Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

Projektarbeit (Informatik)

Weiterentwicklung der Software „Interscriber“ zur Transkribierung von Audiodaten

Autoren	Yannik Roth Peter Unger
Hauptbetreuung	Prof. Dr. Mark Cieliebak
Industriepartner	SpinningBytes AG
Externe Betreuung	Flurin Gishamer
Datum	20.12.2019

Zürcher Hochschule
für Angewandte Wissenschaften

School of Engineering

Erklärung betreffend das selbständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Ort, Datum:

Winterthur, 20.12.2019

Unterschriften:

.....

.....

.....

Das Original dieses Formulars ist bei der ZHAW-Version aller abgegebenen Projektarbeiten zu Beginn der Dokumentation nach dem Titelblatt mit Original-Unterschriften und -Datum (keine Kopie) einzufügen.

1. Zusammenfassung

Diese Arbeit behandelt das Weiterentwickeln bestehender Features, sowie das Planen und Implementieren neuer Features für das Softwareprojekt „Interscriber“, welches von der SpinningBytes AG entwickelt wird. Es werden theoretische Grundlagen zu den eingesetzten Komponenten der Software behandelt. Die Anforderungserhebung erläutert die Analyse der gewählten Software-Architektur und der bestehenden Features. Unter Berücksichtigung der Erkenntnisse aus der Anforderungserhebung wird die Entwicklung neuer Features beschrieben. Der Fokus liegt dabei auf der Verwaltung der Transkriptionen in Form von Projekten. Die Features umfassen eine neue Projektübersicht, neue Dialoge für die Erstellung und den Import von Projekten, Eine Transkiptionsvorschau sowie eine Verifizierung eines eindeutigen Projektnamens und die Unterstützung mehrerer Audioformate. Der Stand der Software wird anschliessend im Hinblick auf die Änderungen beurteilt und ein Ausblick auf die Weiterentwicklung wird erläutert.

2. Abstract

This thesis deals with the further development of existing features as well as the planning and implementation of new features for the software project "Interscriber", which is developed by SpinningBytes AG. Theoretical basics of the used components of the software are covered. The requirements analysis explains the analysis of the selected software architecture and the existing features. The development of new features is described, taking into account the findings from the requirements analysis. The focus is on the management of transcriptions in the form of projects. The features include a new project overview, new dialogs for the creation and import of projects, a transcription preview as well as a verification of a unique project name and the support of multiple audio formats. The status of the software will then be assessed with regard to the changes and an outlook on further development will be explained.

3. Inhaltsverzeichnis

4. Einleitung

4.1. Ausgangslage

Die Software «Interscriber», welche in dieser Arbeit analysiert und weiterentwickelt wird, stellt einen Transkriptionsdienst von Gesprächen in Form von Audiodateien zu Textprotokollen zur Verfügung.

Transkriptionsdienste werden von Forschern, Journalisten, Medienfachleuten, Anwälten, Ärzten und Studenten in großem Umfang genutzt. All diese Berufsgruppen werden mit Interviews konfrontiert, die sie letztendlich auf Papier haben müssen. Heutzutage wird es immer üblicher, dass die Interviewer ein Diktiergerät zu einem Interview mitnehmen und es von einer dritten Partei abtippen lassen. Auch Studenten nutzen häufig einen Service, der ihnen bei der Durchführung von Interviews für ihre Diplomarbeit hilft. Damit ist gewährleistet, dass alle Informationen und Details des Interviews erhalten bleiben.

Traditionell wird ein Interview auf eine von zwei Arten transkribiert. In der Regel wird ein Transkriptionist für die manuelle Abschrift des Interviews eingestellt. Dies ist ein professioneller Dienstleister auf dem Gebiet der Transkription. Er oder sie hört sich die Sprachaufnahme an und tippt sie ab. Dies ist die genaueste Form der Transkription.

Darüber hinaus gibt es heute viele Softwareanwendungen, die für die Transkription von Interviews entwickelt wurden. Speech-to-Text-Engines können Wörter erkennen und dann viel schneller als Menschen tippen. Die Qualität von Transkriptions-Software variiert jedoch sehr stark. Es gibt verschiedene Optionen zwischen freier und kostenpflichtiger Software, wobei die kostenlose Version in der Regel eine geringere Genauigkeit bietet. Einige Anbieter sind auch stark in einer bestimmten Sprache, aber in einer anderen überhaupt nicht geübt.

Das initiale Softwareprojekt «Interscriber» resultierte aus der Bachelorarbeit „Automatische Transkription von Interviews“, welche 2019 an der ZHAW - School of Engineering durchgeführt wurde. Ziel dieser Arbeit war es, basierend auf den Erkenntnissen und Ergebnissen aus einer vorangegangenen Projektarbeit, ein System für die automatische Transkription von Interviews zu entwickeln. Es sollen Audio-Dateien der Anwendung übergeben werden können und das Transkript zurückerhalten werden. Anschliessend soll der Benutzer in einem integrierten Editor Text- sowie Spracherkennungsfehler korrigieren können. "Weiter soll das Transkript im Editor durch Hervorheben einzelner Wörter mit der Audio-Datei synchronisiert werden und es sollen dem Benutzer Möglichkeiten zum Export des Transkriptes bereitgestellt werden." (zitiert aus [1])
Das Projekt wird nun von der Spinning Bytes AG weiterentwickelt. In der Planung der weiteren Produktentwicklung ergaben sich neue Designentscheidungen bezüglich der Software-Architektur, was eine graduelle Anpassung des Codes auf die neue Architektur verlangt. Weiterhin sind neue Features geplant, um Interscriber zur Marktreife zu bringen.

4.2. Zielsetzung

Ein zuverlässiger und automatischer Dienst für die Erstellung und Bearbeitung von Transkriptionsprojekten würde die aufgewendete Zeit und Fehleranfälligkeit von Transkriptionsaufgaben stark reduzieren. Mit Interscriber soll dem Nutzer ein bedienungsfreundlicher Online-Dienst für die Verwaltung von Transkriptionsprojekten zur Verfügung gestellt werden. Nach der Erstellung eines Transkripts wird das Textprotokoll im Transkriptionsprojekt hinterlegt. Der integrierte Editor ermöglicht es, Korrekturen im Transkript vorzunehmen. Die Projekte können in Form eines PDF oder als Projektdatei exportiert werden.

Ziel dieser Arbeit ist es, bestehende Features auszubauen und auf die überarbeitete Software-Architektur anzupassen, sowie die Implementierung neuer Features. Bei der Auswahl neuer Features steht die Verwaltung von Transkriptionsprojekten sowie die Erstellung neuer Transkriptionsprojekte im Fokus dieser Arbeit. Weiterhin sollen mehrere Audioformate für die Transkribierung unterstützt werden.

5. Aufbau der Arbeit

Die eingesetzten Technologien in diesem Softwareprojekt werden im Kapitel «Theoretische Grundlagen» vorgestellt. Es wird hierbei auf die Komponenten für das Frontend, sowie das Backend der Anwendung eingegangen. In dem Kapitel «Anforderungserhebung» wird die gewählte Software-Architektur, sowie die vorhandenen Features analysiert. Eine Übersicht der neu implementierten Features und deren Motivation wird im Kapitel «Implementierung neuer Features» behandelt. Weiterhin wird im Kapitel «Ausblick» die Software in Bezug auf die resultierten Änderungen beurteilt. Abschliessend

6. Theoretische Grundlagen

In diesem Kapitel werden die theoretischen Grundlagen zu den eingesetzten Softwarekomponenten von Interscriber erläutert. Eine genauere Beschreibung zu der Implementierung dieser Komponenten wird in dem Kapitel „Anforderungserhebung“ gegeben.

6.1. MVC Framework

Die Abkürzung MVC steht für das Model-View-Controller Design-Pattern und spezifiziert, dass eine Applikation in 3 logische Schichten aufgeteilt wird. Die Datenschicht (Model), Präsentationsschicht (View) entkoppelt. Die Steuerungsschicht (Controller) dient als Interface zwischen der Datenschicht und der Präsentationsschicht.

6.2. REST API

Eine REST API ist eine Programmier-Schnittstelle, um eine Kommunikation zwischen 2 Systemen in Netzwerken zu ermöglichen. REST steht für Representational State Transfer, und beschreibt wie verteilte Systeme miteinander kommunizieren können. Dabei gibt REST keine konkreten Vorgaben an die Implementierung, sondern setzt folgende 6 Architekturprinzipien Voraus.

- **Client-Server-Modell**: REST verlangt ein Client-Server-Modell, will also das Nutzerinterface von der Datenhaltung getrennt sehen. Damit sollen sich Clients einerseits

leichter auf verschiedenen Plattformen portieren lassen; vereinfachte Serverkomponenten sollen andererseits besonders gut skalieren.

- **Zustandslosigkeit**: Client und Server müssen zustandslos („stateless“) miteinander kommunizieren. Das bedeutet: Jede Anfrage eines Clients beinhaltet alle Informationen, die ein Server benötigt; Server selbst können auf keinen gespeicherten Kontext zurückgreifen. Dieses Constraint verbessert damit Visibilität, Zuverlässigkeit und Skalierbarkeit. Hierfür nimmt REST jedoch Nachteile bei der Netzwerkperformanz in Kauf; überdies verlieren Server die Kontrolle über ein konsistentes Verhalten der Client-App.
- **Caching**: Um die Netzwerkeffizienz zu verbessern, können Clients vom Server gesendete Antworten auch speichern und bei gleichartigen Requests später erneut verwenden. Die Informationen müssen dem entsprechend als „cacheable“ oder „non-cacheable“ gekennzeichnet werden. Die Vorteile responsiverer Anwendungen mit höherer Effizienz und Skalierbarkeit werden dabei mit dem Risiko erkauft, dass Clients auf veraltete Daten aus dem Cache zurückgreifen.
- **Einheitliche Schnittstelle**: Die Komponenten REST-konformer Services nutzen eine einheitliche, allgemeine und vom implementierten Dienst entkoppelte Schnittstelle. Ziel des Ganzen sind eine vereinfachte Architektur und eine erhöhte Visibilität von Interaktionen. Dafür nimmt man eine schlechtere Effizienz in Kauf, wenn Informationen in ein standardisiertes Format gebracht – und nicht für die Bedürfnisse spezieller Anwendungen angepasst – werden.
- **Layered System**: REST setzt auf mehrschichtige, hierarchische Systeme („Layered System“) – jede Komponente kann ausschließlich jeweils direkt angrenzende Schichten sehen. Somit lassen sich beispielsweise Legacy-Anwendungen kapseln. Als Load Balancer agierende Vermittler („Intermediaries“) können überdies die Skalierbarkeit verbessern. Als Nachteile dieses Constraints gelten ein zusätzlicher Overhead und erhöhte Latenzen.
- **Code-On-Demand**: Dieses Constraint fordert, dass die Funktionen von Clients über nachlad- und ausführbare Programmteile erweitert werden können – etwa in Form von Applets oder Skripten. Als optionales Constraint kann diese Bedingung in bestimmten Kontexten jedoch deaktiviert sein.

7. Anforderungserhebung

Bevor neue Funktionen in die Anwendung implementiert werden können, muss ein Überblick über das System erfasst werden. Dazu wird die gewählte Software-Architektur, sowie die bereits bestehenden Funktionen analysiert.

7.1. Analyse Software-Architektur

Das Projekt unterteilt sich grundsätzlich in zwei Bereiche: Zum einen das Backend, welches die kritische Business-Logik der Software beinhaltet und zum anderen das Frontend, welches die grafische Benutzeroberfläche dem Nutzer bereitstellt. Beide Bereiche sind in sich abgeschlossene Systeme, welche über eine REST-Schnittstelle Daten austauschen. Die Backend-Applikation wird auf einem Webserver installiert und gestartet. Das Frontend wird bei einer Anfrage an die Interscriber Webseite an den Browser des Anwenders übertragen und ausgeführt.

REST API

Abbildung 1: Aufbau der Applikation

7.1.1. Backend Architektur

Das Backend wurde in der Programmiersprache Python entwickelt und besteht aus dem Webserver, einer Datenbank und einer „in-memory“ Job Queue. Zudem wird die externe Google Cloud Speech-to-Text API für den Transkribierungsprozess genutzt, da die Entwicklung einer eigenen Transkriptions-Engine für das Projekt erst noch in der Konzeptionsphase ist. Die Speech-to-Text API von Google bietet einen state-of-the-art Transkribierungsservice und unterstützt aktuell 120 verschiedene Sprachen.

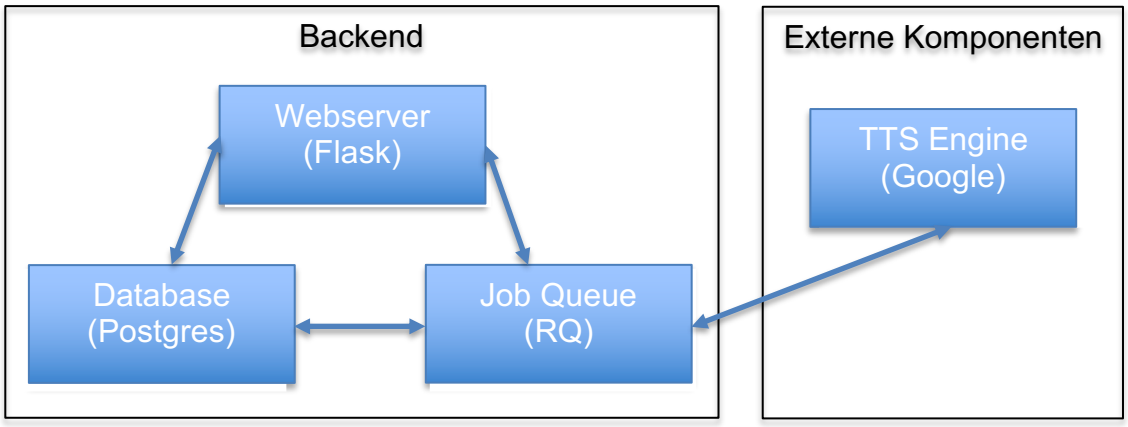

Abbildung 2: Aufbau Backend

Der Webserver wurde mit dem Web-Framework «Flask» implementiert. “Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications.” [2] [3]

Als Datenbanksystem wird Postgres eingesetzt. “PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance.” [4]

Für die Umsetzung der Job-Queue kommt die auf Redis basierende Python Modul «RQ» zum Einsatz. “RQ (Redis Queue) is a simple Python library for queueing jobs and processing them in the background with workers. It is backed by Redis and it is designed to have a low barrier to entry.” [5]

7.1.2. Frontend Architektur

Das Frontend hat die Aufgabe, die Benutzeroberfläche im Webbrowser des Nutzers darzustellen. Da das Frontend die Anwendungsdaten dynamisch über die REST Schnittstelle von dem Backend bezieht, muss es in der Lage sein, die Daten in der Oberfläche dynamisch zu aktualisieren.

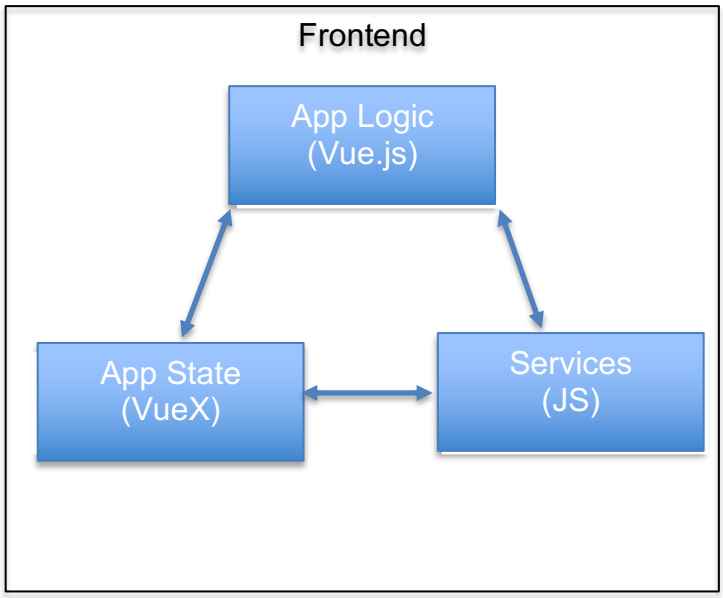

Abbildung 3: Aufbau Frontend

Daher wird die Applikationslogik mit dem JavaScript Framework „Vue.js“ implementiert. Vue.js ist ein MVC Framework und wird für die Programmierung von dynamischen Benutzeroberflächen in modernen Webbrowsern eingesetzt. Es basiert auf die Implementierung einzelner Vue-Komponenten. Vue-Komponenten sind einzelne, wiederverwendbare Bausteine mit benutzerdefinierter Logik. Die Komposition der Komponenten und damit die Darstellung der Benutzeroberfläche wird von dem Vue-Compiler übernommen.

Der Zustand der Daten in der Frontend-Anwendung wird mittels „VueX“, einer state-management Bibliothek für Vue.js, verwaltet. „Vuex is a state management pattern + library for Vue.js applications. It serves as a centralized store for all the components in an application, with rules ensuring that the state can only be mutated in a predictable fashion.” [6] Unter der Verwendung dieses state-management-pattern können die Applikationsdaten in einem zentralen Store verwaltet werden. Die einzelnen Vue-Komponenten beziehen dann die Daten aus dem zentralen Store. Dies sorgt für eine skalierbare Möglichkeit, Daten organisiert in der Anwendung zu verwalten.

Die Logik für das Senden und Empfangen der Applikationsdaten von der Server-Anwendung wird in einzelnen Service Komponenten implementiert. Services beinhalten Funktionen um mit der REST Schnittstelle des Backends zu kommunizieren.

7.2. Analyse bestehender Features

In diesem Kapitel werden die vorhandenen Features der Software in Bezug auf die Zielstellung dieser Arbeit analysiert. Im Fokus dieser Arbeit steht die Erstellung und Verwaltung von Transkriptionsprojekten. Zwei bestehende Features müssen dazu erweitert werden.

7.2.1. Projekt erstellen

Interscriber bietet auf der Startseite zwei Formulare für die Erstellung eines Projektes. Das erste Formular erwartet die Eingabe einer Audiodatei, sowie die Auswahl der Sprache und

die Anzahl der Sprecher. Die Sprache und die Anzahl der Sprecher sind notwendige Informationen um die Transkription durchzuführen, da andernfalls das Text-to-Speech Verfahren mit den falschen Parametern durchgeführt wird. Weiterhin steht noch das Formular für die Importierung eines existierenden Projektes bereit.

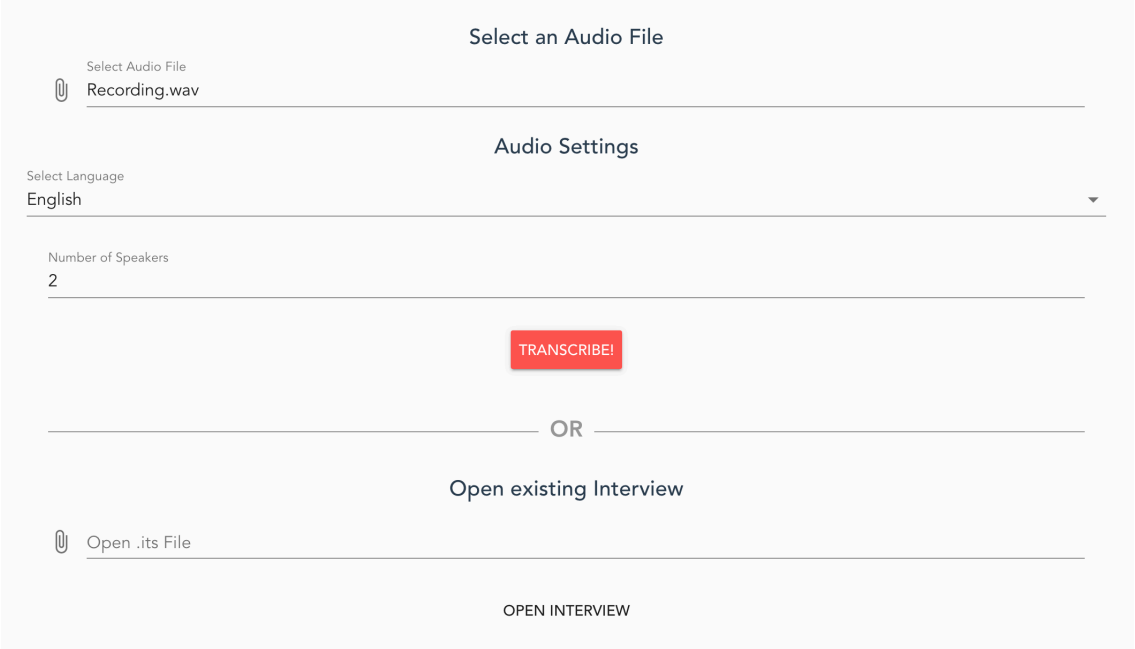

Abbildung 4: Ursprüngliche Dialog für Projekterstellung

7.2.2. Übersicht Transkripte

Auf der Übersichtsseite werden alle Transkripte eines Benutzers mit dem Namen Audiodatei aufgelistet. Mit einem Klick auf den Namen kann der Editor geöffnet werden. Mit der Mülltonne können Transkripte komplett gelöscht werden. Nachfolgend eine Abbildung, welche die Übersichtsseite zeigt.

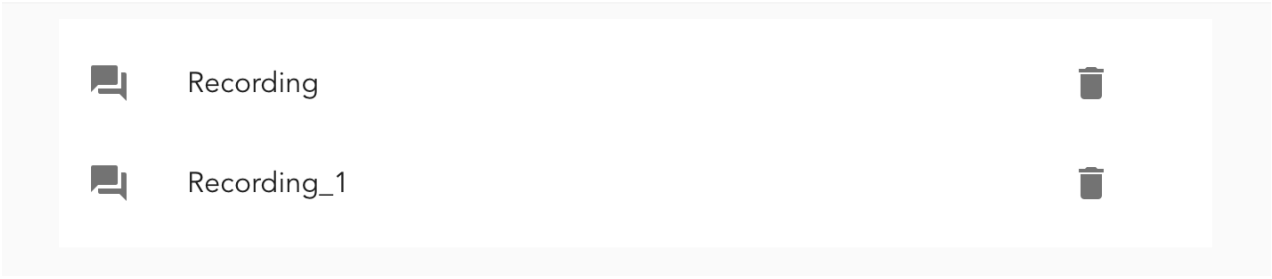

Abbildung 5: Übersicht über Transkripte

8. Implementation neuer Features

8.1. Neue Projektübersicht

Die alte Übersicht über die Transkripte eines Benutzers gibt nicht viele Informationen her. Es wird lediglich der Name der hochgeladenen Datei angezeigt und es existiert die Möglichkeit ein Transkript zu löschen.

Um dem Benutzer mehr und vor allem nützlichere Informationen anzuzeigen, wurde die Seite komplett überarbeitet. Die meisten Informationen waren eigentlich bereits im Backend vorhanden, sie wurden nur nicht im Frontend genutzt. Die Schnittstelle für die Informationsanfrage zu den Transkripten eines Benutzers musste also nur leicht angepasst werden.

Da die Projektübersicht das neue Zentrum und die Startseite von Interscriber sein soll, macht es auch Sinn, Funktionen wie den Export, den Import und die Erstellung eines Projekts auf dieser Seite zu platzieren. So sind die Hauptfunktionen von Interscriber schnell und einfach erreichbar.

In der nachfolgenden Abbildung ist die neue Projektübersicht abgebildet. Diese bietet deutlich mehr Informationen und Funktionalität als die alte Übersicht und eignet sich besser als Startseite.

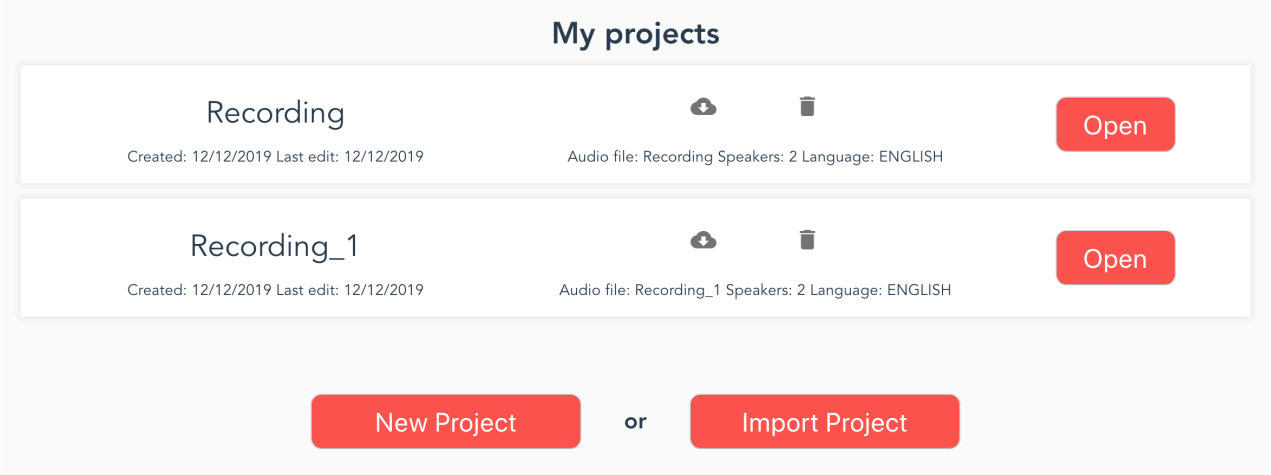

Abbildung 6: Projektübersicht

Die Funktionalität der alten Hauptseite für die Erstellung und den Import wurde aufgeteilt, und ist nun jeweils als eigenständiger Dialog auf der neuen Hauptseite verfügbar.

8.1.1. Dialog Projekterstellung

Der neue Dialog, um ein Projekt zu erstellen ist im Wesentlichen eine Umgestaltung der alten Hauptseite von Interscriber. Da die neue Hauptseite nun die Projektübersicht ist, wurde die alte Hauptseite als Dialog neu gestaltet.

Neu muss die Auswahl der Audiodatei bestätigt werden, bevor der Upload beginnt. So ist auch klar, dass der Benutzer die Audiodatei wirklich hochladen will. Zuvor wurde direkt nach der Auswahl der Upload gestartet.

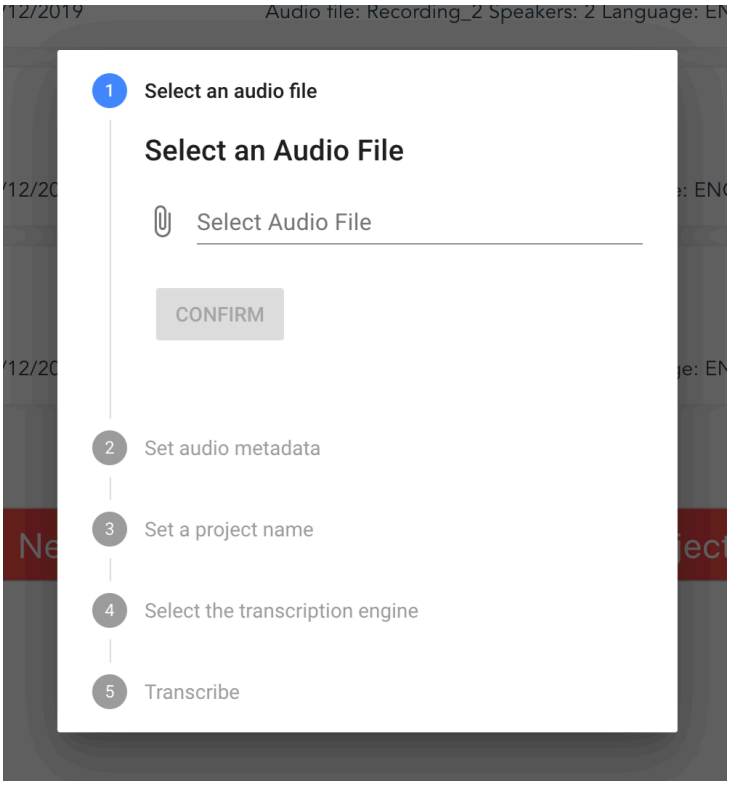

Abbildung 7: Neuer Dialog für Projekterstellung

Zusätzlich wurde die Option eingefügt, dem Projekt einen Namen zu geben zu können. Dieser Name wird dann auf der Projektübersicht als Projektname angezeigt und auch im Backend als Name hinterlegt. Zuvor wurden die Daten einfach mit dem Namen der heraufgeladenen Audiodatei versehen.

8.1.2. Dialog Projektimport

Die Funktionalität für den Import eines Projekts wurde von der alten Startseite übernommen und lediglich in einen Dialog gepackt, der in der Nachfolgenden Abbildung gezeigt wird. Es waren keinerlei Anpassungen im Backend nötig.

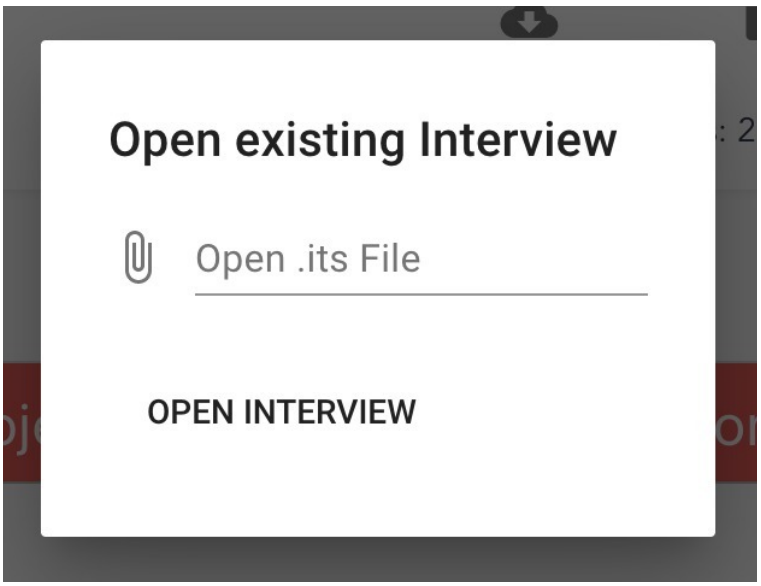

Abbildung 8: Dialog für Projektimport

8.1.3. Eindeutige Projektnamen

In der neuen Projektübersicht können die Projekte/Transkriptionen eines Benutzers am schnellsten über Namen identifiziert werden. Das bedarf einer Überprüfung, ob ein gegebener Name in den Projekten dieses Benutzers schon existiert.

Eine Möglichkeit dies zu lösen wäre, eine Überprüfungdurchzuführen, wenn der Benutzer die Transkription starten will. Eine andere ist es, eine neue REST-Schnittstelle einzuführen, mit der die Existenz eines Projektnamens für einen Benutzer geprüft werden kann. Somit kann direkt bei der Eingabe eines Projektnamens ein Fehler angezeigt werden und nicht erst nachdem der Benutzer die ganzen Informationen ausgefüllt hat und die Transkription starten will. Wir haben uns dafür entschieden beide Lösungen zu implementieren, um auch bei einem Fehler in der REST-Schnittstelle keine doppelten Namen in der Projektliste zuzulassen. Die Eindeutigkeit der Projektnamen ist wichtig, um die Projekte in der Übersicht auseinander halten zu können und da es im Moment keine Möglichkeit gibt, diese nach der Erstellung anzupassen oder zu ändern.

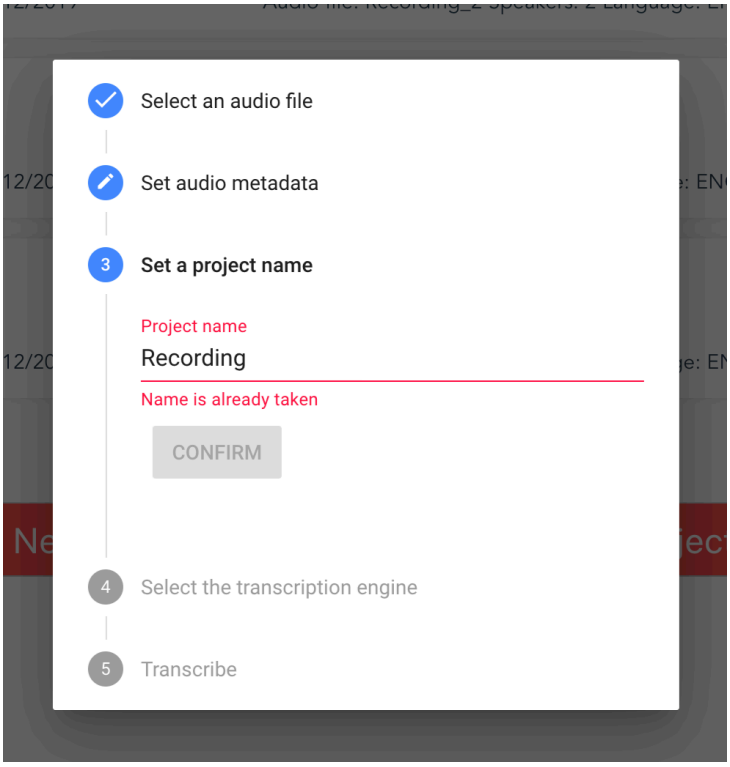

Abbildung 9: Bereits existierender Projektname

Ein weiterer Punkt den es zu beachten gilt, ist, dass auch beim Import eines Projektes der Name nicht schon existiert. Da es hier kein Feld zur Eingabe des Namens gibt, erfolgt die Überprüfung erst bei dem Import des Interviews. Existiert bereits ein Projekt mit diesem Namen, wird ein Bodenstrich mit einer aufsteigenden Zahl angehängt.

8.2. Transkript-Vorschau

Als Benutzer von Interscriber möchte ich gerne im Voraus wissen, wie gut der Dienst funktioniert, bevor ich mein zweistündiges Interview transkribieren lasse. Deshalb bietet sich eine Vorschaufunktion an, welche dem Benutzer einen Ausschnitt seiner Audiodatei transkribiert und dann den Text anzeigt. So kann unter anderem auch dem Upload einer falschen Datei vorgebeugt werden.

Die Grundidee der Vorschaufunktion ist es, einen zeitlich begrenzten Ausschnitt aus der hochgeladenen Audiodatei zu nehmen und dann diesen mit dem bereits existierenden Code zu transkribieren. Danach wird dem Benutzer das Transkript des Ausschnitts angezeigt.

Für die Umsetzung wird der Ausschnitt der Datei mit dem Tool „ffmpeg“ erstellt und als neue Datei auf dem Server abgespeichert. Der Benutzer kann eine Startzeit und die Dauer für die Vorschau angeben. Standardmässig ist die Dauer der Vorschau auf maximal zwei Minuten begrenzt. Dieser Wert hat sich beim Testen als sinnvoll erwiesen, da so die Sprechererkennung gut funktioniert, und der Benutzer eine gute Vorschau für die vollständige Transkription erhält. Diese Begrenzung der Vorschaudauer kann in der Konfigurationsdatei von Interscriber angepasst werden. Die Begrenzung ist global und für alle Benutzer von Interscriber gleich.

Der erstellte Ausschnitt wird dann an die Transkriptionsschnittstelle weitergegeben und von dieser verarbeitet. Es wird die bereits bestehende Schnittstelle verwendet, die auch für die komplette Transkription verwendet wird. Das fertige Transkript wird dann aus der Datenbank geholt, und dem Benutzer präsentiert. Er hat keine Möglichkeiten das Transkript zu editieren, da dies nicht der Sinn der Vorschau ist. Sobald der Benutzer sich für die Transkription der gesamten Datei entscheidet, wird die Vorschaudatei auf dem Server und das dazugehörige Transkript aus der Datenbank entfernt. Die Vorschau wird in keiner Weise für das Ergebnis der kompletten Transkription verwendet.

8.3. Unterstützung mehrerer Audioformate

Interscriber hat bisher nur Audiodateien im FLAC oder WAV Format unterstützt. Beides sind Formate die verlustfreie Audiodatenkompression bieten. Diese Limitierung ist der API-Schnittstelle von Google, welche zur Verarbeitung der Audiodaten im Backend verwendet wird, geschuldet. Grund dafür ist, dass eine verlustfreie Kompression der Audiodaten in den meisten Fällen zu einem besseren Ergebnis bei der Verarbeitung führt.

Da es aber auch Fälle geben kann, in denen eine Audiodatei nicht im FLAC oder WAV Format vorliegen aber trotzdem transkribiert werden soll, wurde die Möglichkeit eingebaut, hochgeladene Audiodaten der Benutzer automatisch in ein unterstütztes Format umzuwandeln. Dies führt zu einer Steigerung der Benutzerfreundlichkeit, da nicht jeder Benutzer gewillt ist, seine Dateien zuerst von Hand zu konvertieren, um sie dann auf Interscriber hochladen zu können.

Es muss jedoch bedacht werden, dass bei einer Audiodatei, welche ursprünglich mit einer verlustbehafteten Methode kodiert wurde, Informationen für immer verloren gegangen sind. Die Konvertierung von einem verlustbehafteten zu einem verlustfreien Format führt deshalb auch nicht zu einer Verbesserung der Audioqualität! Dies hat zur Folge, dass eine

konvertierte Datei nun zwar von der Google-API akzeptiert wird, die Genauigkeit der Spracherkennung eventuell aber darunter leidet. Ein weiterer Nachteil der Konvertierung ist auch, dass die Datei unnötig aufgeblasen wird und mehr Platz braucht, ohne dabei an Qualität zu gewinnen.

In den folgenden zwei Abbildungen wird der Unterschied der Transkription zwischen originaler WAV Aufnahme und konvertierter Aufnahme dargestellt. Bei beiden Varianten klappt aber die Erkennung des Textes recht gut. Bei der originalen Audiodatei funktioniert jedoch die Erkennung der einzelnen Sprecher besser als bei der konvertierten Datei.

Speaker 2	Hello Hi, how are you? I'm fine. How are you? I'm fine. Okay now maybe if you can say something why I'm
Speaker 1	Speaking just so we overlap. Okay. Okay. Yeah, I
Speaker 2	Think that's a good example. Yeah. Okay, I think that's enough.
Speaker 1	Yeah, we will fully cannot go over a minute.
Speaker 2	Yeah.

Abbildung 10: Transkription mit originaler Audiodatei

Speaker 1	Hello Hi, how are you? I'm fine. How are you? I'm fine. Okay now maybe if you can say something why I'm speaking just so we overlap. Okay. Okay. Yeah, I think that's a good example. Yeah. Okay, I think that's enough.
Speaker 2	Yeah, we will fully cannot go over a minute.
Speaker 1	Yeah.

Abbildung 11: Transkription mit konvertierter Audiodatei

Umgesetzt wurde die Konvertierung mit dem Tool «ffmpeg». In Python wird dann die Funktion *subprocess.call() v*erwendet, welche einen Befehl weiter an das darunterliegende Betriebssystem weitergibt. In unserem Fall lautet dieser Befehl:

ffmpeg -i "/Pfad/zur/Eingabe" -acodec pcm_s16le "/Pfad/zur/Ausgabe"

wobei "-acodec pcm_s16le" hier für die LINEAR16 Kodierung, welche von Google für das beste Resultat der Transkription empfohlen wird [3], steht.

Wir haben uns gegen die Nutzung einer einer Python-Bibliothek für die Kovertierung entschieden, da diese im Hintergrund dann sowieso ffmpeg verwendet. So vermeiden wir einen Bibliotheksimport, welcher eine weitere Abhängigkeit, eine weitere Fehlerquelle und unnötige Komplexität zu Interscriber hinzufügt.

9. Ausblick

Kleinere Arbeiten für die Zukunft sind unter anderem das Aufräumen, sowie das Erreichen einer besseren Testabdeckung des Codes, um Fehler, die z.B. durch das Umstrukturieren oder Aufräumen des Codes entstehen, frühzeitig zu entdecken.

Des weiteren ist ein Admin-Interface für die Einstellungen, wie unter anderem die maximale Vorschaudauer, ein Punkt in den man etwas Zeit investieren könnte, um die Handhabung von Interscriber etwas angenehmer zu gestalten.

Allgemein gibt es im Webinterface von Interscriber Dinge, welche man noch verbessern kann. Dazu gehört die Responsiveness der Seite, so dass sie auch auf einem mobilen Gerät problemlos verwendet werden kann. Das ist sicher eine Sache, welche vor der Veröffentlichung angegangen werden sollte.

Als längerfristiges Ziel ist der Aufbau von einem eigenen Spracherkennungsdienst, welcher anstelle von Google verwendet werden kann, denkbar. Dies würde es ermöglichen, die Audiodaten in der Schweiz zu verarbeiten, anstatt sie an Google weitergeben zu müssen.

9.1. Schlusswort

Uns ist es in dieser Arbeit gelungen, die bestenden Features des Sofwareprojektes so weiterzuentwickeln, dass die Erstellung und Verwaltung von Transkriptionsprojekten über die neue Projektübersicht einfacher und intuitiver zu bedienen ist. Viele Probleme konnten gelöst werden, und die gewünschte Funktionalität implementiert werden. Gewisse Dinge wie zum Beispiel das Testing kamen aber zugunsten von neuen Funktionen teils etwas zu kurz.

Literaturverzeichnis

[1] «Automatische Transkription von Interviews,» [Online]. Available: https://ba-pub.engineering.zhaw.ch/BA_WebPublication/Flyer.pdf?version=Bachelorarbeit2019&code=BA19_ciel_01&language=de. [Zugriff am 20. Dezember 2019].

[2] «Flask - The Pallets Projects,» [Online]. Available: https://palletsprojects.com/p/flask/. [Zugriff am 12. Dezember 2019].

[3] Google, «Best Practises,» 12 April 2019. [Online]. Available: https://cloud.google.com/speech-to-text/docs/best-practices. [Zugriff am 25. November 2019].

[4] «PostgreSQL: The World's Most Advanced Open Source Relational Database,» [Online]. Available: https://www.postgresql.org/. [Zugriff am 8. Dezember 2019].

[5] «RQ - Easy job queuer for python,» [Online]. Available: https://python-rq.org/docs/workers/. [Zugriff am 9. Dezember 2019].

[6] «Vuex,» [Online]. Available: https://vuex.vuejs.org/. [Zugriff am 16. Dezember 2019].

10. Abbildungsverzeichnis