bearbeiten am Bildschirm genauso aussieht, wie bei der Ausgabe auf einem anderen Gerät (z.B. Drucker).

WYSIWYG wird sowohl von Programmierern in Editoren genutzt als auch von Redak-

teuren bei Redaktionssystemen und Content-Management-Systemen, um das Editieren ohne HTML-Kenntnisse zu ermöglichen. [36]



Abbildung 5.22: Screenshot: Quill-Editor in Footnote

Für Footnote stellte sich „Quill“ als geeigneter „WYSIWYG“-Editor heraus, da „Quill“ mit Angular und somit auch mit Ionic kompatibel ist.

Quill speichert Texteingaben als HTML-Code, deshalb ist es besonders einfach die Eingaben anzuzeigen. [33]

Installieren:

---

```
npm install ngx-quill
```

---

Einbinden von CSS und JS in index.html:

---

```
<!-- Include stylesheet -->
<link href="https://cdn.quilljs.com/1.3.6/quill.snow.css" rel="stylesheet">

<!-- Include the Quill library -->
<script src="https://cdn.quilljs.com/1.3.6/quill.js"></script>
```

---

Importieren des QuillModules in das app.module.ts:

---

```
import { QuillModule } from 'ngx-quill'
```

---

Hinzufügen des QuillModule in das app.module.ts:

---

```
import { QuillModule } from 'ngx-quill';
```

---

```
@NgModule({
  imports: [
    ...,
    QuillModule
  ],
  ...
})
```

---

Verwenden des Quill-Editor mit Bidirektionalem-Binding in Angular:

```
<quill-editor [(ngModel)]="variable"></quill-editor>
```

---

[48]

### 5.5.8 App-Launcher

Um Vuforia verwenden zu können, wird ein „App-Launcher“ verwendet. Die Ionic-Anwendung öffnet die App, deren Paketname angegeben ist. In unseren Fall wird zusätzlich der „JWT-Token“ in die Android-Vuforia-App mitgegeben.

Erstellen des start\_app.js mit folgendem Inhalt im assets/js.

```
module.exports = {
  packageLaunch: function (appSchemeStr, jwt) {
    // launch app using action name (for Android devices)
    window.plugins.launcher.launch({
      actionName: appSchemeStr,
      extras: [{
        "name": "jwt",
        "value": jwt,
        "dataType": "String"
      }],
      successCallback: function (json) {
        console.log('App opened')
      },
      function () {
        console.log('Failed to open app')
      }
    });
  },
}
```

---

Importieren der start\_app.js in der Typescript-Datei, die Vuforia öffnen möchte.

```
import * as launcher from '../assets/js/start_app';
```

---

Funktion zum Öffnen der Vuforia Anwendung in der Typescript-Datei, die Vuforia öffnen möchte.

---

```
openVuforia() {  
    launcher.packageLaunch("com.vuforia.Objects.intent.action.Vuforia",  
        this.auth.getToken());  
}
```

---

[22]

## 5.6 Android

### 5.6.1 Android-Studio



Abbildung 5.23: Android-Studio Logo

Android Studio ist eine freie „IDE“ (Integrierte Entwicklungsumgebung) von „Google“ und die offizielle Entwicklungsumgebung für Android. [3]

#### 5.6.1.1 Funktionalität

Android-Studio ist

- ein flexibles „Gradle-basiertes“ Buildsystem.
- ein schneller und funktionsreicher Emulator.
- eine einheitliche Umgebung, in der für alle Android-Geräte entwickelt werden kann.

Android Studio unterstützt „instant Run“, um Änderungen an laufenden Apps zu übertragen, ohne einen neuen „APK“ zu erstellen.

Android-Studio umfasst

- Codevorlagen und GitHub-Integration.
- umfangreiche Testwerkzeuge und Frameworks.
- Lint-Tools zum Erfassen von Leistung, Benutzerfreundlichkeit, Versionskompatibilität und anderen Problemen.
- C ++ und NDK-Unterstützung.
- integrierte Unterstützung für die Google Cloud-Plattform zur einfachen Integration von Google Cloud Messaging und App Engine.

[19]

### 5.6.2 Remote Debuggen-Android

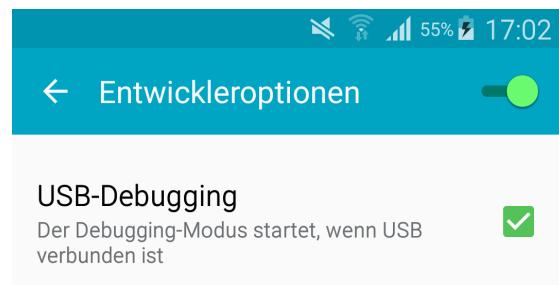


Abbildung 5.24: Screenshot: USB-Debugging erlauben

1. Um Remote Debuggen zu können, müssen als Erstes am Android-Gerät in den Einstellungen die Entwickleroptionen geöffnet und das USB-Debugging aktiviert werden.
2. Mit Chrome die Entwicklertools mit strg+Umschalt+I öffnen.
3. In den „DevTools“ auf das Hauptmenü klicken und unter „Weitere Tools“ Remote-Geräte auswählen.

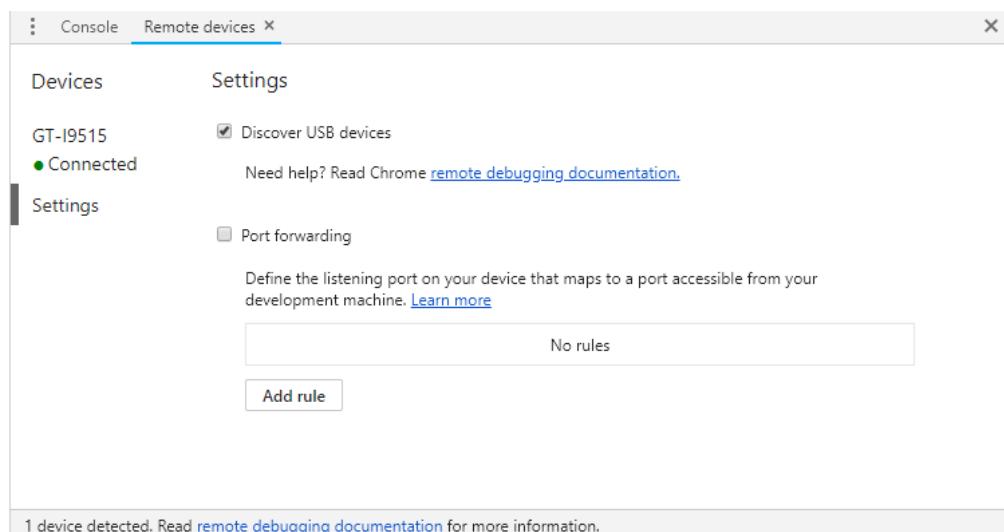


Abbildung 5.25: Screenshot: Kontrollkästchen USB-Geräte erkennen

4. In den Entwicklereinstellungen die Registerkarte „Einstellungen“ öffnen, um sicher zu stellen, dass das Kontrollkästchen USB-Geräte aktiviert ist.
5. Das Android-Smartphone mit einem USB-Kabel am Entwicklungscomputer anschließen.

6. Chrome auf dem Android-Gerät öffnen.

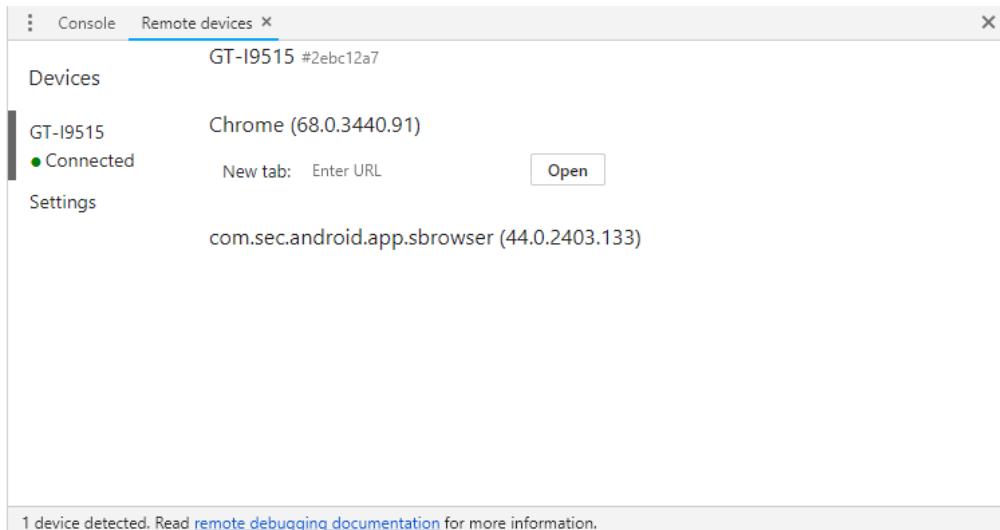


Abbildung 5.26: Screenshot: angeschlossenes Remote-Gerät

7. Unter Remote Devices auf den Modellnamen des angeschossenen Smartphones klicken. Jede geöffnete Chrome-Registerkarte am Smartphone wird angezeigt. Auch Apps, die „WebViews“ verwenden werden angezeigt.
8. Auf „inspect“ klicken um z.B. eine App mit „WebView“ in einem eigenen Entwicklungsfenster zu öffnen. In diesem Fenster kann der HTML-, CSS-, Javascript- und Typescript-Code debuggt werden.

[8]

### 5.6.3 Zugriff auf lokalen Server mit dem Android-Smartphone

#### 5.6.3.1 Portweiterleitung einrichten

Mithilfe der Portweiterleitung kann mit dem Android-Gerät auf Inhalte, die auf dem Webserver, der Entwicklungsmaschine gehostet werden, zugegriffen werden. Dazu muss auf dem Android-Gerät einen TCP-Port erstellt werden, der einem TCP-Port des Entwicklungscomputers zugeordnet wird. Über die USB-Verbindung zwischen dem Android-Gerät und dem Entwicklungscomputer erfolgt der Datentransfer. Die Verbindung ist unabhängig von der Netzwerkkonfiguration.

1. Um die Portweiterleitung einrichten zu können, muss zuerst das Remote-Debugging zwischen dem Entwicklungscomputer und dem Android-Gerät eingerichtet werden. Wenn alles richtig gemacht wurde, ist im linken Menü des Dialogfelds „Geräte prüfen“, das Android Gerät sichtbar.

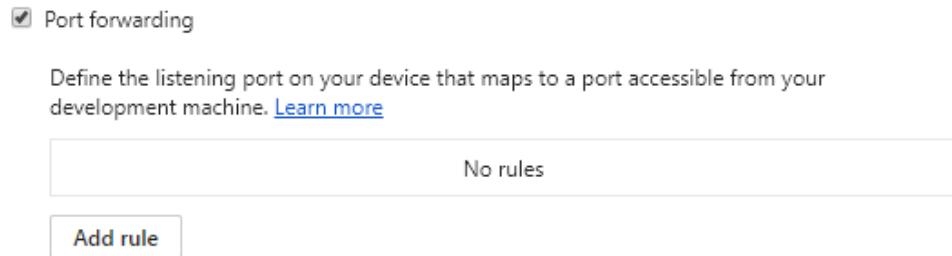


Abbildung 5.27: Screenshot: Port Forwarding

2. Im Dialogfeld „Geräte prüfen“ in den DevTools die Portweiterleitung aktivieren.
3. Auf „add rule“ klicken.
4. Im Textfeld „Device port“ auf der linken Seite die Localhost-Portnummer eingeben, von der aus auf das Android-Gerät auf den Webserver zugegriffen werden soll. Wenn beispielsweise von localhost: 8080 zugegriffen werden möchte, muss 8080 eingegeben sein.
5. Im Textfeld „Local adress“ auf der rechten Seite die IP-Adresse oder den Hostnamen samt Portnummer eingeben, unter der der Webserver auf dem Entwicklungsrechner ausgeführt wird. Wenn der Webserver beispielsweise auf localhost:8080 ausgeführt wird, gib localhost:8080 ein.
6. Auf Hinzufügen klicken.



Abbildung 5.28: Screenshot: Port Forwarding Status

Zum Testen, ob die Port-Weiterleitung funktioniert, wird Chrome auf dem Android-Gerät geöffnet und der localhost-Port, der im Feld Geräteport angegeben wurde, aufgerufen. Wenn beispielsweise 8080 in das Feld eingegeben wurde, wird zu localhost:8080 gewechselt. [1]

## 5.7 Verwendung von Wikitude

Die Objekterkennung von „Wikitude“, stellte sich nach testen mit unseren Testdaten als ungeeignet heraus. Doch die GPS-Funktion von Wikitude stellte sich als geeignet heraus. Footnote verwendet Wikitude um Objekte in der Nähe zu orten. Es werden auf der Kameraansicht Objekte platziert. Je nach Entfernung werden die Objekte unterschiedlich groß angezeigt. Wikitude funktioniert nur unter Android und iOS.

Es ist ein Ionic-Template für Wikitude vorhanden: (<https://github.com/pbreuss/wikitude-ionic-3-starter-app>). Um das Template verwenden zu können müssen die gewünschten Plattformen (Android, iOS) hinzugefügt werden und das Template ins src-Verzeichnis gegeben werden. Danach muss noch das Wikitude-Plugin installiert werden und der Lizenz-Key durch einen eigenen ersetzt werden.

---

```
ionic cordova plugin add
  https://github.com/Wikitude/wikitude-cordova-plugin.git
```

---

### 5.7.1 Lizenz

Für die Diplomarbeit wurde eine freie Education-Lizenz von Wikitude mit dem vollem Funktionsumfang verwendet. Um diese zu erhalten muss ein Formular von Wikitude ausgefüllt werden. In diesem muss der Android-Package-Name oder/und der iOS-Identifier angegeben werden. Ganz wichtig ist, dass das Projekt diesen Namen hat, sonst wird der Lizenz-Schlüssel nicht erkannt.

### 5.7.2 JWT

Um der Wikitude-Ansicht den JW-Token mitzugeben muss von Ionic aus ein „callJavaScript“ aufgerufen werden.

---

```
WikitudePlugin.callJavaScript("World.setToken('" +
  localStorage.getItem("token") + "')");
```

---

In der Architecture-World, die im Falle der Diplomarbeit im reloadContent.js ist, wird die Funktion angegeben, die von Ionic aufgerufen werden soll. Im Code-Beispiel wird gezeigt, wie eine Variable der Architecture-World gesetzt wird. Danach werden die Objekte mithilfe von reloadPlaces() geladen.

---

```
setToken(value) {
  World.jwttoken = value;
  World.reloadPlaces();
}
```

---

## 5.8 Verwendung von Vuforia

### 5.8.1 Einrichten der Android-Entwicklungsumgebung

Um mit Vuforia in Android arbeiten zu können, muss das Java SE Development Kit (JDK) und Android-Studio mit dem Android-SDK installiert sein. Um die Vuforia-Examples starten zu können, muss das Vuforia-SDK von der Seite <https://developer.vuforia.com/downloads/sdk> heruntergeladen werden. In den Ordner „Samples“, kann dann die eigene Vuforia-Anwendung gegeben werden. Um eine Basis für die eigene Vuforia-Anwendung zu haben, kann auf der Seite <https://developer.vuforia.com/downloads/samples> ein Beispiel-Projekt heruntergeladen werden. [9]



Abbildung 5.29: Vuforia Buch Beispiel

Das vuforia-samples-advanced-android-8-0-10.zip bietet Beispiele. In dieser Zip befindet sich das Book-Beispiel. Dieses kann drei Buchcover erkennen und eine Beschreibung dazu anzeigen. Dieses Beispiel arbeitet mit Cloud-Recognition und bot deshalb eine gute Basis für den Android-Client.

### 5.8.2 Erkennen eigener Objekte

Im Books.java gibt es einen „AccessKey“ und einen „SecretKey“. Diese beiden Keys müssen durch Keys für die eigene Vuforia-Anwendung ersetzt werden. Dazu muss unter <https://developer.vuforia.com/target-manager> ein Account erstellt werden. Für diesen Account muss eine Datenbank erstellt werden, um die Datenbank Access-Keys für den „Vuforia-Android-Client“ zu bekommen.

---

```
private static final String kAccessKey = "AccessKey";
private static final String kSecretKey = "SecretAccessKey";
```

---

In der „Target-Manager-Ansicht“ können die hochgeladenen Objekte angesehen werden, diese werden bei einem Status von „Active“ erkannt.

The screenshot shows the 'Target Manager' section of a software application. At the top, there are tabs for 'License Manager' and 'Target Manager', with 'Target Manager' being active. Below the tabs, the path 'Target Manager > Footnote\_Vuforia' is displayed. The main area is titled 'Footnote\_Vuforia' with a 'Edit Name' link. It shows the following details:

- Type:** Cloud
- License Key:** Footnote

Below this, there are two tabs: 'Targets (6)' (which is selected) and 'Database Access Keys'. A search bar at the top right says 'Search by target name or target ID'. There is also a 'Add Target' button.

Target Name	Rating	Recos	Status	Date Modified
31	★★★★★	0	Active	Mar 06, 2019 21:32
wolk	★★★★★	4	Active	Mar 06, 2019 20:00
gemaelde	★★★★★	0	Active	Mar 06, 2019 20:00
sv	★★★★★	0	Active	Mar 06, 2019 19:57
laptop	★★★★★	0	Active	Mar 06, 2019 19:54
drucker	★★★★★	0	Active	Mar 06, 2019 19:53

At the bottom left, it says 'Last updated: Today 09:34 PM' and has a 'Refresh' button.

Abbildung 5.30: Screenshot: Target-Manager

### 5.8.3 JWT

In Android kann der übergebene JW-Token mit folgenden Codezeilen ausgelesen werden.

---

```
Intent intent = getIntent();
jwt = intent.getStringExtra("jwt");
```

---

Wenn Vuforia ein Objekt erkennt, liefert es die Daten zurück, die beim Hochladen mitgegeben wurden. Im Fall von Footnote wird die „Id des Objektes“ mitgegeben. Um zu dieser Id, den Namen, die Beschreibung, die Koordinaten, die Notizen und das Bild zu erhalten, wird am eigenen Server eine Abfrage mit der Id gemacht. Dazu wird wieder der „JWT-Token“ im Header mitgegeben.

---

```
URL url = new URL(mObjectDataJSONFullUrl);
connection = (HttpURLConnection) url.openConnection();

connection.setRequestProperty("Accept-Charset", "UTF-8");
connection.setRequestProperty("Authorization", "Bearer " + jwt);
connection.connect();
```

---

# Kapitel 6

## Server

### 6.1 Anforderungen und deren Umsetzung

Footnote arbeitet mit „RealObjects“, dass sind Objekte, die in der Realität existieren, beispielsweise Plakate, Gemälde, oder auch Maschinen, Geräte und Gebäude. Zu diesen Objekten sollen Informationen abgespeichert werden - „Notes“. Der Server unterscheidet „Notes“ dabei nach ihrem Typ. Text, Links, Termine, Fotos und Dateien können gespeichert werden.

„RealObjects“ können gruppiert werden in „RealObjectFeeds“, deren Funktionalität einem Youtube-Abonnement, oder einem RSS-Feed ähnelt. Benutzer in diesem System werden hierbei in Usergruppen eingeteilt, welche jeweils bestimmten Feeds folgen.

#### Einsatzbeispiel:

Christopher Gусenbauer ist ein Benutzer des Systems, er ist in der Usergruppe Kunstabfans, diese folgt den RealObjectFeeds Renaissancekunst und Renaissancegebäude. Er sieht jetzt also die Notizen zu z.B. Gebäuden dieser Zeit in seiner Umgebung, die andere User hochgeladen haben, oder Informationen über Gemälde in Museen zur Renaissance, wie z.B. ihren Maler, ihr Entstehungsdatum und ihre Entstehungsgeschichte. Der Benutzer kann selber neue Gemälde zum System hinzufügen, Informationen zu diesen speichern und mit anderen Benutzern teilen.

Der Server muss sichergehen, dass Benutzer nur die „RealObjects“ angezeigt bekommen, die sie sehen wollen und dürfen. Es stehen Restschnittstellen für zwei verschiedene Ansichten zur Verfügung:

- Standardansicht: Alle zum Benutzer gehörigen Objekte werden angezeigt
- Hierarchieansicht: Der Benutzer kann sich durch die verschiedenen Ebenen(Usergruppen, Feeds, RealObjects) navigieren, bis er zu den Notizen kommt.

Dies wird mit Rollen und Nutzergruppen realisiert. Technisch geschützt ist die Kommunikation mit „JW-Tokens“ über „HTTPs“.

Die Kommunikation mit dem Vuforia AR-Framework - wie z.B. das Hochladen und erkennbar machen, sowie das Löschen von Gegenständen der echten Welt wird vom Server erledigt, des Weiteren bietet er File-Upload & -Download Funktionen an. Zur besseren Dokumentation existiert eine „Swagger-UI“ Weboberfläche, in der „REST-API“-Methoden beschrieben sind und getestet werden können.

## 6.2 Das Java EE Projekt

Der Server setzt sich aus vielen verschiedenen Klassen und Packages zusammen:

### 6.2.1 Entitäten

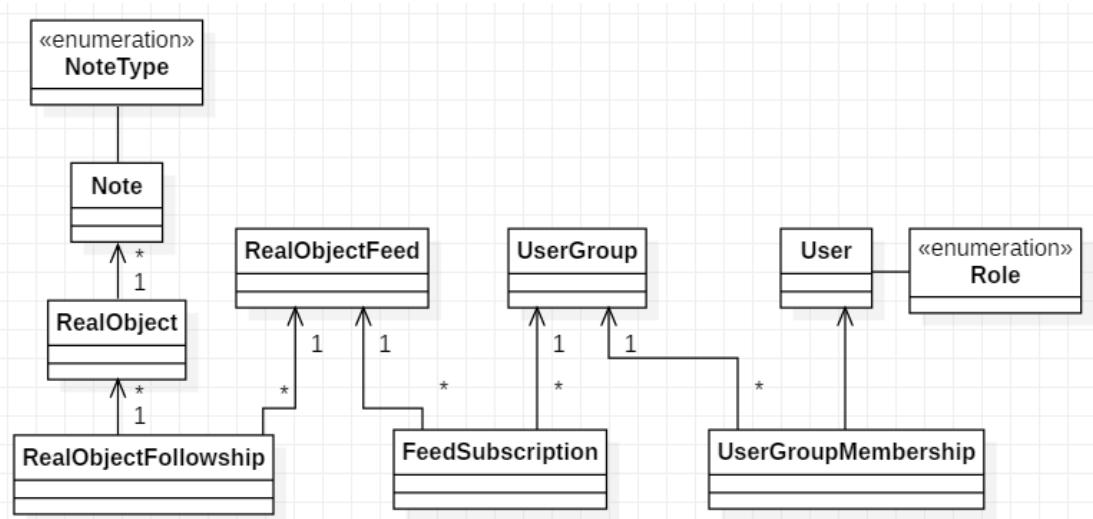


Abbildung 6.1: In der Datenbank gespeicherte Entitäten (ERD)

In der Datenbank gibt es mehrere Hilfsklassen, wie „FeedSubscription“, „RealObjectFollowship“ oder „UserGroupMembership“, deren Sinn eine Vereinfachung der Beziehungsstruktur ist, da sie die Verknüpfung von tatsächlichen Entitäten sind, welche selber keine eigenen Beziehungen haben (Außnahme RealObject). Diese werden benötigt, um in der Datenbank nicht zu viele Inhalte bei Informationsanfragen des Servers auszulesen, sowie um diese Inhalte zu minimieren und vereinfachen. Zu Benutzern wird außerdem eine zugehörige Rolle, zu Notizen der Typ der Notiz gespeichert.

### 6.2.2 Services und Repositories

Das Projekt ist geteilt in Services (REST-Endpoints), welche mit Clients kommunizieren und Repositories, die die logische Verwaltung der Entitäten auf der Datenbank und Businesslogik beinhalten. Zwecks einer besseren Systemarchitektur und einer Minimierung des Wartungsaufwands wurden mithilfe von „Java Generics“ abstrakte „Boilerplateklassen“ erstellt, „AbstractFacade“ für die Repositories und „EntityService“ für Services, auf welchen alle Repositories und Services aufbauen, die sich mit Entitäten beschäftigen. Diese stellen zumindest folgende Grundfunktionalität für ihre Entität zur Verfügung:

- Find: Objekt mithilfe seines Primärschlüssels finden.
- FindAll: Alle Objekte finden. Unterstützt Pagination.
- Update: Aktualisiert den gespeicherten Stand eines Objektes.
- Remove: Löscht ein Objekt mithilfe seines Primärschlüssels.
- Count: Ermittelt die Anzahl der gespeicherten Objekte.

In der „Service“ Klasse werden Funktionen zum Testen des Servers bereitgestellt, „AuthService“ beschäftigt sich mit dem Authentifizierungsprozess, also dem Nachweis (Verifizierung) einer behaupteten Eigenschaft (claim) einer Entität, die beispielsweise ein Mensch, ein Gerät, ein Dokument oder eine Information sein kann. Außerdem kümmert sie sich um den Authorisierungsprozess (Was darf wer aufrufen). Im Package „rest-TransferClasses“ befinden sich „PoJOs“ (Plain old Java Objects), also simple Java Objekte, die nicht auf irgendwelchen Frameworks aufbauen, die zur vereinfachten Rest-Kommunikation existieren (also hin- und hergeschickt werden).

### 6.2.3 Sicherheit

Sämtliche oben gezeigte Klassen kümmern sich um die Serversicherheit, „RestCommunicationFilter“ filtert Anfragen an den Server und hängt „CORS-Header“ an Antworten des Servers an, „RequiresAuthorization“ und „Status“ sind selbst geschriebene Annotations. „RequestSecurityContext“ liefert Informationen über Berechtigungen oder Eigenschaften des aufrufenden Benutzers an die Endpunkte. Genauer erklärt wird das in den Kapiteln 6.5.1 und 6.5.2.1.

### 6.2.4 Kommunikation mit Vuforia

Das Vuforiaframework arbeitet mit sogenannten „Targets“, welche einem zu erkennenden Bild entsprechen. Diese „Targets“ werden vollkommen getrennt von Footnotes „RealObjects“ gespeichert, um sicherzustellen, dass keine Userdaten auf fremden Servern landen. Die wichtigsten Klassen sind „PostNewTarget“, welche einen „Rest-Request“ an die Vuforiaserver mitsamt zugehörigen Bild absetzt und so ein neues „Target“ erstellt, sowie „DeleteTarget“, welche ein „Target“ löscht. Es ist allerdings immer mit einer geringen Zeitverzögerung von ca. einer Minute zu rechnen, da Vuforia Objekte nicht sofort auf

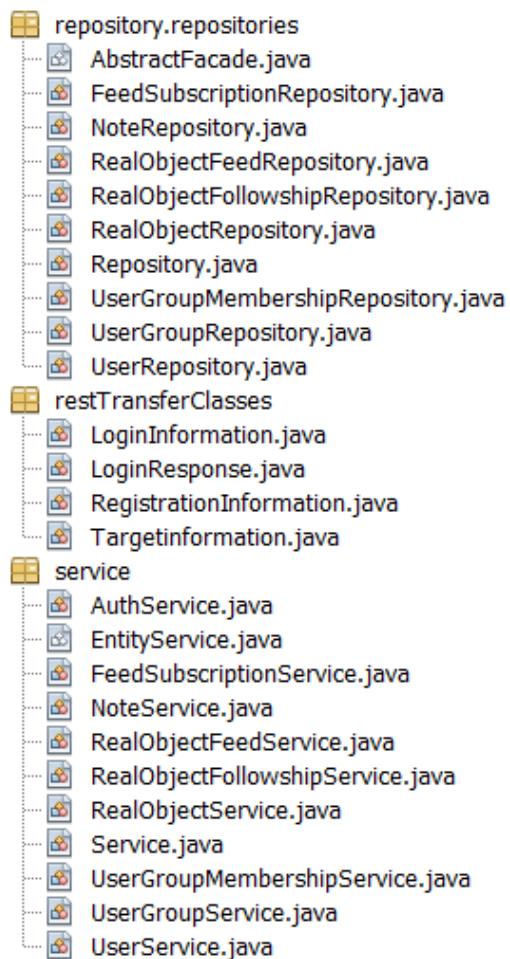


Abbildung 6.2: Screenshot: Die Rest Endpunkte und deren Repositories

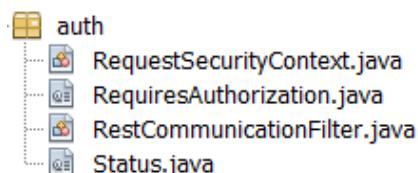


Abbildung 6.3: Screenshot: Struktur des Auth Package

allen verfügbaren Servern erstellt, ändert oder löscht. Für Java gibt es keine offizielle Dokumentation dieses Prozesses vonseiten Vuforias (Sämtliche Dokus sind offline), weswegen ein „Best-Practices“-Beispiel nach zahlreichen Versuchen erfolgreich für Footnote ummodelliert wurde.

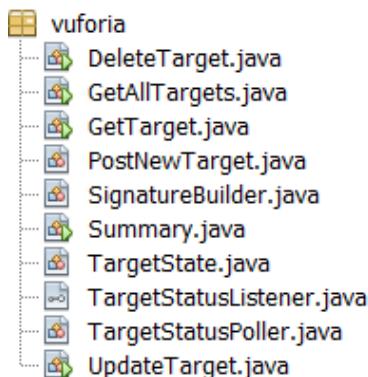


Abbildung 6.4: Screenshot: Klassen zur Erfassung und Löschung von Objekten

### 6.2.5 Sonstige Logik



Abbildung 6.5: Screenshot: Struktur des „logic“-Pakets

„JAXActivator“ ist eine Klasse, die „Application“ „extended“ mit @Application-Path(“rest“) annotiert ist, was einen REST-Endpunkt im Pfad /rest registriert.

LogFactory kümmert sich um das Logging in Footnote, hierfür wird Apache Log4j benutzt. Die Klasse sieht so aus:

---

```
public class LogFactory {
    @Produces
    Logger createLogger(InjectionPoint injectionPoint) {
        String name = injectionPoint.getMember().getDeclaringClass().getName();
        return Logger.getLogger(name);
    }
}
```

---

Via

---

```
@Inject
Logger log;
```

---

kann auf den Logger zugegriffen werden. Es gibt dabei verschiedene Prioritätslevel von Nachrichten, wie „log.warn“, „log.info“ und „log.debug“.

StartupHandler kümmert sich um den Initialisierungsprozess des Servers beim Starten - also darum Testobjekte zu erstellen. „StartupHandler“ ist mit der Annotation „@ApplicationScoped“ versehen. Um Code direkt nach dem Deployment auszuführen benutzt Footnote folgenden Code:

---

```
private void init(@Observes @Initialized(ApplicationScoped.class) Object object) {
```

---

Um Code vor dem Stoppen des Servers auszuführen benutzt Footnote folgenden Code:

---

```
public void destroy(@Observes @Destroyed(ApplicationScoped.class) Object init)
```

---

Die Klasse „FileHandlerServlet“ kümmert sich um Fileupload und Filedownloads. Hierbei verknüpft sie hochgeladene Bilder mit Objekten und Notizen, damit sie später wiedergefunden werden können. Außerdem wird bei Bildern für Notizen ein Texterkennungsalgorithmus (Tesseract) angewandt und so der aus dem Bild gewonnene Text abgespeichert. Hierfür müssen am Server Trainingsdaten gespeichert sein, anhand deren Tesseract das dafür nötige Wissen erhält. Tesseract ist Open-Source und lokal betreibbar. Des Weiteren beinhaltet die Klasse extensive Logging, sowie Fehlerinformationen für aufrufende Clients.

## 6.3 Systemumgebung

Die Java Enterprise Anwendung läuft auf dem Applikationsserver „Wildfly“ in der Version 14.0.1.Final, als Datenbank wird MySQL in der Version 5.7.24-0ubuntu0.18.04.1 genutzt. Prinzipiell ist die Datenbank austauschbar, getestet wurde nur MySQL, als Betriebssystem für den Server wurde Linux sowie Windows getestet. Nach außen aufrufbar ist nur der NGINX Reverse-Proxy, welcher „Requests“ an die richtigen Ziele(Wildfly, Webapp(Ionic Client),..) weiterleitet. Nachdem auf einem Schulserver die Infrastruktur lokal installiert wurde, sind für einen leichteren Installationsprozess mehrere „Docker“-Container geschrieben worden. Diese installieren unseren Java-EE Server, sowie das benötigte Production Environment automatisch.

### 6.3.1 Java

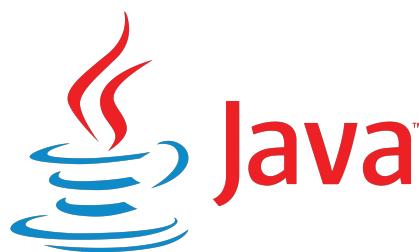


Abbildung 6.6: Das Java Logo

### 6.3.1.1 Geschichte

Das originale Java-Oak (Object Application Kernel) wurde zwischen 1991 und 1992 von Entwicklern des US-Computerherstellers Sun Microsystems entwickelt, hauptsächlich von James Gosling. Das Ziel: Nicht nur die Entwicklung einer neuen Programmiersprache, sondern einer vollständigen Betriebssystemumgebung, inklusive „CPU“, für verschiedene mögliche Zwecke. Java wird später in den Browser „Netscape Navigator“ integriert. Oracle plante ursprünglich Java in Richtung freie und quelloffene Software weiterzuentwickeln. Mittlerweile sieht es aber so aus, als ob Oracle Java möglicherweise teilweise kommerzialisieren will. [39]

### 6.3.1.2 Folgende Vorteile werden Java zugeschrieben

- ist robust
- unterstützt multi-threading
- ist zukunftssicher
- ist objektorientiert
- hat eine automatische Speicher- und Heap-Verwaltung
- kennt keine Pointers
- ist dynamisch, was das Runtime-System angeht. Klassen werden dann gelinkt, wenn sie benötigt werden.
- wurde entwickelt, Anwendungen in Netzwerken zu unterstützen, neue Module können über das Netzwerk implementiert werden
- ist plattformunabhängig, denn Java-Code wird zu Bytecode umgewandelt, welcher lediglich einen virtuellen Java-Processor benötigt, der den Bytecode interpretiert und ausführt. Diese Java Virtual Machine ist in nur ca. 200 KB Code zu implementieren. D.h., Geräte, die diese 200 KB JVM integriert haben, können Java-Programme ausführen.
- stellt standardmäßig eine Bibliothek für einheitliche Grafikfunktionen von verschiedenen Windows-Systemen (Windows, X-Windows etc.) zur Verfügung (JavaFX).

### 6.3.1.3 Folgende Nachteile werden Java zugeschrieben

- Der größte Vorteil ist auch gleichzeitig ein Nachteil: die Performance ist verringert, da Java-Code vom Interpreter (der die Plattformunabhängigkeit erst möglich macht) interpretiert wird. Momentan sind Java-Applikationen etwas langsamer als vergleichbare Anwendungen, die in C++ geschrieben wurden
- Keine Spracherweiterungen

- Mögliche Kommerzialisierung der Sprache aufgrund von strategischen Entscheidungen Oracles

[30]

### 6.3.2 Apache NetBeans

„Netbeans“, eine kostenlose, in Java geschriebene Entwicklungsumgebung eignet sich sehr gut zur Entwicklung in eben jener Sprache. Das von Studenten entwickelte und später von Sun Microsystems gekaufte Entwickler-Tool NetBeans ist eine Opensource-Alternative zu teuren Entwicklungs-Umgebungen. Praktischerweise sind auch gleich zwei Server-Umgebungen mit „GlassFish“ und „Apache Tomcat“ integriert, für Wildfly und Payara gibt es Plugins. Das früher nur für Java-Programmierung entwickelte Tool kann mittlerweile dank seiner modularen Architektur durch Plug-ins um viele Funktionen und weitere Programmier-Sprachen erweitert werden. So werden neben JavaSE, EE, ME und C/C++ auch Ruby, Rails, PHP, Groovy und Grails unterstützt.

Diese Informationen und der Downloadlink des Programms sind auf der „chip.de“ Website verfügbar: [https://www\(chip.de/downloads/Apache-NetBeans\\_29282799.html](https://www(chip.de/downloads/Apache-NetBeans_29282799.html)

## 6.4 Application Server im Vergleich

Ein Anwendungsserver, englisch Application Server, ist im Allgemeinen ein Server in einem Computernetzwerk, der Anwendungsprogramme ausführt. Im engeren Sinne bezeichnet der Begriff eine Software, die spezielle Dienste zur Verfügung stellt, wie beispielsweise Transaktionen, Authentifizierung oder den Zugriff auf Verzeichnisdienste, Webservices und Datenbanken über definierte Schnittstellen. In Bezug mit Java steht die Bezeichnung Application Server meist für ein Rahmenwerk, das die Entwicklung von Webanwendungen erleichtert und einen Server bereitstellt. [38]

Viele Einzelpersonen und Unternehmen stehen bei ihrem ersten größeren Javaprojekt nun vor der Entscheidung: Welchen Application Server benutzen? Der aktuelle Trend in Java geht stark Richtung „Open Source“ Lösungen, oft mit einem Unternehmen im Hintergrund, das in einer kostenpflichtigen Version Enterprise Support anbietet.

### Gründe, die für Open Source sprechen:

- Die meisten Opensource Lösungen sind nicht kostenpflichtig
- Viele talentierte Menschen arbeiten an Open Source, deswegen sind Projekte mit einer großen Gemeinschaft im Hintergrund meistens robust und voller Features.
- Anpassbarkeit an die Bedürfnisse des Unternehmens
- Keine Abhängigkeit von einem Unternehmen, dass finanzielle Gewinnabsichten hegt und pleite gehen kann.

- Test & Kontrollmöglichkeit der Software

Die Entscheidung ist im Endeffekt stark individuell, deswegen vergleicht diese Diplomarbeit nicht, sondern zeigt die Vorteile der einzelnen Server. [21] [5] [37]

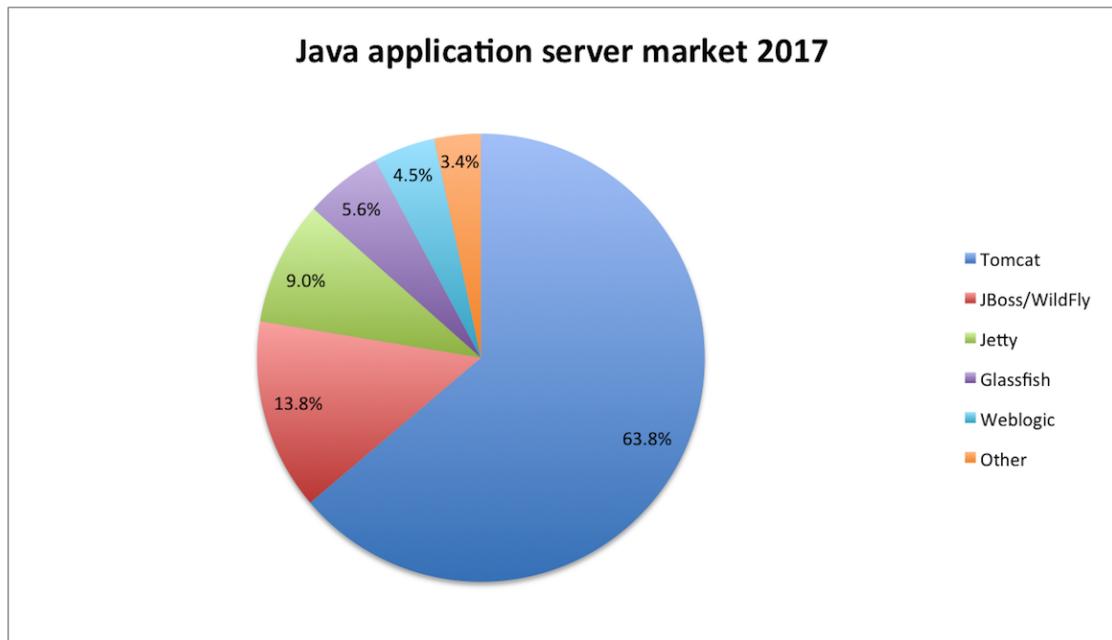


Abbildung 6.7: Vergleich der Marktanteile der Applikationsserver 2017

#### 6.4.1 Wildfly/Jboss EAP



Abbildung 6.8: Das Wildfly Logo

Unterstützt von dem beliebten Unternehmen „Red Hat“, welches unter anderem für seine Linux Distribution bekannt ist, wird Wildfly mit der Community entwickelt. Red Hat’s „JBoss Enterprise Application Platform“ (JBoss EAP) ist dabei die von Red Hat unterstützte kostenpflichtige Version von Wildfly.

#### Eigenschaften von Wildfly:

- Unterstützt von Red Hat, einer Open Source Firma
- Konform mit der Java<sup>TM</sup> EE 8 Spezifikation

- Cloud Ready, Container & Web Services Stack supported.
- Weitreichende Konfigurationsmöglichkeiten & Automatisierung
- Relativ komplexer Einrichtungsprozess
- Red Hat wurde gerade von IBM aufgekauft, weswegen die Zukunft von Wildfly unsicher ist, im Worst Case würde die Community wahrscheinlich alleine weiterentwickeln.
- Guter Clustering Support

Lizenzkosten: JBoss EAP kostet, um in einem Production Environment benutzt werden zu können, jährlich 8000 Dollar, bzw. im Premium Abonnement 12000 Dollar, der Vorteil letzterer ist 24/7 Support für „Severity 1 and 2“ Probleme. Die Standardlizenz enthält Support während den normalen Geschäftszeiten. [35]

#### 6.4.2 Glassfish

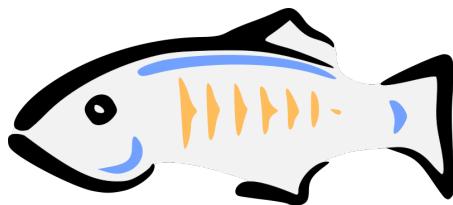


Abbildung 6.9: Das Glassfish Logo

Das Open Source Projekt „Glassfish“ wurde früher von Sun Microsystems und später von Oracle unterstützt, mittlerweile entwickelt es aber nur mehr die trotzdem starke Community weiter. [10]

#### Vorteile von Glassfish:

- Unterstützt Enterprise JavaBeans, JPA, JavaServer Faces, JMS, RMI, JavaServer Pages, servlets und mehr
- Plattform für die Entwicklung portabler, skalierbarer Java-Enterprise-Applikationen
- Kleiner Fußabdruck(lightweight), also geringer benötigter Festplattenspeicher[41]
- Einfacheres Aufsetzen als Wildfly, ermöglicht z.B. in Netbeans Remote Deploying, außerdem weniger genau in der Kontrolle korrekter Programmierpraktiken, beispielsweise bei Datenbankverbindungen(Proxies)

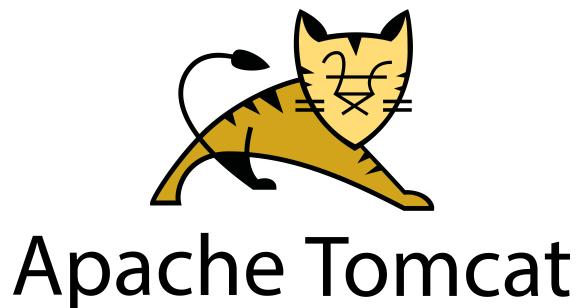


Abbildung 6.10: Das Tomcat Logo

#### 6.4.3 Tomcat

„Apache Tomcat“, das Ergebnis der Zusammenarbeit vieler talentierter Entwickler auf der ganzen Welt, ist eine Open Source Implementierung einiger Java Technologien. Tomcat wird von Wildfly als „Servlet-Container“ benutzt, unterstützt aber nicht selber den Java EE Application Stack. Tomcat hat keine intuitive Administration/Konfiguration im Vergleich zu anderen Konkurrenten, des Weiteren sind Loggingfeatures kaum vorhanden, dafür ist er aber performanter und lightweight, außerdem einfach zu deployen. [4]

#### 6.4.4 Payara Server



Abbildung 6.11: Das Payara Logo

Der „Payara“ Server stammt vom „Glassfish“ Server ab, bietet aber 24/7 Production und Development Support. an.[11] Der Server ist abgesichert und für Production Environments optimiert. Zukunftspläne sind Ausbau der Datenbankfeatures, bessere Diagnoseoptionen und mehr. Payara hat keinerlei Verbindungen zu Oracle oder Sun Microsystems. Payara ist und bleibt Open Source, da die Firma, die dahinter steht, eine Non-Profit-Organisation ist. Payara soll Glassfish ersetzen.

#### 6.4.5 Rechercheergebnisse

Footnote benützt aufgrund der vollen Unterstützung des Java Enterprise Standards und einer großen Nutzerzahl (gute Dokumentation, Fragen sind leicht recherchierbar), sowie einer langjährigen stabilen Entwicklung Wildfly. Tomcat eignet sich für kompakte, simple oder kleinere Projekte, weil eben nicht die komplette Java-EE Spezifikation unterstützt wird. Glassfish ist eigentlich eine Musterimplementierung des Java-EE Standards zu Demonstrationszwecken, weswegen Glassfish nicht für den Realbetrieb bestimmt ist, hier eignet sich Payara als Alternative, da eine Firma dahintersteht und Support anbietet.

### 6.5 JW-Tokens

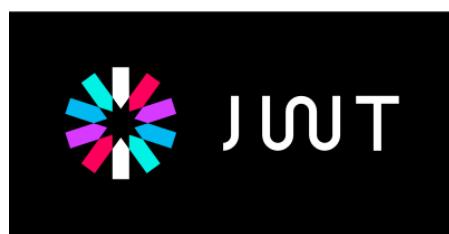


Abbildung 6.12: Das JWT Logo

#### 6.5.1 Was sind JW-Tokens

„JSON Web Tokens“ sind laut RFC 7519 eine in sich geschlossene, kompakte Art des Informationstransfers zwischen mehreren Parteien. Da diese digital signiert sind, sind sie verifiziert und vertrauenswürdig – wobei die Signierung mit einem „Secret“ durchgeführt wird: „HMAC“ (Hash based Message Authentication Code), oder alternativ ein „Public/Private Key“-Verfahren wie beispielsweise „RSA“ oder „ECDSA“. [18] Der Fokus von JW-Tokens – trotz der Möglichkeit sie zu verschlüsseln – liegt darin, Tokens zu signieren, was ermöglicht, dass man die Integrität ihrer Informationen, „Claims“, verifizieren kann, sie können also nicht manipuliert werden. Werden diese verschlüsselt, können sie zusätzlich noch vor den Augen unbeteiligter Parteien versteckt werden.

#### 6.5.2 Anwendungsfälle von JW-Tokens

##### 6.5.2.1 Authentifizierung, Autorisierung und Informationsaustausch

Sobald ein User eingeloggt ist, hat er im JWT Verfahren einen persönlichen Token zugewiesen bekommen. Dieser Token wird dann im Fall einer Client/Server-Architektur vom Client bei einem HTTP-Request (Anfrage) an den Server mitgeschickt – was dem User Zugriff auf die Methoden und Ressourcen gibt, die für seinen Token erreichbar sind. Informationen darüber, worauf der Nutzer zugreifen darf, werden dabei in dem integritätsgesicherten Claims-Bereich des Tokens gespeichert, den der Client selber nicht

unbemerkt manipulieren kann. Die Signierung der Tokens bewirkt also, dass man sicher sein kann, dass der Sender ist, wer er behauptet zu sein und seine Informationen („Header“) und Inhalt („Payload“) nicht verändert wurden.

### 6.5.3 Strukturierung

#### 6.5.3.1 Header

Der Header enthält üblicherweise zwei Teile: Den Hashing Algorithmus, der benutzt wird – HMAC, SHA256, RSA – und den Typ des Tokens, also JWT.

---

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

---

#### 6.5.3.2 Payload

In diesem Teil befinden sich die Claims – diese können beispielsweise eine, oder Teile einer Entität sein, z.B. ein User. In Footnote gibt es beispielsweise die Claims „Rolle“ und „BenutzerID“, wobei mit letzterer mehr Informationen über den User aus der Datenbank geholt werden können. Es gibt fix definierte, wie „Issuer“, „Ablaufdatum“, „Thema“ und mehr, die zwar nicht vorgeschrieben sind, aber in jedem Token enthalten sein sollten. Andere (Public & Private Claims) kann der User nach seinem Willen definieren.

---

```
{  
  "sub": "9898567890",  
  "name": "Christopher Gusenbauer",  
  "Rolle": "Moderator"  
}
```

---

Die Payload wird via „Base64Url“ encodiert.

#### 6.5.3.3 Signatur

Die Signatur wird aus dem encodierten Header erstellt, der encodierten Payload, einem Secret und mit dem definierten Algorithmus signiert.

### 6.5.4 Praxisbeispiel Tokengenerierung Java

In Java gibt es eine nützliche Bibliothek für die Interaktion mit JW-Tokens – hier die Maven Dependency:

---

```
<dependency>  
  <groupId>io.jsonwebtoken</groupId>  
  <artifactId>jjwt</artifactId>  
  <version>0.9.0</version>
```

```
<type>jar</type>
</dependency>
```

---

Mit dieser kann man nun so einfach einen JW-Token generieren:

```
String jwt = Jwts.builder().setSubject("Login_Auth_Footnote")
    .setId(String.valueOf(b.getId()))
    .setExpiration(Repository.getDateAfterDays(2))
    .claim("id", b.getId()).claim("role",
        b.getRole()).signWith(SignatureAlgorithm.HS256,
        RestCommunicationFilter.getSigningKey().getBytes("UTF-8"))
    .compact();
```

---

Der „Registered Claim Subject“ wird gesetzt, genauso eine „TokenID“, als eigene Informationen hängt Footnote noch die BenutzerID und die Rolle des Users an, signiert wird dann mit „HS256“ und dem im RestCommunicationFilter definierten „Secret“. Unter anderem dieser Code wird in Footnote bei einem Login ausgeführt. Kennt man Secret und Token funktioniert das Parsen dann so:

```
Jwts.parser()
    .setSigningKey(SIGNINGKEY.getBytes("UTF-8"))
    .parseClaimsJws(jwt);
```

---

## 6.6 Tesseract



Abbildung 6.13: Logo der Google-Tesseract Zusammenarbeit

Tesseract ist eine „OCR Engine“, sowie ein „Command Line Interface“ (CLI). Das bedeutet, dass es Text in Bildern erkennen kann. Die Engine basiert auf verschiedenen Neuralen Netzwerken (LSTM), die beispielsweise Linien oder Zeichenmuster erkennen. Damit Tesseract im Realbetrieb funktioniert, benötigt es „traineddata files“, also Dateien, die dem Netzwerk zeigen, welcher Buchstabe wie aussieht. Tesseract unterstützt standardmäßig über 100 Sprachen und bietet viele Ausgabeformate wie simplen Text, HTML oder auch PDF.

Es wurde ursprünglich zwischen 1985 und 1994 von Hewlett-Packard (HP) entwickelt und 2005 „open source“ gestellt. Seit 2006 wird es als offene, freie Software unterstützt von Google weiterentwickelt.

### 6.6.1 Tess4J

Um Tesseract in Java zu verwenden, nützt Footnote den Java-Wrapper „Tess4J“, welcher Support für TIFF, JPEG, GIF, PNG und BMP Bilder anbietet. Nachdem die Umgebung und die Trainingsdaten für Tesseract bereit sind, ist es relativ leicht ein Bild zu scannen:

---

```
public class TesseractExample {
    public static void main(String[] args) {
        File imageFile = new File("eurotext.tif");
        Tesseract instance = new Tesseract();
        instance.setDatapath("tessdata"); //Pfad zu den Trainingsdaten
        try {
            String result = instance.doOCR(imageFile);
            System.out.println(result);
        } catch (TesseractException e) {
            System.err.println(e.getMessage());
        }
    }
}
```

---

## 6.7 Logging



Abbildung 6.14: Logo Apache Log4J

Während man in Java theoretisch Informationen nur über „System.out.Print()“ ausgeben kann, macht es in größeren Projekten mehr Sinn ein Logging Framework wie „Log4j“ zu nützen. Dies ermöglicht es, Nachrichten dauerhaft zu speichern und an verschiedene Kanäle weiterzugeben. Außerdem unterstützt Log4J verschiedene Prioritätslevel für auszugebende Inhalte, was im Realbetrieb das Ausblenden unwichtiger Informationen möglich macht. Die genaue Umsetzung wird im Kapitel 6.2.5 beschrieben. Footnote verwendet aus folgenden Gründen Log4J:

- Support für „Messages“, nicht nur „Strings“
- Support für „Lambda-Expressions“

## 6.8 Fehlerbehandlung

Auf Java EE Servern ist es wichtig, Fehler nicht nur auf dem Server zu loggen, sondern auch Informationen an Clients weiterzugeben. Hierfür gibt es in „JAX-RS“ verschiedene Möglichkeiten.

### 6.8.1 Exception Mapper

Einer der Wege, das zu tun sind „Exception Mapper“.

---

```
public class MapperA implements ExceptionMapper<ExceptionA> {
    @Override
    public Response toResponse(ExceptionA ex) {
        return Response.status(400)
            .entity("Mapper A: " + ex.getMessage())
            .build();
    }
}
```

---

Obriger Code lässt bereits die Funktionsweise so eines Mappers erkennen. Wird eine bestimmte Exception geworfen, schaltet sich der Exceptionmapper ein und gibt dem Client eine Fehlermeldung mit detaillierten Informationen über das Problem zurück.

### 6.8.2 Jax-RS WebapplicationException

Footnote benützt allerdings in JAX-RS eingebaute „RuntimeExceptions“: „WebApplicationExceptions“, welche automatisch ein Response mit Fehlercode an den Client zurückschicken, sobald sie geworfen wurden. Für Standardfälle gibt es hierfür mehrere Exceptions: [46]

## 6.9 Authentifizierung, Autorisierung & Benutzer

Eine von Footnotes Entitäten ist User. In diese Klasse werden Informationen wie Vorname, Nachname, Email, Telefonnummer, Rolle, letzter Aufruf, Rolle und Passwortinformationen abgespeichert. Ein User kann sich nun wie folgt identifizieren (einloggen): Am Client gibt der User E-Mail & Passwort ein, worauf der Server via einem „REST-POST-Request“ diese Informationen gesendet bekommt. Am Server wird daraufhin geprüft, ob das Passwort korrekt ist.

### 6.9.1 Passwort-Hashing

Der Ablauf von Passwort-Hashen sieht in Footnote so aus:

Das Passwort wird vor dem Speichern mit einem userspezifischen Salt gehasht und erst dann in der Datenbank gespeichert.

Exception	Status code	Description
BadRequestException	400	Malformed message
NotAuthorizedException	401	Authentication failure
ForbiddenException	403	Not permitted to access
NotFoundException	404	Couldn't find resource
NotAllowedException	405	HTTP method not supported
NotAcceptableException	406	Client media type requested not supported
NotSupportedException	415	Client posted media type not supported
InternalServerErrorException	500	General server error
ServiceUnavailableException	503	Server is temporarily unavailable or busy

Abbildung 6.15: Bereits existierende Implementationen, die auf der Klasse WebApplicationException aufbauen

Will sich ein Systembenutzer nun mit E-Mail und Passwort einloggen, wird das Passwort – unbedingt über eine verschlüsselte Verbindung – an den Server geschickt, dort mit dem angehängten Salt gehasht und mit dem für den User in der Datenbank gespeicherten Hash verglichen.

Stimmen die Hashes überein, wird ein JW-Token generiert und an den Nutzer zurückgesendet. Wenn nicht, erhält er eine Fehlermeldung.

#### 6.9.1.1 Beispiel in Java:

---

```

public Boolean isPasswordCorrect(String pass) throws
    UnsupportedEncodingException, NoSuchAlgorithmException {
    String passwithsalt = pass.concat(passwordsalt);
    MessageDigest md = MessageDigest.getInstance("MD5");
    md.update(passwithsalt.getBytes());
    byte[] digest = md.digest();
    String myHash = DatatypeConverter.printHexBinary(digest).toUpperCase();

    if (myHash.equals(this.passwordhash)) {
        return true;
    }
    return false;
}

```

---

Der Client besitzt nun also seinen ganz persönlichen Token, in dem Informationen über ihn abgespeichert sind. Will er nun einen REST-Request machen oder auf eine Ressource zugreifen, muss er seinen Token mitschicken. In der Praxis wird der Token in den

Authorization Header des HTTP-Requests gepackt. Am Server wird der Request nun in seine Einzelteile zerlegt und der JW-Token geparsst. Mittels einer Annotation wird nun geprüft, ob die Rest-Methode, die aufgerufen wird, Aufrufer mit der Rolle des Benutzers akzeptiert. Wie das praktisch in Java funktioniert, wird auf den nächsten Seiten erklärt.

### 6.9.2 Registrierungsprozess

Ähnlich wie bei anderen Webservices wird beim Registrierungsprozess das „Double-Opt-in-Verfahren“ benutzt. Das Double-Opt-in-Verfahren ist zweistufig aufgebaut. Zuerst registriert sich der User in der App mit Daten wie seiner Email-Adresse und einem Passwort. Der Server verschickt dann, sofern die Registrierung erfolgreich war, eine Bestätigungs-E-Mail an die von dem Interessenten angegebene E-Mail-Adresse. Diese beeinhaltet eine URL, die auf eine bestimmte REST-Methode verweist. Dieser Link ist dabei nach folgendem Schema aufgebaut:

*https://vm68.htl-leonding.ac.at/javaendpoint/footnote/  
rest/user/verifyUser/9WF0SW42P30043WJU8*

Pro User wird ein zufallsgenerierter String an die URL angehängt - hiermit wird vermieden, dass User durch das Erraten eines nicht zufallsgenerierten Strings aktiviert werden können. Dieser String wurde bei der Registrierung erstellt und in einem neuen User Objekt gespeichert. Dieses User Objekt ist standardmäßig auf deaktiviert gesetzt, was bedeutet, dass sämtliche REST-Methoden gesperrt sind, der User also noch keine Berechtigungen hat. Sobald er diesen Rest-Link anklickt werden ihm dann die benötigten Privilegien gewährt.

#### 6.9.2.1 E-Mails senden mit Java Mail

Ein in Java EE standardmäßiges Tool zum Email verschicken ist „Java Mail“. Einer der großen Vorteile von diesem ist es, dass Java Mail mit Transaktionen arbeitet, also Emails erst dann verschickt, wenn der Server sozusagen „fertig“ ist, bzw. in einem Fehlerfall keine Email verschickt wird. Hierfür wurde ein „Google Mail Account“ (Gmail) erstellt und konfiguriert - es muss der Zugang für weniger sichere Apps erlaubt werden. Die benützte Funktion „sendFromGmail“ würde theoretisch auch für z.B. Newsletter oder systemweite Meldungen benützbar sein, da sie an mehrere Empfänger E-Mails senden kann.

---

```
public void sendFromGMail(String from, String pass, String[] to, String  
    subject, String body) {  
    Properties props = System.getProperties();  
    String host = "smtp.gmail.com";  
    props.put("mail.smtp.starttls.enable", "true");  
    props.put("mail.smtp.host", host);  
    props.put("mail.smtp.user", from);  
    props.put("mail.smtp.password", pass);
```

```

props.put("mail.smtp.port", "587");
props.put("mail.smtp.auth", "true");
Session session = Session.getDefaultInstance(props);
MimeMessage message = new MimeMessage(session);
try {
    message.setFrom(new InternetAddress(from));
    InternetAddress[] toAddress = new InternetAddress[to.length];

    // E-Mail Adressen Array erstellen
    for (int i = 0; i < to.length; i++) {
        toAddress[i] = new InternetAddress(to[i]);
    }

    for (int i = 0; i < toAddress.length; i++) {
        message.addRecipient(Message.RecipientType.TO, toAddress[i]);
    }

    message.setSubject(subject);
    message.setText(body);
    Transport transport = session.getTransport("smtp");
    transport.connect(host, from, pass);
    transport.sendMessage(message, message.getAllRecipients());
    transport.close();
} catch (AddressException ae) {
    log.warn(ae);
} catch (MessagingException me) {
    log.warn(me);
}
}

```

---

### 6.9.3 ContainerRequestFilter, ContainerResponsefilter

Klassen, die diese Interfaces in Java implementieren, müssen eine Filter Methode umsetzen. Diese hat unterschiedliche Aufgaben: Im Falle eines „ContainerResponseFilters“ werden beispielsweise an den Header einer Serverantwort auf einen Request „Access-Control“ Informationen angehängt, der „ContainerResponseFilter“ wird also aufgerufen, wenn der Server auf eine Nachricht antwortet und kann die Antwort ändern und ergänzen. Im Falle einer Headermanipulation für „Access-Control“ um das „CORS-Problem“ zu beheben, welches auftritt, falls ein Server mehrere Ports benutzt, bzw. an einen anderen Port weiterleitet. Häufiger benutzt wird der „ContainerRequestFilter“, er wird aufgerufen, wenn der Server eine Nachricht empfängt („Request“) und kann die Informationen des Requests und dessen Inhalt analysieren und verändern. Ein Filter muss mit der „@Provider Annotation“ versehen werden:

---

```

@RequiresAuthorization
@Provider
@Priority(Priorities.AUTHORIZATION)

```

```
public class RestCommunicationFilter implements ContainerRequestFilter,  
    ContainerResponseFilter {
```

---

In Footnote sieht die Filter-Funktion so aus:

```
public void filter(ContainerRequestContext requestContext)  
throws IOException {  
    //Aus dem Request die Headers auslesen  
    MultivaluedMap<String, String> headers = requestContext.getHeaders();  
    //Den Auth Header auslesen  
    String authorization = requestContext.getHeaderString("Authorization");  
    //Gibt es einen Token im Bearer , JWT benutzt Bearer, Format ?  
    if (authorization != null && authorization.startsWith("Bearer")) {  
        //Token für die JWT Bibliothek vorbereiten  
        authorization = authorization.substring("Bearer".length()).trim();  
        authorization = authorization.replace("\\", "");  
        Jws<Claims> credentials;  
        try {  
            //JWT Libary zum Decoden benutzen  
            credentials = decodeJWT(authorization);  
        } catch (IllegalArgumentException e) {  
            credentials = null;  
        }  
    }  
}
```

---

## 6.9.4 Berechtigungen

### 6.9.4.1 Annotation

Soll Java EE davon ausgehen, dass gewisse Klassen und/oder Methoden zugriffsgeschützt sind, kann das, wie in Footnote, mit der „RequiresAuthentication“-Klasse, welche eine eigene Annotation definiert, die für Authorization benutzt wird, realisiert werden.

```
@NameBinding  
@Retention(RUNTIME)  
@Target({TYPE, METHOD})  
public @interface RequiresAuthorization {  
    Role[] value() default {};  
}
```

---

Zum Schützen von Methoden wird nun in z.B. dem REST-Endpoint eine Liste der akzeptierten Rollen mitgegeben. *Rolle* ist dabei als „ENUM-Entität“ gespeichert. Die Annotation kann ganze Klassen, aber auch nur Methoden sichern.

```
@RequiresAuthorization({Role.ADMINISTRATOR, Role.GUEST, Role.MODERATOR,  
    Role.USER})
```

---

#### 6.9.4.2 Rollen

Natürlich muss im Filter auch kontrolliert werden, dass nur die richtigen Rollen Zugriff haben. Zunächst muss der Filter wissen, welche Klasse/Funktion welche Rollen zulässt. Für Klassen funktioniert das folgendermaßen, wobei „resourceInfo“ mit „@Context“ injected worden ist:

---

```
Class<?> resourceClass = resourceInfo.getResourceClass();
List<Role> classRoles = extractRoles(resourceClass);
```

---

Für Methoden so:

---

```
Method resourceMethod = resourceInfo.getResourceMethod();
List<Role> methodRoles = extractRoles(resourceMethod);
```

---

„extractRoles“ sieht so aus:

---

```
private List<Role> extractRoles(AnnotatedElement annotatedElement) {
    if (annotatedElement == null) {
        return new ArrayList<>();
    } else {
        RequiresAuthorization secured =
            annotatedElement.getAnnotation(RequiresAuthorization.class);
        if (secured == null) {
            return new ArrayList<>();
        } else {
            Role[] allowedRoles = secured.value();
            return Arrays.asList(allowedRoles);
        }
    }
}
```

---

Abschließend muss die Rollenliste(n) mit der Rolle des Users verglichen werden, die am besten als Claim im JW-Token mitgegeben wird, oder (schlechter) aus der Datenbank ausgelesen wird.

#### 6.9.4.3 SecurityContext

Was ist aber, wenn innerhalb der „Java EE“-Application Informationen über den Aufrüfer ausgelesen werden möchten? Java stellt hierfür das „SecurityContext“ Interface zur Verfügung. Standardmäßig enthält eine implementierende Klasse z.B. eine „istUserInRolle“ Funktion, es können aber auch selbst Methoden und Attribute, wie beispielsweise eine UserID oder ein User Object angehängt werden.

Das ermöglicht es:

- je nach Rolle unterschiedlichen Code auszuführen.
- Userinformationen auszulesen, bzw. den User identifizieren zu können, ohne Infos wie z.B. seine ID im Pfad der Rest-Methode mitgeben zu müssen.

Erstellt wird ein „SecurityContext“ im „ContainerRequestFilter“. Wie ist dabei stark Applicationserver abhängig und nicht genormt. In Java SE ist es scheinbar nicht möglich, diesen „SecurityContext“ mit dem „REST-Service“ zu verknüpfen, Java EE ermöglicht dies mit Hilfsklassen der Applikationsserver. Da Footnote „Wildfly“ benutzt folgt die Implementation für Wildfly:

---

```
requestContext.setSecurityContext(new RequestSecurityContext(id, role));
ResteasyProviderFactory.pushContext(RequestSecurityContext.class,new
    RequestSecurityContext(id, role));
```

---

„RequestSecurityContext“ implementiert in diesem Beispiel das „SecurityContext“ Interface. Um diese Funktion benutzen zu können, muss „Injection“ verwendet werden.

---

```
@Context private RequestSecurityContext re;
```

---

## 6.10 FileHandlerServlet

Footnote benötigt verschiedenste Funktionen um Daten zu speichern und wieder zur Verfügung zu stellen. Hierbei werden Funktionalitäten angeboten um Files hochzuladen und mit Notizen oder „Realobjects“ zu verknüpfen. Werden Bilder mit Notizen verknüpft, so kann zusätzlich noch Texterkennung ausgeführt werden. Außerdem können Files heruntergeladen werden.<sup>1</sup>

### 6.10.1 FileUpload

Der FileUpload funktioniert mit Data-URIs, diese müssen erst zurechtgestutzt werden, bevor sie verwendet werden können. Sie werden also nach dem ersten Beistrich getrennt.

---

```
protected void processRequest(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
    log.info("File Upload started to" + BASEPATH);
    InputStream is = request.getInputStream();
    String symbol;
    do {
        byte[] buffer = new byte[1];
        is.read(buffer);
        symbol= new String(buffer);

    } while (!symbol.equals(",,"));

    int lenght = is.available();
    System.out.println(lenght);
}
```

---

<sup>1</sup>Da es sich um eine Diplomarbeitsumgebung handelt, wird nicht geprüft, welcher User welche Datei aufrufen darf, in einem realen System ist dies unabdingbar.

Der Name der Datei wird ausgelesen und die OCR-Bibliothek wird initialisiert und ein Stream zum Schreiben des Files wird geöffnet.

---

```
Tesseract tesseract = new Tesseract();
String filename = request.getParameter("filename");
tesseract.setDatapath(TESSERACTPATH);
//Bild am Server speichern
try (FileOutputStream fo = new FileOutputStream(new File(BASEPATH, filename)))
{
    boolean fileInformationSupplied = false;
```

---

Falls gewollt, wird die Datei mit einem „RealObject“ verknüpft.

---

```
try {
    long objectid = Long.valueOf(request.getParameter("realobjectid"));
    RealObject ro = em.find(RealObject.class, objectid);
    ro.setImage(filename);
    fileInformationSupplied = true;
} catch (Exception ex) {
    log.info("objectid not set" + ex.getLocalizedMessage());
}
```

---

Falls gewollt, wird die Datei mit einer Notiz verknüpft und Text - sofern es sich um ein unterstütztes Bildformat handelt - ausgelesen und gespeichert.

---

```
try {
    long objectid = Long.valueOf(request.getParameter("noteid"));
    Note ro = em.find(Note.class, objectid);
    ro.addFile(filename);
    fileInformationSupplied = true;
    String text;
    try {
        text = tesseract.doOCR(new File(BASEPATH, filename));
        log.debug(text);
        ro.setOCR(text);
    } catch (Exception ex) {
        if (Files.notExists(Paths.get(TESSERACTPATH))) {
            log.warn("OCR Directory doesn't exist, upload TestData");
        }
        log.info(ex.getLocalizedMessage());
    }
}
```

---

Abschließend werden noch mögliche Fehler behandelt:

---

```
} catch (Exception ex) {
    log.info("noteid not set" + ex.getLocalizedMessage());
}
if (!fileInformationSupplied) {
    response.sendError(HttpServletRequest.SC_BAD_REQUEST, "NoteID or
```

```

        RealObject id not included, include at least one");
    }
    IOUtils.copy(Base64.getDecoder().wrap(is), fo);
} catch (Exception e) {
    log.error(e.getLocalizedMessage());
    if (!Files.exists(Paths.get(BASEPATH))) {
        response.sendError(HttpServletRequest.SC_NOT_IMPLEMENTED, "Directory to
            upload files to doesnt exist on Server");
        log.warn("directory to upload images to doesnt exist");
    }
    response.sendError(HttpServletRequest.SC_INTERNAL_SERVER_ERROR, "Couldnt
        save File \n" + e.getLocalizedMessage());
}

```

---

Da die Ionic-Http-Bibliothek sehr eingeschränkte Funktionalität hat, wird ein ganz bestimmtes Response zusammengestellt:

```

response.setContentType("application/json");
response.setCharacterEncoding("UTF-8");
GeneralResponse res = new GeneralResponse("sucess");
try (PrintWriter out = response.getWriter()) {
    ObjectMapper objectMapper = new ObjectMapper();
    String jsonString = objectMapper.writeValueAsString(res);
    out.println(jsonString);
} catch (Exception ex) {
    log.warn(ex.getLocalizedMessage());
}
log.info("Fileupload done");
}

```

---

### 6.10.2 FileDownload

Das Herunterladen von Files ist relativ selbsterklärend. Aus den Request-Parametern wird der Name des Files ausgelesen, dannach wird kontrolliert, ob auf dem Server ein Ordner existiert, in dem Files gespeichert werden können, bzw. ob es den File mit diesem Namen überhaupt gibt.

```

public void doGet(HttpServletRequest request, HttpServletResponse response)
throws IOException, ServletException {
    log.info("Filedownload started");
    String filename = request.getParameter("filename");
    File my_file = new File(BASEPATH, filename);
    if (!my_file.exists()) {
        if (!Files.exists(Paths.get(BASEPATH))) {
            response.sendError(HttpServletRequest.SC_NOT_IMPLEMENTED, "directory
                to download images from doesnt exist");
            log.warn("directory to download images from doesnt exist");
        }
    }
}

```

```
        response.sendError(HttpServletRequest.SC_BAD_REQUEST, "File doesn't
                           exist");
    }
```

---

Jetzt wird der File stückweise in den OutputStream geschrieben, also an den Client gesendet:

```
OutputStream out = response.getOutputStream();

try (FileInputStream in = new FileInputStream(my_file)) {
    byte[] buffer = new byte[4096];
    int length;
    while ((length = in.read(buffer)) > 0) {
        out.write(buffer, 0, length);
    }
    in.close();
    out.flush();
}

log.info("Filedownload ended");
}
```

---

## 6.11 Vuforia Codebeispiele

Während es zwar praktisch keine Dokumentation für das Benützen des Vuforia-Frameworks gibt, ist die Interaktion meistens relativ verständlich. Sämtlichen Code zu erklären würde zur Unlesbarkeit führen, deswegen sind folgend nur die wichtigsten Teile erklärt, die zum Teil aus einem „Best-Practices“ Beispiel von Vuforia genommen und angepasst wurden. Um mit dem Framework kommunizieren zu können, müssen bestimmte individuelle Keys mitgegeben werden, die man sich als Entwickler oder Kunde auf der Vuforia-Website generieren lassen kann.

### 6.11.1 Target hochladen

Das Hochladen eines Targets auf Vuforia funktioniert über einen REST-Request, Vuforia weist hochgeladenen Objekten dann eine eigene ID zu, diese muss dann gespeichert werden.

```
private String postTarget(String filename) throws URISyntaxException,
    ClientProtocolException, IOException, JSONException {
    HttpPost postRequest = new HttpPost();
    HttpClient client = new DefaultHttpClient();
    postRequest.setURI(new URI(url + "/targets"));
    JSONObject requestBody = new JSONObject();

    setRequestBody(requestBody, filename);

    ...
```

```

postRequest.setEntity(new StringEntity(requestBody.toString()));
setHeaders(postRequest); // Must be done after setting the body

HttpResponse response = client.execute(postRequest);
String responseBody = EntityUtils.toString(response.getEntity());
System.out.println(responseBody);

JSONObject jobj = new JSONObject(responseBody);

String uniqueTargetId = jobj.has("target_id") ? jobj.getString("target_id")
    : "";
System.out.println("\nCreated target with id: " + uniqueTargetId);

return uniqueTargetId;
}

```

---

Einen Request kann man nun in Java z.B. so strukturieren:

```

private void setRequestBody(JSONObject requestBody, String filename) throws
    IOException, JSONException {

    File imageFile = new File(FileHandlerServlet.BASEPATH, filename);
    if (!imageFile.exists()) {
        System.out.println("File location does not exist!");
        System.exit(1);
    }
    byte[] image = FileUtils.readFileToByteArray(imageFile);
    requestBody.put("name", targetName); // Mandatory
    requestBody.put("width", 1); // Mandatory
    requestBody.put("image", Base64.encodeBase64String(image));
    requestBody.put("active_flag", 1); // Optional
    requestBody.put("application_metadata",
        Base64.encodeBase64String(metadata.getBytes())); // Optional
}

```

---

### 6.11.2 Target löschen

Um ein Target zu löschen muss man es erst deaktivieren, da Vuforia etwas Zeit braucht, bis sich Änderungen auf den Servern widerspiegeln:

```

public void deactivateThenDeleteTarget() {
    // Update the target's active_flag to false and then Delete the target when
    // the state change has been processed;
    try {
        updateTargetActivation(false);
    } catch (URISyntaxException | IOException | JSONException e) {
        e.printStackTrace();
        return;
    }
}

```

```

    }

    // Poll the target status until the active_flag is confirmed to be set to
    // false
    // The TargetState will be passed to the OnTargetStatusUpdate callback
    targetStatusPoller = new TargetStatusPoller(pollingIntervalMinutes,
        targetId, accessKey, secretKey, this);
    targetStatusPoller.startPolling();
}

```

---

Gelöscht wird es dann mit einem so programmierten „DELETE-Request“:

```

private void deleteTarget() throws URISyntaxException,
    ClientProtocolException, IOException {
    HttpDelete deleteRequest = new HttpDelete();
    HttpClient client = new DefaultHttpClient();
    deleteRequest.setURI(new URI(url + "/targets/" + targetId));
    setHeaders(deleteRequest);

    HttpResponse response = client.execute(deleteRequest);
    System.out.println("Delete Response " +
        EntityUtils.toString(response.getEntity()));
}

```

---

## 6.12 OpenAPI(Swagger)



Abbildung 6.16: Logo der Open API Initiative

Die „OpenAPI“-Spezifikation ist eine gemeinschaftsbasierte, offene Spezifikation, die eine standardisierte, sprachenunabhängige Schnittstellenbeschreibung für REST APIs definiert. Sie erlaubt Computern wie Menschen die Funktionalitäten von REST Services zu entdecken und zu verstehen. OpenAPI ermöglicht die Interaktion mit richtig implementierten Services mit minimalem Code-Aufwand. OpenAPI ist „opinionated“, d.h. es werden nicht alle möglichen Arten von „REST-API“ Stilen unterstützt.

Diese Spezifikation kann für folgende Anforderungen verwendet werden:

- Interaktive Dokumentation

# Footnote-Server Swagger-UI 1.0.1

[ Base URL: vm68.html-leonding.ac.at/javaendpoint/footnote/rest ]

Footnote combines real objects with digital information

Contact Christopher Gusenbauer

The screenshot shows the Footnote-Server Swagger-UI interface. At the top, there's a header with the title "Footnote-Server Swagger-UI 1.0.1", a base URL "[ Base URL: vm68.html-leonding.ac.at/javaendpoint/footnote/rest ]", and a note about Footnote combining real objects with digital information. Below this is a contact link for Christopher Gusenbauer.

On the left, there's a dropdown menu for "Schemes" with "HTTPS" selected. To the right is an "Authorize" button with a lock icon.

The main content area is divided into two sections:

- Authentication Rest Service**: This section has a green "POST" button for the endpoint "/auth/login" which returns a valid JWT token that expires after 2 days.
- FeedSubscription Rest Service**: This section has a blue "GET" button for the endpoint "/feedSubscription/{id}" which finds an entity with the sent key, and a red "DELETE" button for the same endpoint which deletes the entity.

Abbildung 6.17: Screenshot: Swagger

- Codegeneration von „Stubs“, also Programmiergrundgerüsten, für Clients und Server in vielen verschiedenen Programmiersprachen und die Automatisierung von Tests.
- Mit „Swagger-UI“ erhält man außerdem ein schönes Interface zum Aufrufen der REST Funktionen, kann sich so also Arbeiten mit Fiddler/Postman ersparen.

In Footnote wird Swagger hauptsächlich zur Dokumentation und zum Testen vom Server benutzt. Dabei wird die OpenAPI-Definition automatisch aus den Java-Klassen und Methoden generiert und dann eine Swagger-UI in Form einer Website generiert.

## 6.12.0.1 Apiee

Nach mehreren Versuchen, den oben angeführten Prozess einfach zu automatisieren, wurde eine Lösung gefunden: „Apiee“. Diese Bibliothek generiert anhand des Java-EE

Projekts eine Swagger-UI Website.[44]

Via „Maven“ geht das nach dem Hinzufügen einer einzigen Dependency:

```
<!-- Apiee -->
<dependency>
<groupId>com.github.phillip-kruger</groupId>
<artifactId>apiee-core</artifactId>
<version>1.0.8</version>
</dependency>
```

Die Dokumentation wird mithilfe von Annotations in Java generiert. Hierbei werden REST-Services mit „@Api(value = “Name des Services“)“ definiert, einzelne Methoden können mit „@ApiOperation(value = “Retrieve some example content“, notes = “This will return some json to the client“, response = JsonObject.class)“ beschrieben werden. Allgemeine Informationen werden in der JAX-RS Applikationsklasse via „@Swaggerdefinition“ beschrieben:

```
@ApplicationPath("/api")
@SwaggerDefinition (info = @Info (
    title = "Example Service",
    description = "A simple example of apiee",
    version = "1.0.0",
    contact = @Contact (
        name = "Phillip Kruger",
        email = "apiee@phillip-kruger.com",
        url = "http://phillip-kruger.com"
    )))
)
public class ApplicationConfig extends Application {
}
```

## 6.13 Designphilosophie am Server

Der Server ist in folgende Pakete geteilt:

- entities - die in der Datenbank zu speichernden Pakete
- repositories - Controller Klassen
- services - REST-Endpunkte
- logic - andere Logik
- restTransferClasses - Java Pojos

Code der sich nur mit der Erstellung, Wartung und Löschung von Entitäten beschäftigt basiert auf „Boilerplate Klassen“ - d.h. egal ob Note oder Userobject, die Grundfunktionalität bzw. die Services sind gleich implementiert. Es wird versucht, Paket-, Klassen-, Methoden- und Variablennamen möglichst selbsterklärend zu wählen, um die für das Verständnis notwendige Dokumentation zu minimieren. Codelänge wird möglichst minimiert.

## 6.14 Ubuntu: Wildfly aufsetzen und Datenbankverbindung hinzufügen



Abbildung 6.18: Logo Ubuntu

Eine große Hürde für Footnote war das Aufsetzen der Infrastruktur. Hierbei nutzt Footnote zum Entwickeln Windows - als Production Environment Ubuntu - mit einer MySQL Datenbank, auf die der Application Server Wildfly zugreift und einem NGINX Reverse-Proxy.

Versionen:

Wildfly: 14.0.1.Final

MySQL: 5.7.24-0ubuntu0.18.04.1

### 6.14.1 Voraussetzungen:

- Ein lauffähiger Ubuntu-Server mit MySQL Instanz (min. Version 18.04)
- Ein User mit „Sudo“-Privilegien.

Vorbereitung: Packages und Repositories aktualisieren:

---

```
sudo apt update -y && sudo apt upgrade -y
```

---

#### 6.14.1.1 Wildfly User & Gruppe Erstellen

---

```
groupadd -r wildfly
useradd -r -g wildfly -d /opt/wildfly -s /sbin/nologin wildfly
```

---

#### 6.14.1.2 Java installieren

---

```
sudo apt-get install default-jdk -y
```

---

Kontrollieren, ob das Ganze funktioniert hat, kann man mit dem Befehl „java -version“.

### 6.14.2 Wildfly installieren

---

```
wget http://download.jboss.org/wildfly/14.0.1.Final/wildfly-14.0.1.Final.tar.gz
$ mkdir /opt/wildfly
$ tar -xvzf wildfly-14.0.1.Final.tar.gz -C /opt/wildfly/
```

---

Ein lauffähiger Wildfly Application Server ist jetzt im Verzeichnis /opt/wildfly installiert. Dieser Anwendungsserver ist aber noch nicht konfiguriert. Zuallererst erstellen wir deswegen einen ManagementUser.

#### 6.14.2.1 User erstellen

Im „bin“-Verzeichnis von Wildfly findet sich das „add-user“-Script:

---

```
cd /opt/wildfly/bin
./add-user.sh
```

---

#### 6.14.2.2 Wildfly starten

---

```
cd /opt/wildfly/standalone
./standalone.sh
```

---

Wildfly startet nun. Ist dieser Vorgang abgeschlossen, kann man den Server unter 127.0.0.1:8080 erreichen, das Admin-Dashboard unter 127.0.0.1:9090. Der Server Log befindet sich im Verzeichnis „Standalone/Log/server.log“. Man kann sich nun im Admin-Dashboard mit dem vorher konfigurierten User einloggen. [12] [40] [13]

#### 6.14.2.3 Konfigurationsdatei erstellen

Ein Konfig-File erstellen:

---

```
nano /etc/default/wildfly
Inhalt:
WILDFLY_USER="wildfly"
STARTUP_WAIT=180
SHUTDOWN_WAIT=30
WILDFLY_CONFIG=standalone.xml
WILDFLY_MODE=standalone
WILDFLY_BIND=0.0.0.0
```

---

```

What type of user do you wish to add?
  a) Management User (mgmt-users.properties)
  b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : USERNAME
Password recommendations are listed below. To modify these restrictions edit the
  add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admi
n, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s),
  1 digit(s), 1 non-alphanumeric symbol(s)
Password :
WFLYDM0101: Password should have at least 1 digit.
Are you sure you want to use the password entered yes/no? y
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated
list, or leave blank for none)[ ]: wildfly
About to add user 'USERNAME' for realm 'ManagementRealm'
Is this correct yes/no? y
Added user 'USERNAME' to file '/opt/wildfly/wildfly-14.0.1.Final/standalone/conf
iguration/mgmt-users.properties'
Added user 'USERNAME' to file '/opt/wildfly/wildfly-14.0.1.Final/domain/configur
ation/mgmt-users.properties'
Added user 'USERNAME' with groups wildfly to file '/opt/wildfly/wildfly-14.0.1.F
inal/standalone/configuration/mgmt-groups.properties'
Added user 'USERNAME' with groups wildfly to file '/opt/wildfly/wildfly-14.0.1.F
inal/domain/configuration/mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS pr
ocess?
e.g. for a slave host controller connecting to the master or for a Remoting conn
ection for server to server EJB calls.
yes/no? y
To represent the user add the following to the server-identities definition <sec
ret value="UEFTU1dPUkQ=" />

```

Abbildung 6.19: Screenshot: Add-User Dialog

#### 6.14.2.4 Startup Script

Ein Skript wird benötigt, das den Server mit obigen Parametern startet:

---

```

nano /opt/wildfly/bin/launch.sh
#!/bin/sh

if [ "x\$WILDFLY_HOME" = "x" ]; then
WILDFLY_HOME=/opt/wildfly
fi

if [ "x\$1" = "xdomain" ]; then
echo 'Starting Wildfly in domain mode.'

```

```
\$WILDFLY_HOME/bin/domain.sh -c \$2 -b \$3
else
echo 'Starting Wildfly in standalone mode.'
\$WILDFLY_HOME/bin/standalone.sh -c \$2 -b \$3
fi
```

---

Schlussendlich muss das Skript ausführbar gemacht werden.

```
chmod 755 /opt/wildfly/bin/launch.sh
```

---

#### 6.14.2.5 Service für automatisches Starten und Stoppen konfigurieren

Um sicherzugehen, dass Wildfly auch nach Ausfällen und Serverneustarts noch läuft und um ihn einfach starten und stoppen zu können, bietet Linux Services an. Folgender ist für Wildfly geschrieben:

```
nano /etc/systemd/system/wildfly.service
[Unit]
Description=The WildFly Application Server
After=syslog.target network.target
Before=nginx.service

[Service]
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
EnvironmentFile=/etc/default/wildfly
User=wildfly
LimitNOFILE=102642
PIDFile=/var/run/wildfly/wildfly.pid
ExecStart=/opt/wildfly/bin/launch.sh \$WILDFLY_MODE \$WILDFLY_CONFIG
    \$WILDFLY_BIND
StandardOutput=null

[Install]
WantedBy=multi-user.target

chown wildfly:wildfly -R /opt/wildfly/
systemctl daemon-reload
systemctl start wildfly
systemctl enable wildfly
```

---

#### 6.14.3 Remotezugang

Möchte man nun, dass der Server extern erreichbar ist, hat man zwei Möglichkeiten:

#### 6.14.4 Möglichkeit 1: NGINX Reverse Proxy



Abbildung 6.20: NGINX Logo

##### 6.14.4.1 Was ist ein Reverse Proxy?

Ein Reverse-Proxy ist ein Server, der als Schnittstelle zwischen internen Applikationen und externen Clients steht und letztere zu den richtigen internen Servern weiterleitet. „NGINX“ bietet einige Funktionalitäten wie Load-Balancing, Sicherheitsfeatures und hohes Tempo an, die es dafür ideal machen.

##### 6.14.4.2 Vorbereitungen:

Falls es einen vorinstallierten, laufenden Apache gibt, muss dieser gestoppt und entfernt werden:

---

```
service apache2 stop
apt-get remove apache2
apt-get autoremove
apt-get install nginx
```

---

In das NGINX Verzeichnis wechseln:

---

```
nano /etc/nginx/sites-enabled/default
```

---

Folgende Zeile löschen:

---

```
listen [::]:80 default_server;
```

---

NGINX automatisch starten lassen:

---

```
systemctl enable nginx
systemctl start nginx.service
```

---

##### 6.14.4.3 Wildfly-Block erstellen:

NGINX leitet bestimmte Requests an z.B. Ports weiter, falls man das so wünscht. Das „Location“-Attribut im folgenden Beispiel leitet also alle Requests - wegen “/“ - weiter auf “http://wildfly“, was dem Port 8080 entspricht. Da dies ein Beispiel ist, müsste in der Realität „your-domain.com“ durch die eigene Domain ersetzt werden.

---

```
nano /etc/nginx/sites-available/wildfly
upstream wildfly {
    server 127.0.0.1:8080;
}

server {
    listen      80;
    server_name your-domain.com;

    access_log /var/log/nginx/wildfly.access.log;
    error_log  /var/log/nginx/wildfly.error.log;

    proxy_buffers 16 64k;
    proxy_buffer_size 128k;

    location / {
        proxy_pass http://wildfly;
        proxy_next_upstream error timeout invalid_header http_500 http_502
            http_503 http_504;
        proxy_redirect off;

        proxy_set_header Host          $host;
        proxy_set_header X-Real-IP     $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}

ln -s /etc/nginx/sites-available/wildfly /etc/nginx/sites-enabled/
systemctl restart nginx.service
```

---

#### 6.14.5 Möglichkeit 2: Config ändern

Wildflys Hauptkonfigurationsdatei nennt sich „standalone.xml“. In dieser muss die IP-Adresse auf die des tatsächlichen Servers geändert werden. Dazu muss die Datei im Terminal-Texteditor des Vertrauens geöffnet werden:

---

```
nano /opt/wildfly/standalone/configuration/standalone.xml>
```

---

Dort gehören dann sämtliche Einträge die 127.0.0.1 lauten durch die IP-Adresse des Servers ersetzt.

#### 6.14.6 Datenbankverbindung konfigurieren

Hierfür gibt es wieder zwei verschiedene Varianten, wobei aufgrund von schnellerer Erledigung nur auf die zweite eingegangen wird.

#### a) JBoss-CLI

Das „JBoss-CLI“ (Command Line Interface) ermöglicht es, Wildfly-Server zu verwalten, es kann sich also im Terminal ein User mit dem Server verbinden und Administrationsaufgaben erledigen. Mehr Informationen hierzu gibt es unter: <https://docs.jboss.org/author/display/WFLY/Command+Line+Interface>

#### b) Standalone.xml konfigurieren

Standardmäßig kommt Wildfly nur mit einem Treiber für die interne „H2-Datenbank“. Da aber mit MySQL gearbeitet wird, muss hierfür ein Treiber installiert werden.

Wildfly Extensions werden unter WILDFLY\_HOME/modules/system/layers/base gespeichert, wobei WILDFLY\_HOME dem Installationsverzeichnis von Wildfly entspricht.

Dort sollte jetzt der Ordnerbaum com/mysql/main erstellt werden. In diesem Ordner „main“ muss die „JDBC-Treiber-Datei“ und eine „module.xml“-Datei platziert werden. Die aktuelle JDBC Treiber Datei ist auf der MYSQL Homepage herunterzuladen. In die „module.xml“ Datei gehört folgender Inhalt:

---

```
<module xmlns="urn:jboss:module:1.5" name="com.mysql">
<resources>
<resource-root path="mysql-connector-java-8.0.12.jar" />
</resources>
<dependencies>
<module name="javax.api"/>
<module name="javax.transaction.api"/>
</dependencies>
</module>
```

---

Natürlich muss der Pfad des „mysql-connectors...“ an die Version angepasst werden. Nun informiert man Wildfly über diesen verfügbaren Treiber:

---

```
<driver name="mysql" module="com.mysql">
<driver-class>com.mysql.cj.jdbc.Driver</driver-class>
<xa-datasource-class>com.mysql.cj.jdbc.MysqlXADataSource</xa-datasource-class>
</driver>
```

---

Dieser Code muss in WILDFLY\_HOME/standalone/configuration/standalone.xml im „Drivers“-Block unter den Block des H2-Treibers eingefügt werden. [2]

#### 6.14.6.1 Datenbankverbindung hinzufügen

Nun öffnet sich ein Fenster, in dem der MySQL Treiber und die datenbankspezifische Verbindungs-URL ausgewählt werden muss. Das Schema für den JNDI-Namen sieht dabei so aus: java:/MySqlDS

Das Schema für die Verbindungs-URL sieht so aus: jdbc:mysql://localhost/footnote  
Nun sollte eine Verbindung erfolgreich abgeschlossen werden.

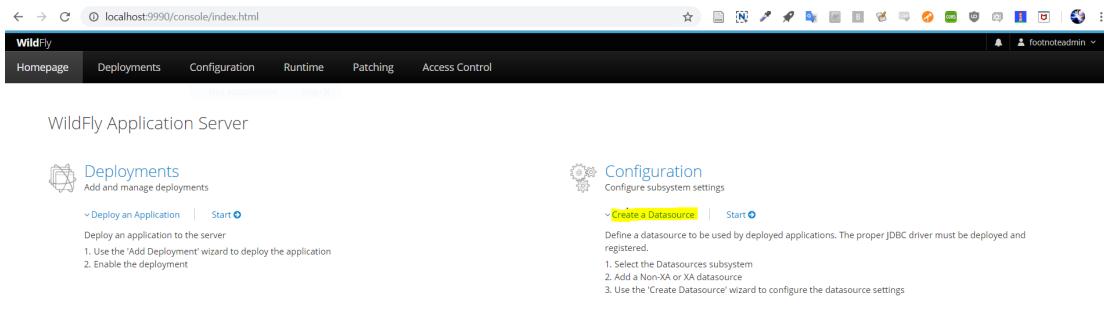


Abbildung 6.21: Screenshot: Im Admin-Interface in den Configuration Tab wechseln

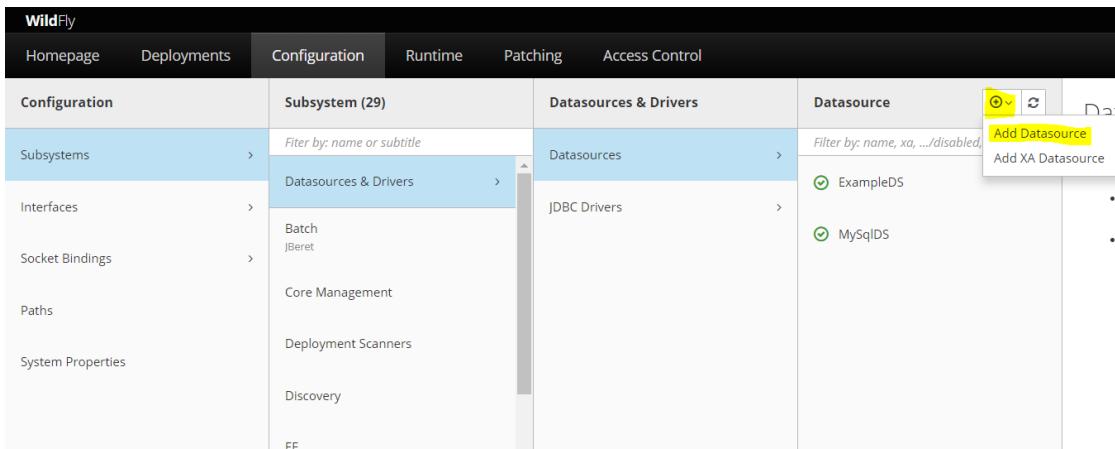


Abbildung 6.22: Screenshot: Eine Datasource hinzufügen

### 6.14.7 Deployment

Um ein Java EE Projekt auf den Server zu deployen, muss erst das Projekt „gebuildet“ werden. Netbeans (bzw. eigentlich Maven) erstellt beispielsweise im Projektverzeichnis den Ordner „Target“. In diesem findet sich nach dem Bauen eine „.war“ Datei, welche per Drag and Drop in den Deployment Dialog gezogen werden muss.

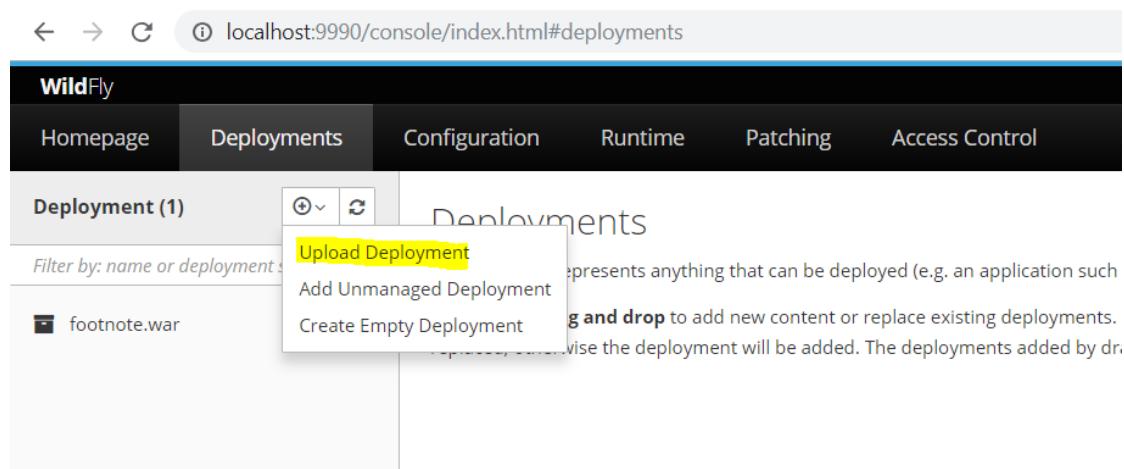


Abbildung 6.23: Screenshot: Deployment Dialog öffnen

# Kapitel 7

## Logo, Webseite, Plakat

### 7.1 Projektwebsite

#### 7.1.1 Motivation

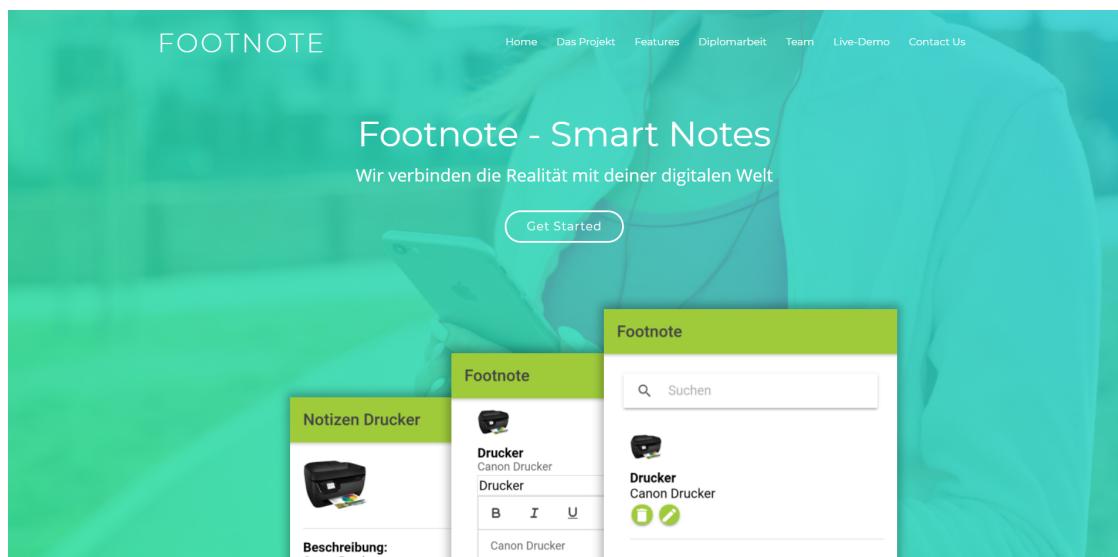


Abbildung 7.1: Screenshot: Projektwebsite-Startansicht

Neue spannende Diplomarbeiten werden oft nach ihrer Veröffentlichung schnell vergessen. Wir empfanden es als wichtig, über die Möglichkeiten unserer Anwendung zu informieren und diese auch für Alltagsanwender verständlich und spannend zu präsentieren, weswegen eine Website gestaltet wurde.

**FOOTNOTE**

Home Das Projekt Features Diplomarbeit Team Live-Demo Contact Us

**Funktionen des Produkts**

- GPS-Karte**  
Radarfunktion: Alle gespeicherten und geteilten Objekte in der Umgebung anzeigen.
- Augmented Reality**  
Informationen über die Welt live in der Kameraansicht angezeigt bekommen.
- Texterkennung(OCR)**  
Aus JPG-Bildern automatisch Text auslesen und am Server speichern.
- Verschiedene Notiztypen**  
Bilder, Text, Dokumente, Daten und mehr, bearbeitbar in einem komfortablen WYSIWYG-Editor

Abbildung 7.2: Screenshot: Webseite

**FOOTNOTE**

Home Das Projekt Features Diplomarbeit Team Live-Demo Contact Us

**Footnote verknüpft Objekte aus dem echten Leben mit der digitalen Welt.**

Beispielsweise können zu Plakaten, Gemälden, aber auch Druckern und Gebäuden digitale Informationen abgespeichert werden - "Notes".

- Erkennung von Objekten anhand ihres Aussehens
- GPS-Karte der nächsten Objekte
- Text in hochgeladenen Bildern wird automatisch erkannt und mitgespeichert

Der Server unterscheidet "Notes" dabei nach ihrem Typ. Text, Links, Termine, Fotos und Dateien können gespeichert werden. Objekte können gruppiert werden in "RealObjectFeeds", deren Funktionalität einem Youtube-Abohnment oder einem RSS-Feed ähnelt. Benutzer in diesem System werden hierbei in Usergruppen eingeteilt, welche jeweils bestimmten Feeds folgen können.

Abbildung 7.3: Screenshot: Webseite

### 7.1.2 Umsetzung

Der ausgewählte Dienst „Git-Pages“ ist ein statischer Webseiten-Hosting-Service, der es möglich macht, Projektwebseiten direkt mit Gitlab zu hosten. Git-Pages unterstützen keinen serverseitigen Code wie PHP, Ruby oder Python.

Die Entscheidung fiel darauf, die Website mit „Bootstrap“ umzusetzen. Bootstrap ist ein Open Source Framework, zum Entwickeln mit HTML, CSS und JavaScript. Für die Webseite wurde ein Bootstrap-Template verwendet, das wir uns auf unsere Bedürfnisse anpassten. Die Webseite ist mit der URL <https://chrisi.gusenbauer101.gitlab.io/Footnote/> aufrufbar. Die Webseite ist als „Single-Page-Webseite“ entworfen, wobei es verschiedene Sektionen gibt: Das Projekt, die Features, die Diplomarbeit, das Team, FAQ (Häufig gestellte Fragen), ein Link zur Live-Demo und einen Kontaktbereich.

## 7.2 Logo



Footnote

Abbildung 7.4: Footnote-Logo

Für den Startbildschirm von Footnote wurde ein Logo benötigt. Es wurde lange überlegt, was eine Augmented Reality Notizapp repräsentiert. Es wurden die Logos von anderen Notizapps analysiert. Sehr viele Notizapps haben Logos, die Tieren ähneln, deshalb fiel die Entscheidung dann auch auf ein Tier. Genauer gesagt wurde ein Affe ausgewählt, weil Affen clever sind und unsere App eine clevere Notizapp sein sollte.

Zusätzlich ist noch ein Schriftzug mit Logo entstanden. Das Logo und der Schriftzug sind mit Adobe Illustrator erstellt worden.



Abbildung 7.5: Footnote-Schriftzug

### **7.3 Plakat**

Eine Anforderung für den Tag der offenen Tür der HTL Leonding war es, ein Plakat zu erstellen. Ziel des Plakates ist es Leute für unser Projekt zu begeistern. Für uns war es wichtig, dass aus dem Plakat hervorkommt, was das Hauptziel von Footnote ist und wie im alltäglichem Leben von Footnote profitiert werden kann. Es sollten auch unbedingt unsere verwendeten Technologien auf dem Plakat ersichtlich sein. Das Plakat wurde mit Adobe „Illustrator“ designt.

# Footnote

Link your digital information to the real world



HTL Leonding 2018/19 | Diplomarbeit | Christopher Gusenbauer | Isabella Hundstorfer

Abbildung 7.6: Footnote-Plakat

# Literaturverzeichnis

- [1] Access local servers | tools for web developers. URL: <https://developers.google.com/web/tools/chrome-devtools/remote-debugging/local-server>.
- [2] Adding the MySQL JDBC driver into Wildfly | Synaptik Labs. URL: <https://synaptiklabs.com/posts/adding-the-mysql-jdbc-driver-into-wildfly/>.
- [3] Android studio. Page Version ID: 183213593. URL: [https://de.wikipedia.org/w/index.php?title=Android\\_Studio&oldid=183213593](https://de.wikipedia.org/w/index.php?title=Android_Studio&oldid=183213593).
- [4] Apache Tomcat® - Welcome! URL: <http://tomcat.apache.org/>.
- [5] Best Application Server Software in 2018. URL: <https://www.g2crowd.com/categories/application-server>.
- [6] Beste Notiz-App | Organisiere deine Notizen mit Evernote. URL: <https://evernote.com/intl/de>.
- [7] Erweiterte realität. Page Version ID: 182671128. URL: [https://de.wikipedia.org/w/index.php?title=Erweiterte\\_Realit%C3%A4t&oldid=182671128](https://de.wikipedia.org/w/index.php?title=Erweiterte_Realit%C3%A4t&oldid=182671128).
- [8] Get started with remote debugging android devices | tools for web developers. URL: <https://developers.google.com/web/tools/chrome-devtools/remote-debugging/>.
- [9] Getting started with vuforia for android development. URL: <https://library.vuforia.com/articles/Solution/Getting-Started-with-Vuforia-for-Android-Development.html>.
- [10] GlassFish. URL: <https://javaee.github.io/glassfish/>.
- [11] Home. URL: <https://www.payara.fish/>.
- [12] How to install WildFly (JBoss) Java Application Server on Ubuntu 18.04. URL: <https://www.howtoforge.com/tutorial/ubuntu-wildfly-jboss-installation/>.
- [13] How to install WildFly Server on an Ubuntu 18.04 VPS or Dedicated Server. URL: <https://hostadvice.com/how-to/how-to-install-wildfly-server-on-an-ubuntu-18-04-vps-or-dedicated-server/>.

- [14] ionic-image-loader - npm. URL: <https://www.npmjs.com/package/ionic-image-loader>.
- [15] Ionic native - geolocation. URL: <https://ionicframework.com/docs/native/geolocation/>.
- [16] Ionic native - HTTP. URL: <https://ionicframework.com/docs/native/http/>.
- [17] Ionic native HTTP instead of angulars http. URL: <https://forum.ionicframework.com/t/ionic-native-http-instead-of-angulars-http/101326/8>.
- [18] JSON Web Tokens - jwt.io. URL: <https://jwt.io/>.
- [19] Meet android studio. URL: <https://developer.android.com/studio/intro/>.
- [20] Mobile computing WS 2013/2014 - AR - geschichte der augmented reality. URL: <http://media2mult.uni-osnabrueck.de/pmwiki/fields/wp13/index.php?n=AR.GeschichteDerAugmentedReality>.
- [21] Most popular Java application servers: 2017 edition - Plumbr. URL: <https://plumbr.io/blog/java/most-popular-java-application-servers-2017-edition>.
- [22] nchutchind/cordova-plugin-app-launcher: Simple cordova plugin for launching apps. URL: <https://github.com/nchutchind/cordova-plugin-app-launcher>.
- [23] OneNote 2016, die App für digitale Notizen – Office. URL: <https://products.office.com/de-at/onenote/digital-note-taking-app>.
- [24] Pros and cons of angular development. URL: <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/>.
- [25] Pros and cons of the vue.js framework. URL: <https://naturaily.com/blog/pros-cons-vue-js>.
- [26] React native pros and cons. URL: <https://citrusbits.com/react-native-pros-and-cons/>.
- [27] Standard Notes | A Simple And Private Notes App. URL: <https://standardnotes.org/>.
- [28] Turtl features | Turtl. URL: <https://turtlapp.com/features/>.
- [29] Von IKEA bis toyota - interaktive printwerbung - xposeprint® blog. URL: <https://www.xposeprint.de/blog/423-interaktive-printwerbung-augmented-reality>.
- [30] Vor- und Nachteile von JAVA. URL: [http://ddi.cs.uni-potsdam.de/HyFISCH/Produzieren/SeminarDidaktik/OOP/java\\_vor\\_nachteile.html](http://ddi.cs.uni-potsdam.de/HyFISCH/Produzieren/SeminarDidaktik/OOP/java_vor_nachteile.html).

- [31] Vuforia augmented reality SDK. Page Version ID: 868057495. URL: [https://en.wikipedia.org/w/index.php?title=Vuforia\\_Augmented\\_Reality\\_SDK&oldid=868057495](https://en.wikipedia.org/w/index.php?title=Vuforia_Augmented_Reality_SDK&oldid=868057495).
- [32] What is augmented reality technology and how does AR work. URL: <https://thinkmobiles.com/blog/what-is-augmented-reality/>.
- [33] Why quill. URL: <https://quilljs.com/guides/why-quill/>.
- [34] Wikitude. Page Version ID: 869024691. URL: <https://en.wikipedia.org/w/index.php?title=Wikitude&oldid=869024691>.
- [35] WildFly Homepage · WildFly. URL: <http://www.wildfly.org/>.
- [36] WYSIWYG. Page Version ID: 183751896. URL: <https://de.wikipedia.org/w/index.php?title=WYSIWYG&oldid=183751896>.
- [37] 5 Open Source Application Servers (Comparison and Review), July 2017. URL: <https://shadow-soft.com/open-source-application-servers/>.
- [38] Anwendungsserver, August 2018. Page Version ID: 179830996. URL: <https://de.wikipedia.org/w/index.php?title=Anwendungsserver&oldid=179830996>.
- [39] Oracle to charge for Java from Jan 2019, May 2018. URL: <https://www.itassetmanagement.net/2018/05/01/oracle-to-charge-for-java-from-jan-2019/>.
- [40] Aug 17, 2016 | Tutorials, Ubuntu, Web Frameworks, and Web Servers | 13. How to install WildFly on Ubuntu 16.04, August 2016. URL: <https://www.rosehosting.com/blog/install-wildfly-with-nginx-as-a-reverse-proxy-on-ubuntu-16-04/>.
- [41] Author alexismp. Top reasons why GlassFish v3 is a lightweight server, July 2009. URL: <https://alexismp.wordpress.com/2009/07/10/top-reasons-why-glassfish-v3-is-a-lightweight-server/>.
- [42] Drifty. Ionic framework. URL: <https://ionicframework.com/docs/native/geolocation/>.
- [43] Ionic. Free mobile app development: Getting started with ionic apps. URL: <https://ionicframework.com/getting-started>.
- [44] Phillip Krüger. Swagger documentation for Java EE projects. Contribute to phillip-kruger/apiee development by creating an account on GitHub, October 2018. original-date: 2017-05-27T08:37:33Z. URL: <https://github.com/phillip-kruger/apiee>.

- [45] Alexander Pinker. AR healthcare: Augmented reality in der medizin. URL: <https://medialist.info/2017/01/13/ar-healthcare-augmented-reality-in-der-medizin/>.
- [46] Cristian Sulea. RESTful Java JAX RS Exception Handling - Knowledge Base - Cristian Sulea. URL: <http://cristian.sulea.net/blog/rest-java-jax-rs-exception-handling/>.
- [47] Breitenauer Bernd und Frösch Bernhard und Hanner Jakob. In *Diplomarbeit: Augmented Reality Machbarkeitsstudie Hofburg-App, HTL Leonding*, page 91f.
- [48] Bengt Weiße. An angular (>=2) component for the quill rich text editor: KillerCodeMonkey/ngx-quill. original-date: 2016-11-23T16:09:33Z. URL: <https://github.com/KillerCodeMonkey/ngx-quill>.

# Abbildungsverzeichnis

1.1	https://evernote.com/intl/de/ . . . . .	7
1.2	www.onenote.com . . . . .	8
1.3	https://turtlapp.com/ . . . . .	8
1.4	https://standardnotes.org/ . . . . .	9
3.1	https://alivenewspaper.com/2017/11/dr-mark-mckenna-reports-on-augmented-reality-in-medicine/ . . . . .	17
3.2	https://www.imore.com/pokemon-go-tips-tricks . . . . .	17
3.3	https://www.dezeen.com/2013/08/05/ikea-launches-augmented-reality-catalogue/ . . . . .	18
3.4	https://thinkmobiles.com/blog/what-is-augmented-reality/ . . . . .	19
3.5	https://blog.vakoms.com/everything-you-need-to-knowto-build-location-based-ar-app/ . . . . .	20
3.6	https://thinkmobiles.com/blog/what-is-augmented-reality/ . . . . .	20
3.7	https://thinkmobiles.com/blog/what-is-augmented-reality/ . . . . .	21
4.1	https://docs.unity3d.com/uploads/Main/vuforia_logo.png . . . . .	23
4.2	Screenshot: Diplomarbeit „Augmented Reality Machbarkeitsstudie Hofburg-App“ Seite 92 . . . . .	23
4.3	https://developer.vuforia.com/vui/pricing . . . . .	24
4.4	https://www.wikitude.com/media-resources/ . . . . .	25
4.5	https://www.wikitude.com/store/ . . . . .	26
5.1	Screenshot: Desktop Loginansicht . . . . .	28
5.2	Screenshot: Desktop Registrierenansicht . . . . .	28
5.3	Screenshot: Desktop Homeansicht mit Details . . . . .	29
5.4	Screenshot: Desktop Objekt bearbeiten . . . . .	29
5.5	Screenshot: Desktop Objekt erstellen . . . . .	30
5.6	Screenshot: Layout Desktop Bild hochladen . . . . .	31
5.7	Screenshot: Desktop Feed . . . . .	31
5.8	Screenshot: Desktop Benutzergruppen . . . . .	32
5.9	Screenshot: Smartphone Loginansicht . . . . .	32
5.10	Screenshot: Smartphone Registrierenansicht . . . . .	33
5.11	Screenshot: Smartphone Homeansicht . . . . .	34

5.12 Screenshot: Smartphone Objekt-Detailansicht . . . . .	34
5.13 Screenshot: Smartphone Objekt bearbeiten . . . . .	35
5.14 Screenshot: Smartphone Benutzergruppen . . . . .	35
5.15 Screenshot: Smartphone Feed . . . . .	36
5.16 Screenshot: Smartphone Verwendung von Wikitude . . . . .	37
5.17 Screenshot: Smartphone Verwendung von Vuforia . . . . .	37
5.18 <a href="https://de.wikipedia.org/wiki/Datei:Ionic_Logo.svg">https://de.wikipedia.org/wiki/Datei:Ionic_Logo.svg</a> . . . . .	38
5.19 <a href="https://cdn.ukad-group.com/media/2081/angular-ionic.jpg">https://cdn.ukad-group.com/media/2081/angular-ionic.jpg</a> . . . . .	40
5.20 Screenshot: Compodoc . . . . .	41
5.21 <a href="https://ionicframework.com/img/getting-started/starter-app-thumbnails-2.png">https://ionicframework.com/img/getting-started/starter-app-thumbnails-2.png</a> . . . . .	42
5.22 Screenshot: Quill-Editor in Footnote . . . . .	46
5.23 <a href="https://developer.android.com/studio">https://developer.android.com/studio</a> . . . . .	48
5.24 Screenshot: USB-Debugging erlauben . . . . .	49
5.25 Screenshot: Kontrollkästchen USB-Geräte erkennen . . . . .	49
5.26 Screenshot: angeschlossenes Remote-Gerät . . . . .	50
5.27 Screenshot: Port Forwarding . . . . .	51
5.28 Screenshot: Port Forwarding Status . . . . .	51
5.29 <a href="https://assetstore.unity.com/packages/templates/packs/vuforia-books-sample-102254">https://assetstore.unity.com/packages/templates/packs/vuforia-books-sample-102254</a> . . . . .	53
5.30 <a href="https://developer.vuforia.com/targetmanager/project/deviceTargetListing">https://developer.vuforia.com/targetmanager/project/deviceTargetListing</a>	54
6.1 UML . . . . .	56
6.2 Screenshot: Die Rest Endpunkte und deren Repositories . . . . .	58
6.3 Screenshot: Struktur des Auth Package . . . . .	58
6.4 Screenshot: Klassen zur Erfassung und Löschung von Objekten . . . . .	59
6.5 Screenshot: Struktur des „logic“-Pakets . . . . .	59
6.6 <a href="https://hirschtec.eu/java-logo-icon">https://hirschtec.eu/java-logo-icon</a> . . . . .	60
6.7 <a href="https://plumbr.io/blog/java/most-popular-java-application-servers-2017-edition">https://plumbr.io/blog/java/most-popular-java-application-servers-2017-edition</a> . . . . .	63
6.8 <a href="http://design.jboss.org/wildfly/">http://design.jboss.org/wildfly/</a> . . . . .	63
6.9 <a href="https://www.jumpingbean.co.za/glassfish">https://www.jumpingbean.co.za/glassfish</a> . . . . .	64
6.10 <a href="http://tomcat.apache.org/">http://tomcat.apache.org/</a> . . . . .	65
6.11 payara.fish . . . . .	65
6.12 <a href="https://jwt.io/">https://jwt.io/</a> . . . . .	66
6.13 <a href="http://www.avanzegroup.com/tSuite/images/thirdparty/tesseract.png">http://www.avanzegroup.com/tSuite/images/thirdparty/tesseract.png</a> . .	68
6.14 <a href="http://logging.apache.org/log4j/2.x/">http://logging.apache.org/log4j/2.x/</a> . . . . .	69
6.15 Bereits existierende Implementationen, die auf der Klasse WebApplicationException aufbauen . . . . .	71
6.16 <a href="https://swagger.io/swagger/media/blog/wp-hub/OpenAPI-Whats-New.png">https://swagger.io/swagger/media/blog/wp-hub/OpenAPI-Whats-New.png</a>	81
6.17 Screenshot: Swagger . . . . .	82
6.18 <a href="https://assets.ubuntu.com/v1/Ubuntu JPEG">https://assets.ubuntu.com/v1/Ubuntu JPEG</a> . . . . .	84
6.19 Screenshot: Add-User Dialog . . . . .	86

6.20 https://www.nginx.com/ . . . . .	88
6.21 Screenshot: Im Admin-Interface in den Configuration Tab wechseln . . . . .	91
6.22 Screenshot: Eine Datasource hinzufügen . . . . .	91
6.23 Screenshot: Deployment Dialog öffnen . . . . .	92
7.1 Screenshot: Projektwebsite-Startansicht . . . . .	93
7.2 Screenshot: Webseite . . . . .	94
7.3 Screenshot: Webseite . . . . .	94
7.4 Footnote-Logo . . . . .	95
7.5 Footnote-Schriftzug . . . . .	95
7.6 Footnote-Plakat . . . . .	97

# Tätigkeitsnachweis

Isabella Hundstorfer

Datum	Tätigkeiten	Dauer
17.07.2018	Projektstrukturplan, Pflichtenheft und Planung erstellen	3 h
01.08.2018	Pflichtenheft überarbeitet	1 h
07.08.2018	Android-Studio installiert, Android Grundlagen aneignen, AR Recherche	3 h
08.08.2018	ARCore Recherche	1 h
09.08.2018	„Vuforia“ Recherche	2 h
16.08.2018	Ionic Recherche, Ionic Installation, Unity-Ionic-Integration Recherche	4 h
17.08.2018	Ionic Grundlagen aneignen, Ionic Objekt-Liste mit Detailansicht erstellt	5 h
18.08.2018	Ionic „Notizen hinzufügen“ erstellen	1 h
19.08.2018	Skype-Gespräch: Themen: Planung, Wildfly Probleme	1 h
19.08.2018	Lauffähiges Wildfly-Rest-Datebank Projekt	1 h
22.08.2018	Server ausführen: Probleme mit Transaktionen	1 h
23.08.2018	Lokale Probleme mit der Serverinstallation beheben	1 h
23.08.2018	erste AR-App mit Unity und „Vuforia“ erstellen	1 h
25.08.2018	Zusammenfügen von Client und Server	2 h
26.08.2018	Beheben von: Unsupported Media Type-Error	1 h
27.08.2018	Beheben von: 404 Error Method not found	2 h
28.08.2018	Erweitern von Server und Client damit statische Objekte und Notizen angezeigt, hinzugefügt, geändert und gelöscht werden können und Recherche: „Vuforia“ und „Wikitude“	5 h
29.08.2018	LaTeX-Diplomarbeitstemplate anpassen, Ionic-App auf Android-Smartphone zum Laufen bringen, „Wikitude“-Ionic-Beispiel erstellen	7 h
30.08.2018	Recherche zum Thema Navigation in Ionic	2 h
31.08.2018	Objekte durchsuchen clientseitig, Dateien Upload clientseitig	3 h
05.09.2018	Skype-Gespräch: Themen: aktuellen Stand, Fileupload, Quill-Editor	5 h
09.09.2018	Quill-Editor einbauen am Client	2 h

Datum	Tätigkeiten	Dauer
11.09.2018	Serverprobleme beheben	2 h
12.09.2018	„Wikitude“ Lizenz Recherche	2 h
12.09.2018	Dateiupload verbessern	2 h
25.09.2018	„Wikitude“ Lizenzanfrage	1 h
30.09.2018	Login mit JWT	2 h
05.10.2018	„Wikitude“ Education-Lizenz	1 h
10.10.2018	JWT erweitern	1 h
12.10.2018	Server und Client zusammenfügen	3 h
14.10.2018	„Wikitude“ Image Recognition testen	4 h
15.10.2018	Recherche und Testen der Ionic „Vuforia“ Integration	3 h
16.10.2018	Client Probleme beheben	3 h
18.10.2018	Testen von Cloud Recognition mit Unity und „Vuforia“	1 h
21.10.2018	Testen von mehreren Targets mit Unity und „Vuforia“	2 h
29.10.2018	Recherche: Ionic „Vuforia“ Integration	2 h
30.10.2018	Schriftliche Arbeit Grundgerüst und erstellen der Versionsverwaltung mit Git	4 h
31.10.2018	schriftliche Arbeit Kapitel: Augmented Reality	4 h
02.11.2018	Skype Gespräch: Thema: LaTeX und Zotero Verwendung und schriftliche Arbeit Kapitel: Augmented Reality erweitern	2 h
13.11.2018	ausprobieren der Ionic-Unity Kommunikation mit nativen Android	2 h
16.11.2018	ausprobieren der Ionic-Unity Kommunikation mit nativen Android	4 h
20.11.2018	Recherche: „Vuforia“ mit nativen Android	3 h
30.11.2018	Probleme mit Wildfly und Ionic, 405 Method not allowed	2 h
03.11.2018	ausprobieren der Ionic-Unity Kommunikation	2 h
04.12.2018	Client an neue Serverversion anpassen	3 h
06.12.2018	Wildfly SQL Error beheben	1 h
09.12.2018	„Vuforia“ Image Upload Problem serverseitig lösen	3 h
11.12.2018	„Vuforia“ Bildupload clientseitig umsetzen	2 h
23.12.2018	Plakat designen	2 h
30.12.2018	„Vuforia“ für natives Android installieren, schriftliche Arbeit: „Vuforia“, „Wikitude“ beschreiben	4 h
31.12.2018	Plakat designen	3 h
01.01.2019	Android Objekterkennung mit „Vuforia“ für Footnote implementieren	5 h
02.01.2019	Android Problem lösen	3 h
03.01.2019	Optimierung der „Vuforia“ Objektanzeige	4 h
05.01.2019	schriftliche Arbeit erweitern, Quill und Pflichtenheft schreiben	3 h
13.01.2019	weiterarbeiten an Ionic	2 h
15.01.2019	Skype-Gespräch: Thema: weiteres Vorgehen	1 h

Datum	Tätigkeiten	Dauer
15.01.2019	CORS-Problem lösen, Login designen, Kapitel Port Forwarding schreiben	4 h
17.01.2019	Dateiupload abändern	2 h
18.01.2019	schriftliche Arbeit: Remote Debuggen, Ionic beschreiben	2 h
02.02.2019	schriftliche Arbeit Änderungswünsche von Herrn Professor Rager einarbeiten	2 h
04.02.2019	erstellen der Webseite mit Gitlab-Pages und Bootstrap	3 h
07.02.2019	Android: Design, Bilder, JWT	2 h
12.02.2019	„Wikitude“: Objekte in der Nähe erkennen	7 h
19.02.2019	Benutzergruppen und Feed clientseitig umsetzen	7 h
20.02.2019	Benutzergruppen fertigstellen, Dateien bei Notiz anzeigen	6 h
22.02.2019	schriftliche Arbeit überarbeiten, Code dokumentieren	3 h
23.02.2019	Server und Client Problem bei Rest-Aufrufen korrigieren	4 h
24.02.2019	Server und Client Problem bei Rest-Aufrufen korrigieren	6 h
02.03.2019	Abstract umschreiben, Client beschreiben	3 h
03.03.2019	Übergabe des JW-Tokens von Ionic nach „Wikitude“ implementieren, schriftliche Arbeit erweitern	3 h
06.03.2019	Client Beschreibung mit Screenshots fertigstellen, Preise „Wikitude“ und „Vuforia“ in die schriftliche Arbeit einfügen, Bilder zentrieren	3 h
17.03.2019	Abbildungsverzeichnis Client und AR korrigieren	1 h
23.03.2019	Gesamte Arbeit überarbeiten	5 h
01.04.2019	Gesamte Arbeit fertigstellen	5 h
<b>Gesamt:</b>		<b>210 h</b>

## Christopher Gusenbauer

Datum	Tätigkeiten	Dauer
03.07.2018	Austausch mit Diplomarbeitskollegin über LaTeX	1 h
17.07.2018	Projektstrukturplan, Pflichtenheft und Planung erstellen	3 h
02.08.2018	Vergleichen mehrerer Application Server, Registrierung von einer Dev-DB, Testen von Docker, Projektcloudordner anlegen	5 h
03.08.2018	Recherche zu Payara	3 h
04.08.2018	Wildfly Grundgerüst erstellen, Datenbank mit Wildfly verbinden	3 h
17.08.2018	serverseitige Programmierung einer Basisnotizapp, Wildfly Fehler beheben, Gitlab Repo erstellen	4 h
18.08.2018	Wildfly neu aufsetzen, JAX-RS Doku studieren	4 h
19.08.2018	Skype Gespräch: Planung optimieren, Wildfly Problem besprechen; Lauffähiges Wildfly-REST-Projekt mit Datenbank-Anbindung erstellen	2 h
20.08.2018	„Insert“ und „Get“ Funktionen programmieren, erstellen von Entities, Datenbank-Struktur erstellen	1 h
22.08.2018	Server Problem mit Transaktions reparieren	2 h
24.08.2018	Server testen, Persistence Problem beheben	2 h
27.08.2018	Mehrere Skype Gespräche mit Kollegin wegen Java-EE Problemen	2 h
28.08.2018	Probleme mit Server-Client Kommunikation beheben, Server-Client-Schnittstelle anpassen, CORS Fehlersuche	5 h
02.09.2018	Remote Server erstellen, Wildfly konfigurieren, Mysql Datenbank erstellen, GPS Funktionalität am Server implementieren, xrdp Installation für Remote Desktop Verbindungen, Ionic ausprobieren, File Upload am Server möglich machen	6 h
05.09.2018	Server mit Beispielwerten populieren, Bugfixes, Skype-Gespräch über aktuellen Stand, Fileupload clientseitig, Quill	3 h
13.09.2018	Serverarbeit	2 h
28.09.2018	Production Environment fertigstellen, Swagger.io testen	6 h
05.10.2018	Dokumentation aktualisieren	2 h
06.10.2018	Swagger.io verwenden, Recherche zu Security & Auth Lösungen	2 h
11.10.2018	Hinzufügen von User, Auth, Login, Server absichern mit JW-Tokens, Swagger & Auth zusammen möglich machen, Passwort-Hashing implementierten	6 h
12.10.2018	Zusammenfügen der Projektteile, zusammenarbeiten um Client und Server bei Auth & Login kommunizieren zu lassen	3 h
16.10.2018	OCR einbauen	1 h
17.10.2018	Inhaltsverzeichnis erstellen, Server umbauen auf AbstractFacades, Generics, normierte REST Werte, „init“ automatisiert bei „startup“	6 h
28.10.2018	Recherche zu JW-Tokens, Wildfly, LaTeX einarbeiten	3 h

Datum	Tätigkeiten	Dauer
02.11.2018	Texstudio installieren, schreiben der Kapitel: JW-Token, Application Server, Skype Gespäch über LaTeX	8 h
15.11.2018	Schriftlicher Teil - allgemeine Informationen über Digitale Informationen und Abgrenzungskriterien schreiben	3 h
18.11.2018	Files „trackbar“ machen und zu Notes anhängen	3 h
19.11.2018	Posten eines Bildes auf Vuforia implementieren, „java server best practices“ Recherche	3 h
26.11.2018	Schriftliche überarbeiten: Citations, Wildfly auf Version 14 updaten	4 h
03.12.2018	Gespräch mit dem Entwickler des Apiee Repository wegen Swagger und Generics	1 h
04.12.2018	Logging, Exceptionmapping, Rest Responses ansehen, Objekte abonnierbar machen	4 h
08.12.2018	Swaggerentwickler Antwort umsetzen, Serverbugfixes	2 h
16.12.2018	Fileupload für Client umändern auf Servlets, schriftliche Arbeit erweitern	2 h
19.12.2018	Logo überarbeiten, Protokoll erstellen	2 h
01.01.2019	Debugging am Server ermöglichen, Erkenntnis: gleiche „.war“ verhält sich auf verschiedenen Wildfly-Installationen anders	4 h
02.01.2019	Serverarbeiten, Plakat überarbeiten	2 h
06.01.2019	Server optimieren, Informationen über Objekte mit Vuforia Objekten verknüpfen	2 h
07.01.2019	Usergruppen, Feeds, rollenverteilten Content implementieren	4 h
08.0.2019	Datenmodell umschreiben, JPQL Querys schreiben	3 h
09.01.2019	Umschreiben des FileHandlings am Server	2 h
12.01.2019	Recherche zu REST	1 h
13.01.2019	CORS-Fehlersuche am Server	1 h
15.01.2019	Telefonat, weiterarbeiten an Server	2 h
17.01.2019	Imageupload abändern, am Server Performance des Uploads verbessern	3 h
18.01.2019	Kapitel Wildfly schreiben	3 h
24.01.2019	OCR für Server bereit machen, mehrere kleine Bugfixes, Doku	1 h
26.01.2019	Kleinere Arbeiten	1 h
30.01.2019	Schriftlichen Serverteil überarbeiten	3 h
02.02.2019	“exception handling“, Gitlab Pages recherchieren und probieren, besseres Feedback am Server für Clients programmieren	2 h
11.02.2019	Projektwebsite	3 h
18.02.2019	Recherche Markdown, Komfortfunktionen für Client, Pagination-count	3 h
19.02.2019	Gemeinsam an UserGruppen und Feed arbeiten	7 h

Datum	Tätigkeiten	Dauer
21.02.2019	„Delete“-Funktionen für Beziehungsobjekte implementiert, Server in finale Version bringen	5 h
21.02.2019	Schriftliche Arbeit: Allgemeine Erklärungen zum Projekt. schriftliche Arbeit zusammenfügen	5 h
22.02.2019	Änderungen für Client, Schriftliche Arbeit: Logging, Tesseract	5 h
24.02.2019	Serveränderungen für Clientanforderungen	1 h
25.02.2019	Abstract schreiben und Https-Forwarding einrichten	2 h
03.03.2019	Port Problem suchen und mithilfe des Systemadministrators lösen	1 h
10.03.2019	Fertigstellen der schriftlichen Arbeit: Formattierung, Beistriche, Screenshots, Kapitel zu Vuforia, Tesseract, Netbeans, Fileupload, Zitieren, Quellen	10 h
18.03.2019	Überarbeiten des schriftlichen Teils mit Herr Professor Ragers Korrekturen	4 h
20.03.2019	Bildüberschriften, Abbildungsverzeichnis	2 h
24.03.2019	Feedback der Korrekturleser einarbeiten	2 h
01.04.2019	schriftliche Arbeit überarbeiten	5 h
02.04.2019	Projektwebsite, Projekt und Diplomarbeit bereit für die Abgabe machen	2 h
	<b>Gesamt:</b>	<b>200 h</b>

# Kapitelzuordnung

## **Gemeinsam**

Zusammenfassung und Abstract  
Kapitel 1  
Kapitel 2

## **Isabella Hundstorfer**

Kapitel 3  
Kapitel 4  
Kapitel 5

## **Christopher Gusenbauer**

Kapitel 6  
Kapitel 7

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gусенбауэр

Besprechungsnummer:	1
Datum:	17.09.2018
Dauer:	30 min
Ort:	HTL Leonding EDV 7
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gусенбауэр Christopher
Themen:	AR-Frameworks, Server: JavaEE, Wildfly, JPA Client: Ionic, Angular
Inhalt:	Schwierigkeiten beim Finden eines geeigneten AR-Frameworks, Probleme beim Aufsetzen von Wildfly und JavaEE, Derzeitige Funktionen des Clients, siehe erfüllte Ziele Optimierungen im Design
Erfüllte Ziele:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Erster lauffähiger Prototyp mit Ionic und Angular</li><li>• Objekte können samt Koordinaten eingefügt werden</li><li>• Notizen können eingefügt werden</li><li>• Wikitude zeigt Objekte an bestimmten Koordinaten an</li></ul> Gусенбауэр Christopher: <ul style="list-style-type: none"><li>• Wildfly aufgesetzt</li><li>• Lauffähiger JavaEE-Server</li><li>• Benötigte REST-Funktionen für Client</li><li>• MySQL Datenbank aufgesetzt</li></ul>
Aufgaben/Ziele bis Mitte Oktober:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Fortführen der AR-Framework Recherche</li><li>• User einführen</li></ul> Gусенбауэр Christopher: <ul style="list-style-type: none"><li>• Vorzeigefähiges Runtime Environment</li><li>• User einführen</li></ul>

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	2
Datum:	15.11.2018
Dauer:	50 min
Ort:	HTL Leonding Raum 153
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gusenbauer Christopher
Themen:	Präsentation vom aktuellen Stand, definieren neuer Ziele
Inhalt:	Aktueller Stand: Vuforia, Ionic, Server, Wildfly, Abstract Facade, OCR Darlegung des Problems der Kommunikation zwischen Vuforia und Ionic. FileUpload funktioniert.
Erfüllte Ziele:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Fortführen der AR-Framework Recherche</li><li>• User einführen</li></ul> Gusenbauer Christopher: <ul style="list-style-type: none"><li>• Vorzeigefähiges Runtime Environment</li><li>• User einführen</li></ul>
Aufgaben/Ziele bis zum 6.12.2018:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Hochladen von Objekten auf Vuforia</li><li>• Am Client Liste mit Bildern und Objekten</li><li>• (Vuforia-Ionic Verknüpfung)</li></ul> Gusenbauer Christopher: <ul style="list-style-type: none"><li>• Benutzer registrieren</li><li>• Benutzer Objekte zuordnen</li><li>• (Unterschiedliche Notiztypen)</li><li>• (Überlegungen zu Nachrichten anzeigen und Filter)</li></ul>

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	3
Datum:	17.12.2018
Dauer:	30 min
Ort:	HTL Leonding Raum 253
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gusenbauer Christopher
Themen:	Schriftliche Arbeit, Zitieren
Inhalt:	Durchgehen des aktuellen Stands der schriftlichen Arbeit Richtiges Zitieren Spezifische Fragen zur schriftlichen Arbeit klären

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	4
Datum:	21.12.2018
Dauer:	20 min
Ort:	HTL Leonding Raum EDV 7
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gusenbauer Christopher
Themen:	Präsentation aktueller Stand, Ziele definieren
Inhalt:	Ionic Anwendung präsentiert, Objekt auf Vuforia hochgeladen und wiedererkannt, Ziele definiert
Erfüllte Ziele:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Hochladen von Objekten auf Vuforia</li><li>• Am Client Liste mit Bildern und Objekten</li></ul> Gusenbauer Christopher: <ul style="list-style-type: none"><li>• Benutzer registrieren</li><li>• Benutzer Objekte zuordnen</li></ul>
Aufgaben/Ziele bis zum 17.1.2018:	Hundstorfer Isabella: <ul style="list-style-type: none"><li>• Benutzbarkeit verbessern</li><li>• Design verbessern</li><li>• Informationen zu Objekten anzeigen</li><li>• Vuforia Ionic Kommunikation</li></ul> Gusenbauer Christopher: <ul style="list-style-type: none"><li>• Usergruppen fertig machen</li><li>• Server Informationen mit Vuforia-Objekt verknüpfen</li><li>• Neuer Notiz Typ: Reminder</li></ul>

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	5
Datum:	08.02.2019
Dauer:	50 min
Ort:	HTL Leonding Raum EDV 7
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gusenbauer Christopher
Themen:	Präsentation aktueller Stand, Ziele definieren
Inhalt:	Präsentation Kommunikation Vuforia Ionic, Designänderung, Umstellung des Servers Ziele definiert
Erfüllte Ziele:	<p>Hundstorfer Isabella:</p> <ul style="list-style-type: none"><li>• Benutzbarkeit verbessern</li><li>• Design verbessern</li><li>• Informationen zu Objekten anzeigen</li><li>• Vuforia Ionic Kommunikation mit JWT</li><li>• Neue Benutzer registrieren</li><li>• Informationen erstellen, löschen, bearbeiten</li></ul> <p>Gusenbauer Christopher:</p> <ul style="list-style-type: none"><li>• Usergruppen fertig machen</li><li>• Server Informationen mit Vuforia-Objekt verknüpfen</li><li>• Notiztypen</li><li>• Error-Handling</li><li>• Text aus jpeg erkennen</li></ul>
Aufgaben/Ziele bis zum 1.3.2019:	<p>Hundstorfer Isabella:</p> <ul style="list-style-type: none"><li>• Objekte in der Nähe anzeigen über GPS</li><li>• User-Gruppen am Client implementieren</li><li>• Desktop-Design anpassen</li><li>• Notiztypen</li></ul> <p>Gusenbauer Christopher:</p> <ul style="list-style-type: none"><li>• Objekte und Bilder aus Vuforia löschen</li><li>• Projektwebseite</li><li>• Usergruppen für Client umbauen</li></ul>

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	6
Datum:	28.02.2019
Dauer:	20 min
Ort:	HTL Leonding Raum 153
Anwesende:	Herr Professor Rager, Hundstorfer Isabella, Gusenbauer Christopher
Themen:	Abstract
Inhalt:	Durchbesprechen des Abstracts, Überarbeiten des Abstracts

# Footnote-Besprechungsprotokoll

Diplomarbeit Hundstorfer, Gusenbauer

Besprechungsnummer:	7
Datum:	04.03.2019
Dauer:	50 min
Ort:	HTL Leonding Raum EDV 7
Anwesende:	Herr Professor Rager, Hundstorfer Isabella
Themen:	Präsentation aktueller Stand, Ziele definiert
Inhalt:	Präsentation aktueller Stand, Ziele definiert
Erfüllte Ziele:	<p>Hundstorfer Isabella:</p> <ul style="list-style-type: none"><li>• Objekte in der Nähe anzeigen über GPS</li><li>• User-Gruppen am Client implementieren</li><li>• Desktop-Design anpassen</li><li>• Notizen mit Dateien</li><li>• eine Android App nicht in 2 verteilt</li></ul> <p>Gusenbauer Christopher:</p> <ul style="list-style-type: none"><li>• Objekte und Bilder aus Vuforia löschen</li><li>• Projektwebseite erweitern</li><li>• Usergruppen für Client umbauen</li></ul>
Aufgaben/Ziele bis zum 8.3.2019:	<ul style="list-style-type: none"><li>• Objekt auf Vuforia hochladen vom Smartphone</li></ul>
Aufgaben/Ziele bis zum 11.3.2019:	<ul style="list-style-type: none"><li>• Schriftliche Arbeit finalisieren</li></ul>