



Duale Hochschule Baden-Württemberg Mannheim

Projektrealisierung
Projektabschlussbericht

Studiengang Wirtschaftsinformatik

Studienrichtung Data Science

| | |
|-----------------------|--------------------------------------------------|
| Verfasser(in): | J. Brenzinger, L. Steinert, P. Breucker, C. Rech |
| Matrikelnummern: | 1960679, 2617416, 5800129, 8028907 |
| Kurs: | WWI20DSA |
| Bearbeitungszeitraum: | 08.05.2023 – 27.07.2023 |

Inhaltsverzeichnis

| | | |
|----------|-----------------------------------------------|-----------|
| 1 | Pflichtenheft (updated) | 1 |
| 1.1 | Vorbemerkung | 1 |
| 1.2 | Ausgangssituation und Zielsetzung | 1 |
| 1.3 | Schnittstelle und Beteiligte | 2 |
| 1.4 | Rahmenbedingungen | 2 |
| 1.5 | Dekomposition des Gesamtsystems | 3 |
| 1.6 | Bearbeitung von Änderungswünschen | 4 |
| 1.7 | Funktionale Anforderungen | 5 |
| 1.7.1 | Sonstige Anforderungen (Kann) | 7 |
| 1.8 | Nicht-funktionale Anforderungen | 7 |
| 1.9 | Qualitätsüberprüfung | 8 |
| 1.9.1 | Abnahmekriterien | 8 |
| 1.9.2 | Vorgehen zur Ausgangsprüfung: | 9 |
| 1.10 | Lieferumfang | 9 |
| 2 | Gegenüberstellung der Projektkriterien | 10 |
| 2.1 | Erfüllte Kriterien | 10 |
| 2.1.1 | Prio 1 Anforderungen | 10 |
| 2.1.2 | Prio 2 Anforderungen | 11 |
| 2.2 | Nicht erfüllte Kriterien | 12 |
| 2.2.1 | Prio 1 Anforderungen | 12 |
| 2.2.2 | Prio 2 Anforderungen | 12 |
| 3 | Projektübergabe | 14 |
| 3.1 | Bewertung der Modellgüte | 14 |
| 3.1.1 | Evaluierung Textsummarization | 15 |
| 3.1.2 | Evaluierung Textclassification | 15 |
| 3.1.3 | Evaluierung Textsentiment | 16 |
| 3.2 | Übergabe der technische Anwendung | 17 |
| 3.3 | Verbesserungsmöglichkeiten | 17 |
| 4 | Benutzerhandbuch | 19 |
| 4.1 | Systemanforderungen | 19 |
| 4.2 | Benutzeroberfläche | 19 |
| 4.3 | Anwendungsanleitung | 20 |
| 4.4 | Modellverwendung | 24 |

| | | |
|----------|-----------------------------|-----------|
| 5 | Fazit und Rückblick | 25 |
| 5.1 | Reflexion | 25 |
| 5.2 | Lessons Learned | 27 |
| | Literaturverzeichnis | 30 |

1 Pflichtenheft (updated)

1.1 Vorbemerkung

Basierend auf dem Lastenheft erfolgt anschließend die Beschreibung der Umsetzung der Anforderung im Zuge der Erstellung des Pflichtenheftes. Nach der DIN 69901 enthält ein Pflichtenheft „vom Auftragnehmer erarbeitete Realisierungsvorgaben auf der Basis des vom Auftraggeber vorgegebenen Lastenhefts“. Juristisch stellt das Pflichtenheft die im Lastenheft definierten Spezifikation in feingranulare fachlicher Art und Weise dar. Dagegen betrachtet die IT das Pflichtenheft als Beschreibung der technischen Lösung zu den im Lastenheft definierten Anforderungen. Aus der Sicht des Auftragnehmers stellt das Pflichtenheft demnach die formelle und detaillierte Antwort auf die Anforderungen des Auftraggebers dar, die zuvor im Lastenheft beschrieben wurden. Die zu erbringenden Ergebnisse des Auftragnehmers werden dadurch in erforderliche Tätigkeiten (Pflichten) umgesetzt (vgl. Aichele et al., 2014, S.173 ff).

1.2 Ausgangssituation und Zielsetzung

Die Ausgangssituation besteht darin, dass das Meinungsforschungsinstitut regelmäßig große Mengen an Texten analysiert, diese zusammengefasst und klassifiziert werden müssen. Das Ziel ist daher die Entwicklung eines Tools, das die Effizienz und Genauigkeit dieses Prozesses verbessert, indem es automatische Textzusammenfassung und -klassifikation ermöglicht.

Basierend auf dem Lastenheft des Meinungsforschungsinstitutes widmen sich die nachstehenden Kapitel der Beschreibung jeglicher Anforderungen und wie diese funktional und nicht-funktional umgesetzt werden sollen. Abgrenzend zu erwähnen sei die Tatsache, dass wie im Lastenheft beschrieben der Projektauftrag eine komplette Neuentwicklung bedeutet. Es bestehen keine Systeme oder Produkte bisher in der Nutzung.

1.3 Schnittstelle und Beteiligte

Beteiligt an diesem Projekt sind zum einen die Auftraggeber Enzo Hilzinger und Michael Lang, sowie die Gruppe 1 als Entwickler und Auftragnehmer. Der fiktive Usecase eines Meinungsforschungsinstitut geht dabei zurück auf die von den Auftraggebern geforderte Entwicklung des Produktes im Rahmen der Vorlesung "Projektrealisierung".

1.4 Rahmenbedingungen

Im Rahmen dieses Projekts „Entwicklung eines Tools zur Textzusammenfassung und -klassifikation“ sind bestimmte Rahmenbedingungen festzulegen, um eine reibungslose Umsetzung zu gewährleisten. Diese Rahmenbedingungen beinhalten die Bearbeitungszeit sowie die geplanten Betriebs- und Arbeitszeiten. Im Folgenden sind die spezifischen Regelungen für diese Aspekte detailliert beschrieben:

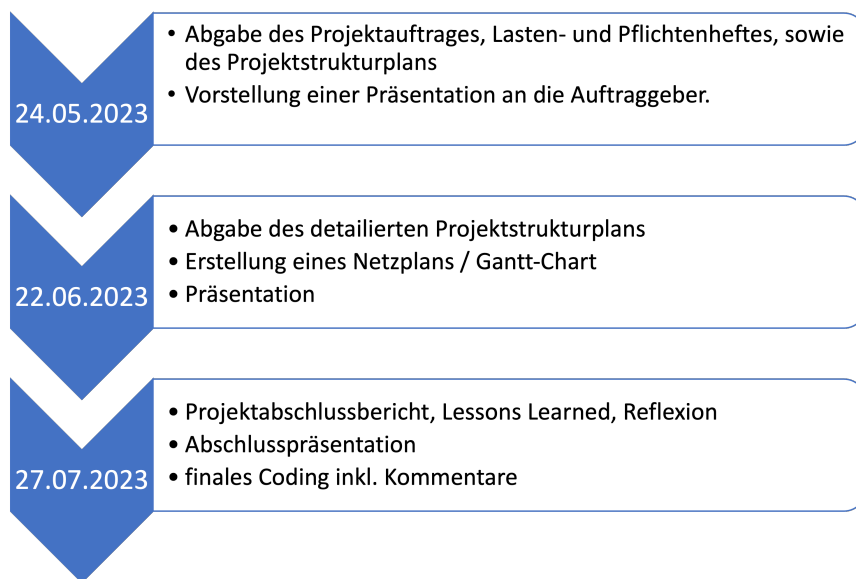


Abbildung 1.1: Projektmeilensteine. Quelle: Eigene Darstellung.

Bearbeitungszeit:

Die Bearbeitungszeit für das Projekt erstreckt sich von dem 08.05 bis zum 27.07. Dieser Zeitraum umfasst die Durchführung des Projekts parallel zu den regulären Vorlesungen der vier beteiligten Studenten. Es wird erwartet, dass die Studenten während dieses Zeitraums

ausreichend Zeit für das Projekt aufbringen und entsprechende Arbeitsstunden investieren.

Geplante Betriebszeiten:

Das entwickelte Tool zur Textzusammenfassung und -klassifikation soll grundsätzlich rund um die Uhr verfügbar sein, um den Nutzern eine maximale Flexibilität zu bieten. Es wird angestrebt, das Tool auf einer geeigneten Serverinfrastruktur mithilfe von Streamlit zu hosten, um die Verfügbarkeit zu gewährleisten.

Geplante Arbeitszeiten:

Die geplanten Arbeitszeiten für die Studenten während des Projekts umfassen die Zeiten außerhalb der Vorlesungen und individuellen Verpflichtungen. Es wird erwartet, dass die Studenten in Absprache miteinander und unter Berücksichtigung ihrer individuellen Verfügbarkeit regelmäßige Arbeitszeiten festlegen, um das Projekt voranzutreiben. Es ist ratsam, regelmäßige Treffen oder virtuelle Konferenzen zu vereinbaren, um den Fortschritt zu besprechen, Fragen zu klären und gemeinsame Entscheidungen zu treffen.

Es ist wichtig zu beachten, dass die oben genannten Rahmenbedingungen als allgemeine Leitlinien dienen und bei Bedarf angepasst werden können. Es wird empfohlen, eine klare Kommunikation und Zusammenarbeit zwischen den Studenten sicherzustellen, um die Arbeitszeiten bestmöglich zu koordinieren und das Projekt erfolgreich abzuschließen.

1.5 Dekomposition des Gesamtsystems

Das Gesamtsystem wird nachstehend in Hauptkomponenten zusammenfassend unterteilt:

Benutzerschnittstelle: Eine benutzerfreundliche Oberfläche, die den Benutzern die Eingabe von Texten, die Steuerung des Zusammenfassungs- und Klassifikationsprozesses sowie den Zugriff auf die Ergebnisse ermöglicht. Bereitgestellt wird dies per Web-Applikation, welche mittels Streamlit entwickelt wird.

Externe Deepl API: Aufgrund der besseren Performanz der Modelle auf englischsprachigen Texten werden deutschsprachige Eingabetexte für die weitere Verarbeitung übersetzt.

Textkompression: Eine leistungsfähige Textanalyse-Engine, die in der Lage ist, Texte zu analysieren, Schlüsselinformationen zu extrahieren und automatische Zusammenfassungen zu generieren.

Klassifikationsmodul: Ein Modul, das die Texte basierend auf vordefinierten Kategorien oder benutzerdefinierten Klassifikationen einordnet.

Stimmungsanalyse: Eine weitere Klassifikation des Eingabetexts in die Kategorien positiv, neutral und negativ, um ein Stimmungsbild zu erhalten.

Datenablage: Eine persistente Speicherung der Eingabe- oder Ausgabeteixe, sowie der Kategorisierung findet nicht statt.

Eine feingranulare Aufgliederung des Gesamtsystems wird im Projektstrukturplan dargestellt.

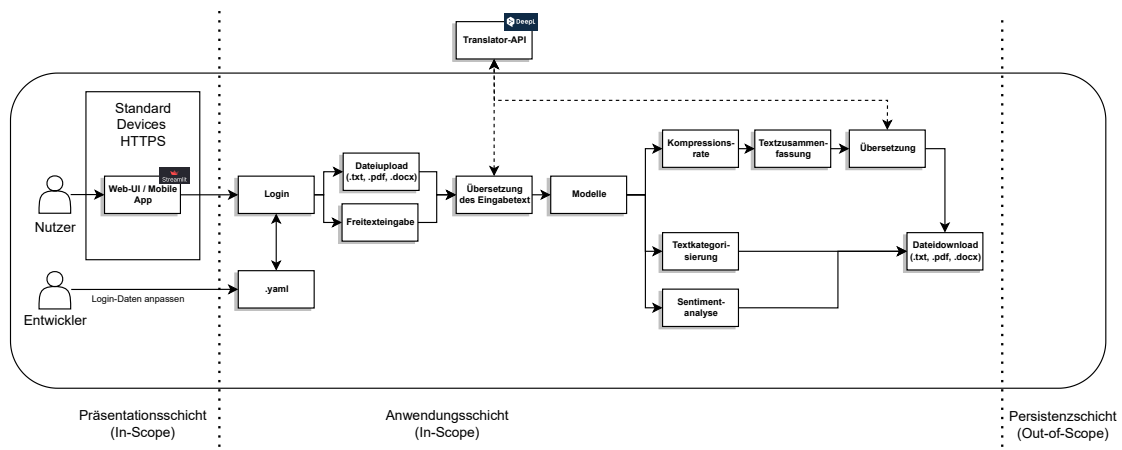


Abbildung 1.2: Vorläufige Systemskizze der geplanten Funktionalitäten. Quelle: Eigene Darstellung.

1.6 Bearbeitung von Änderungswünschen

Die vom Auftraggeber geforderten Änderungswünsche werden auf Basis eines Entscheidungsmusters bearbeitet. Grund dafür ist die eindeutige Dokumentation sowie Einhaltung etwaiger Projekt- und Zeitrahmen. Es wird dabei argumentativ entschieden, ob ein Änderungswunsch in die Anforderungen mit aufgenommen werden kann. Änderungswünsche seitens des Auftraggebers werden bis spätestens 19.06.2023 eingereicht werden. Folgender Ablauf für Change Request wird gewählt:

1. Change Requests werden als Änderungen an den Anforderungen, dem Umfang, den Funktionen oder anderen Aspekten des Projekts definiert, die nach der Festlegung im Lasten- und Pflichtenheft auftreten.

2. Change Requests können vom ausschließlich von den Auftraggebern eingereicht werden. Die Einreichung erfolgt schriftlich, per E-Mail. Dabei müssen relevante Informationen zu den vorgeschlagenen Änderungen angegeben werden.
3. Auswirkungen auf Umfang, die Zeitplanung, die Kosten und andere relevante Faktoren werden analysiert. Die Bewertung erfolgt durch das Projektteam unter Berücksichtigung der Projektziele und -prioritäten sowie Zeitrahmen.
4. Die endgültige Genehmigung von Change Requests erfolgt durch das für die Entwicklung beauftragte Projektteam. Dabei werden die Ergebnisse der Bewertung und die Auswirkungen auf das Projekt berücksichtigt. Die Genehmigung oder Ablehnung wird schriftlich kommuniziert und dokumentiert.
5. Genehmigte Change Requests werden in das Projekt integriert und umgesetzt. Die entsprechenden Änderungen werden dokumentiert und in den relevanten Projektunterlagen, einschließlich des Lasten- und Pflichtenhefts, festgehalten. Die Umsetzung erfolgt gemäß den definierten Projektprozessen und -verfahren.
6. Alle relevanten Informationen zu den genehmigten Change Requests werden ordnungsgemäß dokumentiert. Dies umfasst die Beschreibung der Änderungen, die Auswirkungen auf den Umfang, die Zeitplanung und die Kosten sowie etwaige Anpassungen in anderen Projektunterlagen.

1.7 Funktionale Anforderungen

REQ101 Das Produkt ermöglicht die Zusammenfassung von Texten

Umsetzung: Verwendung eines Transformer-basierten Modells zur automatischen Textzusammenfassung in der Web-Applikation inklusive Data Pipeline für notwendiges Preprocessing und Vorbereiten der Daten auf den Usecase. Dafür werden verschieden vortrainierte Transformer Modelle dem finetuning unterzogen, auf unseren Datensatz angepasst und deren Performance verglichen.

REQ102 Das Produkt ermöglicht die Eingabe von Freitexten, sowie einen Dateiupload von verschiedenen Dateitypen (.txt, .pdf und .docx)

Umsetzung: Implementierung durch Streamlit-Input-Widgets zur Erfassung von Textdaten wie auch dem Dateiupload von einer Dateigröße bis 200mb.

REQ103 Bei der Textkompression des Produkts handelt es sich um eine neu generierte Textzusammenfassung

Umsetzung: Anzeige von generierter Zusammenfassung auf Benutzeroberfläche. Wie in REQ101 beschrieben wird ein Transformer Modell, speziell für die abstraktive Zusammenfassung gewählt und finetuning vorgenommen. Abstraktiv wird gegenüber der extraktiven Zusammenfassung gewählt, da diese der Anforderung, neue Sätze zu generieren, gerecht wird und nicht nur einzelne prägnante Inhalte des originalen Textes extrahiert.

REQ104 Die Benutzeroberfläche ermöglicht eine individuelle Festlegung der Kompressionsrate zwischen 20 und 80 Prozent.

Umsetzung: Streamlit ermöglicht per Inputfeld die Auswahl des Wertes anhand eines verschiebbaren, stufenlosen Reglers.

REQ105 Das Ergebnis der Zusammenfassung wird in der Web-Applikation angezeigt.

Umsetzung: Anzeige von generierter Zusammenfassung auf Benutzeroberfläche. Output von Kompressionsmodell wird direkt dargestellt und nicht gespeichert.

REQ106 Die Benutzeroberfläche ermöglicht die Speicherung der Ergebnisse als Word, Text oder PDF-Datei.

Umsetzung: Einbindung in Layoutvorlage für den jeweiligen Dateityp und Download unter Zuhilfenahme etwaiger vordefinierter Python Libraries.

REQ107 Das System unterstützt die Verarbeitung von deutschen und englischen Texten.

Umsetzung: Diese, über den Projektscope hinaus angeforderte Funktionalität wird mit einer Anbindung, an die DeepL API realisiert. Das Tool bietet eine schon präzise Übersetzungsfunktion, welche jedoch noch auf den Anwendungsfall angepasst werden muss.

REQ108 In den Eingabetexten wird zwischen den Kategorien (emails/ news/ paper / poems / recipes / reviews) unterschieden

Umsetzung: Anhand der definierten Evaluationsmetriken werden Transformermodelle verglichen, um die beste Performance für den hier gewählten Usecase zu identifizieren. Die Evaluierung erfolgt auf einem Testdatensplit von einem Datensatz, der für diesen Usecase erstellt wurde.

REQ109 Die Oberkategorie des Eingabetextes wird angezeigt im UI der Web-Applikation

Umsetzung: Ergebnisse von Klassifikationsmodell werden auf der Benutzeroberfläche angezeigt

REQ110 Es ist möglich, ausschließlich Dateien der gegebenen Kategorien hochzuladen

Umsetzung: Der Textklassifikator erkennt keine anderen, als die vorgegebenen Klassifizierungen.

REQ111 Das Produkt führt eine Stimmungsanalyse des Eingabetextes durch.

Umsetzung: Die Stimmungsanalyse wird ebenfalls mit einem Transformer Modell entwickelt, umgesetzt und angewendet.

1.7.1 Sonstige Anforderungen (Kann)

REQ501 Das System unterstützt die Eingabe mittels Sprache

Umsetzung: Durch ein Streamlit Widget wird über den Browser Zugriff auf das Mikrofon angefragt, wird dies erlaubt, so kann per Start/Stop Button eine Aufnahme gestartet werden. Diese wird im Backend durch das Modell "whisper-small" von openAI verarbeitet.

1.8 Nicht-funktionale Anforderungen

REQ301 Zugang zur Web-Applikation ist nur bestimmten Nutzergruppen möglich

Umsetzung: Login über Benutzername und Passwort, welche mithilfe einer .yaml Datei (beim Prototyp) gespeichert werden

REQ302 Die Web-Applikation muss keinem Employer Branding entsprechen

Umsetzung: Nutzung von Streamlit-komponenten mit der Möglichkeit grafische Anpassungen vorzunehmen

REQ303 Die Web-Applikation sollte eine Benutzerfreundlichkeit aufweisen und ein intuitives Benutzerinterface bieten.

Umsetzung: Die Anwendung wird nach den Grundsätzen zeitgemäßer Benutzeroberflächen bzw. Webdesigns entwickelt und gemäß der ISO 9241-110 bereitgestellt.

REQ304 Das System muss bei Bedarf verfügbar sein.

Umsetzung: Hosting per Streamlit Cloud und Zugriff über öffentliche URL

REQ305 Die Web-Applikation sollte auf verschiedenen Browsern und Endgeräten (Desktop, Tablet, Mobil) gut funktionieren und ein responsives Design aufweisen.

Umsetzung: Streamlit ermöglicht das Verwenden der Applikation auf allen gängigen Endgeräten.

REQ306 Die Qualität der Ergebnisse soll bewertet werden.

Umsetzung: Vergleich der Transformermodelle anhand der oben genannten Evaluierungsmetriken.

1.9 Qualitätsüberprüfung

Die Qualität des entwickelten Tools zur Textzusammenfassung und -klassifikation spielt eine entscheidende Rolle für den Erfolg des Projekts. Um sicherzustellen, dass die definierten Qualitätsanforderungen erfüllt werden, werden klare Abnahmekriterien und ein strukturiertes Vorgehen zur Ausgangsprüfung festgelegt. Im Folgenden werden diese Aspekte im Detail beschrieben.

1.9.1 Abnahmekriterien

Die Abnahmekriterien dienen als Richtlinien, anhand derer die Qualität des Tools bewertet wird. Die spezifischen Abnahmekriterien werden in Absprache mit den Projektbeteiligten festgelegt und sollten quantifizierbare und messbare Merkmale umfassen. Beispiele für Abnahmekriterien könnten sein:

Funktionalität: Das Tool erfüllt die definierten funktionalen Anforderungen, einschließlich der Textzusammenfassung und -klassifikation von Inhalten.

Genauigkeit: Das Tool erzielt eine hohe Genauigkeit bei der Zusammenfassung und Klassifikation von Texten. Basierend auf den gewählten Evaluierungsmetriken wird hierbei die Genauigkeit bestimmt.

Benutzerfreundlichkeit: Das Tool ist intuitiv und einfach zu bedienen. Dazu werden 4–5 Personen die Verwendung des Tools beispielhaft durchführen und Feedback geben.

Leistung: Das Tool zeigt eine angemessene Reaktionsgeschwindigkeit und Skalierbarkeit. Die Zusammenfassung und Klassifizierung des Textes soll in keinem Fall länger als eine Minute andauern.

Stabilität: Das Tool läuft stabil und weist keine schwerwiegenden Fehler/Abstürze auf. Hierfür wird ausführlich getestet, ob Fehler auftreten und ob das System dauerhaft verfügbar ist.

1.9.2 Vorgehen zur Ausgangsprüfung:

Die Ausgangsprüfung dient dazu, die Qualität des entwickelten Tools vor der endgültigen Abnahme zu überprüfen. Das Vorgehen zur Ausgangsprüfung umfasst typischerweise die folgenden Schritte:

Testplanung: Es wird ein detaillierter Testplan erstellt, der die verschiedenen Testfälle, Szenarien und Daten abdeckt, um eine umfassende Überprüfung der Funktionalität, Genauigkeit, Benutzerfreundlichkeit, Leistung und Stabilität des Tools zu gewährleisten.

Durchführung von Tests: Die definierten Testfälle werden systematisch und methodisch durchgeführt. Dies umfasst sowohl manuelle Tests als auch automatisierte Tests, um sicherzustellen, dass alle Aspekte des Tools gründlich überprüft werden.

Fehlerbehebung: Wenn während der Tests Fehler oder Abweichungen von den definierten Abnahmekriterien festgestellt werden, werden diese dokumentiert und an das Entwicklungsteam gemeldet. Das Team ist dann dafür verantwortlich, die Fehler zu beheben und das Tool entsprechend zu optimieren.

Review und Abnahme: Nach Abschluss der Tests wird das Tool einer umfassenden Bewertung unterzogen, bei der die Einhaltung der Abnahmekriterien und die Erfüllung der definierten Qualitätsanforderungen bewertet werden. Wenn das Tool die festgelegten Kriterien erfüllt, erfolgt die formelle Abnahme. Die Abnahme des Projektes erfolgt zusammen mit den Auftraggebern, nachdem die Vollständigkeit des Systems zur Kenntnis genommen wurde und dieses für die Verwendung einsatzbereit ist.

1.10 Lieferumfang

Der Lieferumfang umfasst das vollständige entwickelte System, einschließlich der Benutzeroberfläche, der Textanalyse-Engine, des Klassifikationsmodells und der Datenbank. Darüber hinaus sollten auch eine umfassende Dokumentation, Installationsanweisungen und erforderliche Schulungsunterlagen bereitgestellt werden. Die ausführliche Aufschlüsselung der Projektentwicklung inklusive der Teilschritte erfolgt im Projektstrukturplan.

2 Gegenüberstellung der Projektkriterien

Dieser Abschnitt der Arbeit gilt als einleitendes Kapitel in die finale Projektübergabe an die relevanten Auftraggeber. Es werden die zu Beginn geplanten Anforderungen, respektiv unterteilt in Kann und Muss Anforderungen, erneut aufgegriffen und bestätigt oder abgelehnt. Die Projektübergabe findet im folgenden Kapitel 3 statt.

Im Laufe der Entwicklung und Umsetzung aller Anforderungen wurden diese im Detail unterteilt in ihre für den Projekterfolg relevante Priorität zugeteilt. Demnach sind Anforderungen der „Prio 1“ als solche zu behandeln und im Rahmen des Minimal Viable umzusetzen.

2.1 Erfüllte Kriterien

2.1.1 Prio 1 Anforderungen

REQ101: Die Zusammenfassung von Texten ist vollständig umgesetzt und findet automatisch anhand von transformer-basierten Modellen in der übergebenen Web-Applikation statt. Auch die geforderte Datenpipeline für das notwendige Preprocessing und Vorbereiten der Daten, sowie des Finetuning dieser sind Teil der übergebenen Programmcodebasis.

REQ102: Die Web-Anwendung unterstützt ein Freitextfeld zur Eingabe von Freitexten, sowie eine Funktion für den Dateiupload der Dateitypen .docx, .pdf, .txt

REQ104: Die Benutzeroberfläche der Web-Applikation unterstützt eine individuelle Festlegung der Kompressionsrate. Das Streamlit Inputfeld ermöglicht die Auswahl des Wertes anhand eines verschiebaren, stufenlosen Reglers. Die Kompressionsrate ist limitiert auf einen Wert zwischen 20 und 80 Prozent.

REQ105: Der zusammengefasste Ergebnistext wird auf der Web-Applikation ausgegeben und ist dort ersichtlich. Der Output des Kompressionsmodells wird standardmäßig in der Applikation angezeigt und nicht automatisiert abgespeichert.

REQ106: Die Speicherung der Outputs des Kompressionsmodells kann in den Dateiformaten .docx, .pdf und .txt anhand einer dedizierten Funktion auf der Benutzeroberfläche der Web-Applikation stattfinden.

REQ107: Die Eingabe verschiedenster Sprachen ist möglich sowie die Ausgabe in Deutsch, englisch, französisch, italienisch und spanisch

REQ108: Die Klassifikation der Kategorien 'Emails, News, Paper, Poems, Recipes, Reviews' wird von der übergebenen Web-Applikation unterstützt. Das umfasst alle definierten Kategorien der Projektanforderung.

REQ109: Die Ausgabe des Klassifikationsmodells der Oberkategorien wird in der Benutzeroberfläche der Web-Applikation dargestellt.

REQ110: Ausschließlich Dateien der bereits definierten Dateitypen werden im Rahmen der Klassifikation und Kompressions unterstützt. Auch die Sentiment-Analyse unterstützt lediglich diese Formate.

REQ111: Die Stimmungsanalyse, Sentiment-Analyse, wird durch die Web-Applikation auf den Eingabetext durchgeführt und als Ausgabe in der Benutzeroberfläche angezeigt. Die Stimmungsanalyse wurde ebenfalls mit einem transformer-basierten Modells erstellt.

REQ301: Wurde vollständig erfüllt. Der Zugang ist auf bestimmte, vorab definierte Nutzergruppen beschränkt.

REQ305: Die Web-Applikation ist auf verschiedenen Browsern und Engeräten, wie z.B. Desktop, Tablet oder Mobil, verfügbar und ist im Rahmen eines responsive Designs umgesetzt.

REQ306: Die Modellgüte wurde bewertet und ist Teil der übergebenen Codebasis, sowie auch der Dokumentation.

2.1.2 Prio 2 Anforderungen

REQ501: Die Web-Applikation unterstützt die Spracheingabe von Freitexten anhand einer externen Speech-To-Text Bibliothek und API.

REQ302: Wurde vollständig erfüllt.

REQ303: Die Benutzeroberfläche wurde möglichst intuitiv und benutzerfreundlich gestaltet. Diese Anforderung wurde als Prio 2 Anforderung eingestuft, da diese nicht durch die Auftraggeber spezifisch angefordert war.

2.2 Nicht erfüllte Kriterien

2.2.1 Prio 1 Anforderungen

REQ304: Die funktionale Anforderung des Hostings der Website und der trainierten Machine Learning Modelle. Problematisch hierbei war die Tatsache, dass die gewählte Option bzw. der Hostinganbieter Streamlit die Bezahloption für das Hosting von Server abgeschafft hat. Dies war zu Beginn des Projektes nicht der Fall und diese Änderung nicht absehbar. Das Hosting ist auf 1 GB Speicher begrenzt, wobei der Download aller für die Anwendung notwendigen Modelle diese Begrenzung weit übersteigt. Die Zwischenlösung ist aktuell das lokale Starten des Servers auf dem Port localhost:8000, mit einer potenziellen Lösung als Ausblick, die Streamlit App über eine AWS oder Azure Instanz betreiben zu lassen.

2.2.2 Prio 2 Anforderungen

REQ103: Eine grundlegende, eigen-gestellte Anforderung an die Textkompression des Produkts war es, dass die Ausgabe der Zusammenfassung ein neu generierter, paraphrasierter Text ist. Extraktive Zusammenfassung: Es werden die Hauptmerkmale extrahiert und in einem logischen Zusammenhang dargestellt. Eine weitere Anforderung war es, dass transformer-basierte Modelle verwendet werden. Da es keine open-source Transformer-Modelle für die abstraktive Zusammenfassung gibt, wird in dem übergebenen Produkt eine Kompromisslösung gewählt. Zuerst wird eine extraktive Zusammenfassung des Eingabetexts durchgeführt. Der Output des extraktiven Modells wird durch mehrfaches Übersetzen über die Deepl API paraphrasiert. Die Ausgabe ist somit eine komprimierte, paraphrasierte Zusammenfassung des Eingabetexts, ohne, dass der Einsatz eines separaten Modells für die Paraphrasierung notwendig ist.

REQ102: Eine weitere grundlegende, eigen-gestellte Anforderung im Projektscope war die Möglichkeit, eine Datei des Formats .doc einzulesen. Im Rahmen des Changelagements wurde von den Auftraggebern gefordert, dies um das Format .docx zu erweitern. Während

.docx implementiert werden konnte, handelt es sich bei dem Dateityp .doc um ein veraltetes Format, mit dem die verwendeten Libraries nicht arbeiten konnten. Demnach wurde sich entschieden, dass dieser Dateityp nicht umgesetzt werden sollte. Aufgrund der Umsetzung des Dateiuploads und -downloads von .docx Dateien wurden dies als nicht erfüllte Prio 2 Anforderung eingestuft.

3 Projektübergabe

Dieser Abschnitt der Projektdokumentation dient der vollständigen Übergabe des finalen Produkts. Ausgehend von der im vorherigen Kapitel zwei beschriebenen Gegenüberstellung der erfüllten und nicht erfüllten Anforderungen werden hier die Funktionalitäten in Form eines Benutzerhandbuchs sowie die Verbesserungsmöglichkeiten - umgesetzte und mögliche Umsetzungsvorschläge - dargestellt. Unser System ist das Ergebnis einer Kombination maschineller Lernmodelle, die speziell für verschiedene Anwendungen im Bereich Natural Language Processing entwickelt wurden. Konkret basiert unsere Lösung auf den folgenden renommierten Modellen: ProsusAI/finbert, cardiffnlp/twitter-roberta-base-sentiment-latest, nlptown/bert-base-multilingual-uncased-sentiment und siebert/sentiment-roberta-large-english, microsoft/deberta-base-mnli, bert-base-uncased xlm-roberta-large, sowie die Modelle distilbert-base-uncased und flan-t5-base. Die Auswahl dieser Modelle gewährleisten eine breite Abdeckung von Funktionen und Fähigkeiten, die für diese Anwendung unerlässlich sind.

3.1 Bewertung der Modellgüte

Ein entscheidender Aspekt des übergebenen Produkts ist die Güte der verwendeten Modelle. Das übergebene Produkt enthält drei separate Modelle, die im Rahmen der Übergabe evaluiert werden müssen. Das verwendete Gütemaß, bei der Textclassification und der Textsentiment, war der Accuracy Score. Ein Accuracy Score (oder einfach Accuracy) ist hierbei ein Klassifizierungsmaß beim maschinellen Lernen, das den Prozentsatz der richtigen Vorhersagen eines Modells angibt. Ausgewählt wurde diese Metrik aufgrund der Einfachheit in der Berechnung und Interpretation. Somit war es letztendlich möglich die Leistung des Modells durch eine einzige Zahl zu quantifizieren. Bei der Textsummarization wurde der Rouge Score als Gütermaß eingesetzt. Beim Rouge Score wird eine automatisch erstellte Zusammenfassung mit einer Referenz (von Menschen erstellte) Zusammenfassung verglichen und mathematisch bewertet. Hierdurch kann numerisch festgestellt werden wie ähnlich sich die beiden Texte sind. Die Ergebnisse der jeweiligen Evaluierungen werden folgend dargestellt.

3.1.1 Evaluierung Textsummarization

Bei der Textsummarization wurden zwei verschiedene Ansätze miteinander verglichen. Hierbei wurde zum das Modell „distilbert-base-uncased“ getestet, sowie eine Erweiterung des Modells um ein Paraphrasing Modell namens „flan-t5-base“. Um die Modelle in der Evaluierung vergleichen zu können wurde die Rogue Metrik verwendet. Folgende Ergebnisse wurde erzielt:

| Modell | Rogue Score | Zeit |
|----------------------------------------|-------------|--------|
| distilbert-base-uncased | 0.177 | 18min |
| distilbert-base-uncased + flan-t5-base | 0.194 | 122min |

Tabelle 3.1: Ergebnisse der Trainingsepochen der Modellevaluierung der Textzusammenfassung

Auffällig ist hierbei der zeitliche Unterschied der Modelle so hat das kombinierte Modell fast die sechsfache Zeit für die selbe Datengrundlage benötigt. Für diesen großen Zeitunterschied unterscheiden sich die Rogue Scores der beiden Modelle jedoch nur gering. Der generelle niedrige Rogue Score ist auf den genutzten Datensatz zurückzuführen. Bei diesem wurde keine Kompressionsrate bei der „richtigen“ Zusammenfassung beigefügt. Daher konnte man diese nur raten. Bei dieser Evaluierung wurde eine Kompressionsrate von 0,6 gewählt. Durch die gleiche Kompressionsrate und die gleiche Datengrundlage sind die beiden Modell-Ansätze vergleichbar. Durch den zeitlichen Unterschied der Modelle und geringfügigen Unterschied der Modelle wird in der Webapplikation das „distilbert-base-uncased“ genutzt. Ausschlaggebend war hierbei der zeitliche Aspekt um eine schnellere Bereitstellung der Zusammenfassung zu gewährleisten.

3.1.2 Evaluierung Textclassification

Um den Eingabetext des Benutzer in die jeweilige Oberkategorie zu klassifizieren wurden verschiedene das transformerbasierte Modelle mit Hilfe der Accuracy verglichen. Die Evaluierung ergab die folgenden Ergebnisse:

| Modell | Accuracy |
|-----------------------------|----------|
| bert-base-uncased | 0.979 |
| xlm-roberta-large | 0.988 |
| microsoft/deberta-base-mnli | 0.997 |

Tabelle 3.2: Ergebnisse der Modellevaluierung der Textklassifikation

Alle Ergebnisse der Evaluierungen waren extrem zufriedenstellend, wodurch auch von einem Hyperparameter Tunings abgesehen wurde. Das finegetunte „microsoft/deberta-base-mnli“ wurde am Ende der Evaluierung exportiert und in der Web-Applikation verwendet. Grund hierfür war der höchste Accuracy Score.

3.1.3 Evaluierung Textsentiment

Im Rahmen der Stimmungsanalyse wurden die folgenden Modelle verwendet, die es zu evaluieren galt: ProsusAI/finbert, cardiffnlp/twitter-roberta-base-sentiment-latest, nlptown/bert-base-multilingual-uncased-sentiment und siebert/sentiment-roberta-large-english.

Die Evaluierung ergab folgende Ergebnisse:

| Modell | Accuracy |
|--------------------------------------------------|----------|
| ProsusAI/finbert | 0.819 |
| cardiffnlp/twitter-roberta-base-sentiment-latest | 0.855 |
| nlptown/bert-base-multilingual-uncased-sentiment | 0.816 |
| siebert/sentiment-roberta-large-english | 0.867 |

Tabelle 3.3: Ergebnisse der Modellevaluierung der Textsentiment Analyse

Aufgrund der bereits ausreichend hohen Güte, des Accuracy Scores, von 0.819, wurde bei der Textsentiment Analyse auch auf weiteres Hyperparameter-tuning verzichtet. In der Web-Applikation wurde auf Grund des höchsten Accuracy Scores das „siebert/sentiment-roberta-large-english“ verwendet.

3.2 Übergabe der technische Anwendung

Die Übergabe der technischen Anwendung umfasst folgend die Verbesserungsmöglichkeiten in Abschnitt 3.3 und das Benutzerhandbuch des folgenden Kapitels 4. Grundsätzlich wird das Projekt als Docker-File übergeben, wodurch die nötige Umgebung zur Ausführung der Anwendung bereits mitübergeben wird. Die manuelle Installation der notwendigen Bibliotheken, Modelle oder Dependencies ist somit nicht durchzuführen. Das containerized Projekt sollte ausgeführt werden. Die genauen Schritte zur Ausführung und Hosting der Anwendung sind in der Git-Dokumentation des Projekts ersichtlich. GitHub-Repository: <https://github.com/Projektrealisierung-Gruppe-1/Codebase>.

3.3 Verbesserungsmöglichkeiten

Die Verbesserungsmöglichkeiten des Produkts können sowohl in die Erweiterung der bestehenden Anwendung um zusätzliche Funktionalitäten als auch in die Umsetzung der nicht erfüllten Anforderungen unterteilt werden. In diesem Abschnitt werden diese Punkte aufgegriffen und Vorschläge dokumentiert, sowie eine Beschreibung der bereits umgesetzten Verbesserungen und Erweiterungen gegeben. Eine der zentralen Anforderungen, die nicht umgesetzt wurden, ist das Hosting der Web-Applikation über einen eigenen Server oder über einen externen Dienstleister [REQ304]. Vollständig umgesetzt im aktuellen Stand des Produkts ist das lokale Hosting der Web-Applikation in Form einer Streamlit Anwendung, welche Teil des Dockers ist. Die Grenze von kostenlosen Hosting-Angeboten, wie z.B. Amazon AWS, Microsoft Azure oder Streamlit beträgt zwischen 5 und 10GB. Diese werden von dem Produkt überschritten.

Mögliche Lösungsschritte:

1. Da das User Interface der Anwendung bereits besteht, könnte dies innerhalb des firmeninternen Netzwerks gehostet beziehungsweise zur Verfügung gestellt werden. Dies umfasst wenig Aufwand, da der Docker-File des Projekts lediglich einzulesen ist.
2. Eine weitere Möglichkeit ist der Einkauf einer Drittanbieterlösung, eines Host-Providers. Die berechneten Kosten liegen, am Beispiel des Hosting-Angebots von Microsoft Azure, bei 127 Euro / Monat. Die Auftraggeber könnten sich entscheiden, dass diese Kosten tragbar für die Funktionalität der Anwendung sind.

Um die erhöhten Kosten für das Projekt zu umgehen, empfiehlt sich somit das Einbeziehen der lokalen Serverleistungen und das Hosting der Web-Applikation auf den unternehmensinternen Servern. Sollte dies nicht möglich sein, steht ein weiterer Lösungsweg über die Nutzung von Host-Providern zur Verfügung.

Bereits vorgenommene Verbesserungen und Erweiterungen umfassen die Umsetzung der Kann-Anforderung [REQ501] und [REQ107]. Die Web-Applikation bietet die Funktionalität der Spracheingabe, Speech-to-text, für das Freitextfeld. Auch die Möglichkeit für die Änderungen der Farbkontraste der Web-Applikation, sodass diese den Web content accessibility guidelines (WCAG2.0) Anforderungen entsprechen, konnte umgesetzt werden. Außerdem unterstützt das Produkt die mehrsprachige Eingabe und den Upload von Dokumenten und Texten. Die genauen Informationen über die Benutzung und zur Verfügung stehenden Funktionalitäten sind im folgenden Kapitel 4, Benutzerhandbuch, einzusehen.

4 Benutzerhandbuch

In den folgenden Abschnitten bieten wir detaillierte Beschreibungen und Anleitungen, die darauf ausgerichtet sind, dem Endanwender den Einstieg in unser System zu erleichtern. Unser Ziel ist es, den Nutzern eine intuitive Bedienung zu ermöglichen und sicherzustellen, dass der erste Kontakt mit dem System positiv und problemlos verläuft. Hierdurch wollen wir eine effiziente und angenehme Nutzungserfahrung gewährleisten.

4.1 Systemanforderungen

Die Benutzeroberfläche des Systems wurde unter allen Aspekten der Benutzerfreundlichkeit und für eine einfache Navigation entwickelt.

- **Browserkompatibilität:** Streamlit ermöglicht die Darstellung und Verwendung des Tools auf allen gängigen Geräten und Browsern
- **Internetverbindung:** Notwendig für die Verwendung ist eine stabile ununterbrochene Internetverbindung

4.2 Benutzeroberfläche

Die Benutzeroberfläche des Systems wurde unter allen Aspekten der Benutzerfreundlichkeit und für eine einfache Navigation entwickelt.

- **Startbildschirm:** Der Startbildschirm (Welcome Page) ermöglicht zunächst den Überblick auf der rechten Seite des Bildschirms, die Auswahl der erforderlichen Funktionalität. Dafür kann mittels Dropdown Menü entweder Textzusammenfassung oder die Klassifizierung gewählt werden.
- **Menüleiste:** Über den Rand an der linken Bildschirmseite kann ausgewählt werden, welche Seite angezeigt werden soll (Landing, App, Themeanpassung)
- **Dateiupload:** Über den Button „Dateiupload“ können Dateien der Typen .pdf, .txt und .docx hochgeladen werden.

- **Texteingabe:** Über das in der Mitte der Anwendung zu findende Feld kann beliebig Text eingetragen bzw. eingefügt werden.
- **Speech-to-Text:** Soll die S2T Funktionalität genutzt werden, so ist die Aufnahme mithilfe des „Start/Stopp“ Knopfes zu bedienen.
- **Funktionen ausführen:** Die Ausführung der vom System ermöglichten Funktionalitäten erfolgt über die Auswahl der Zielsprache per Auswahl aus dem Menü und der anschließenden Wahl der Kompressionsrate mithilfe des Reglers.
- **Dateidownload:** Im Anschluss kann ausgewählt werden, in welchem Dateiformat die Ergebnisse heruntergeladen werden sollen (.pdf, .docx und .txt).

4.3 Anwendungsanleitung

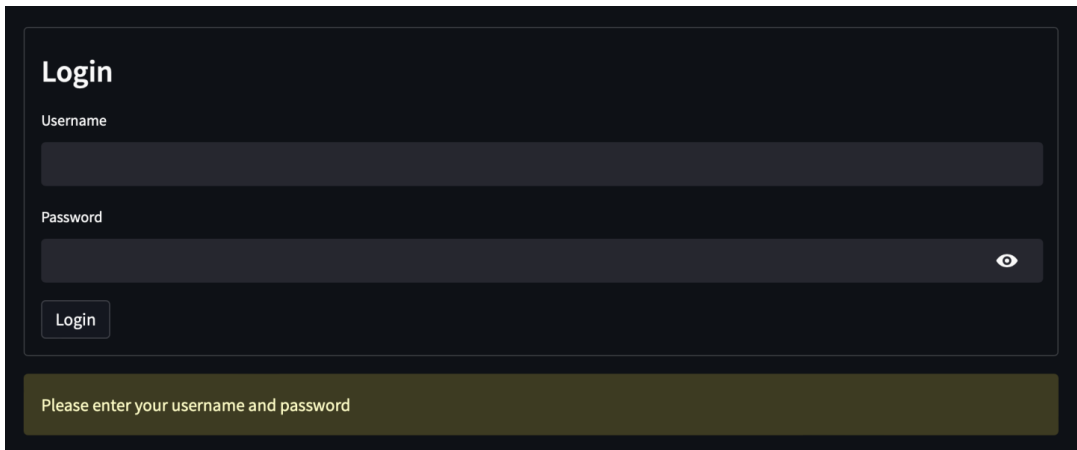


Abbildung 4.1: Login-Funktion der Web-Applikation

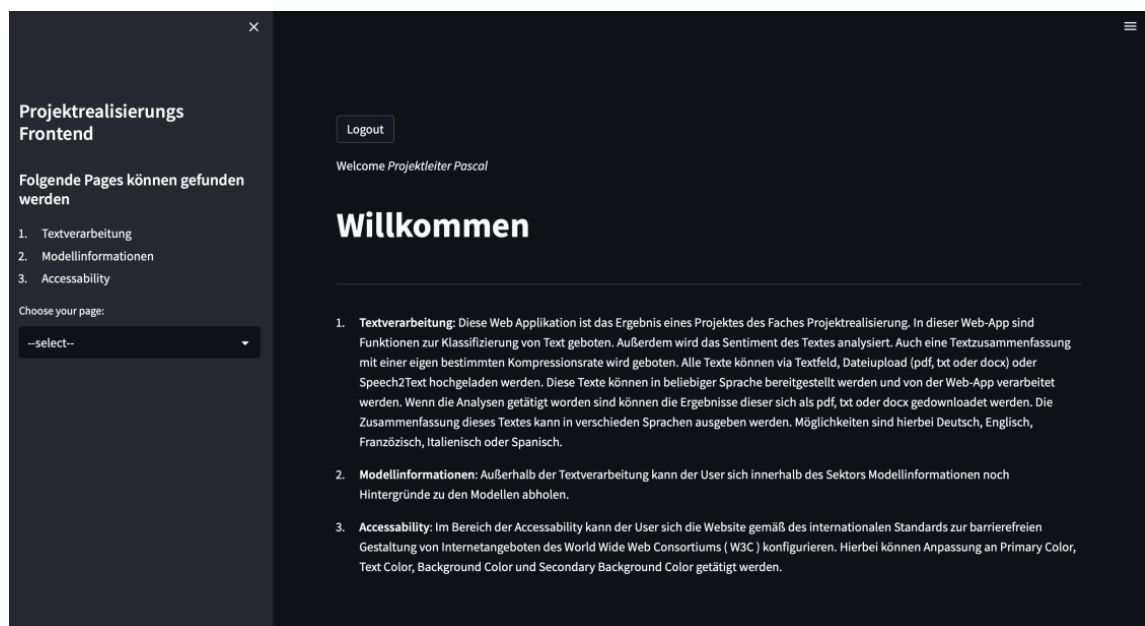


Abbildung 4.2: Welcomepage der Web-Applikation

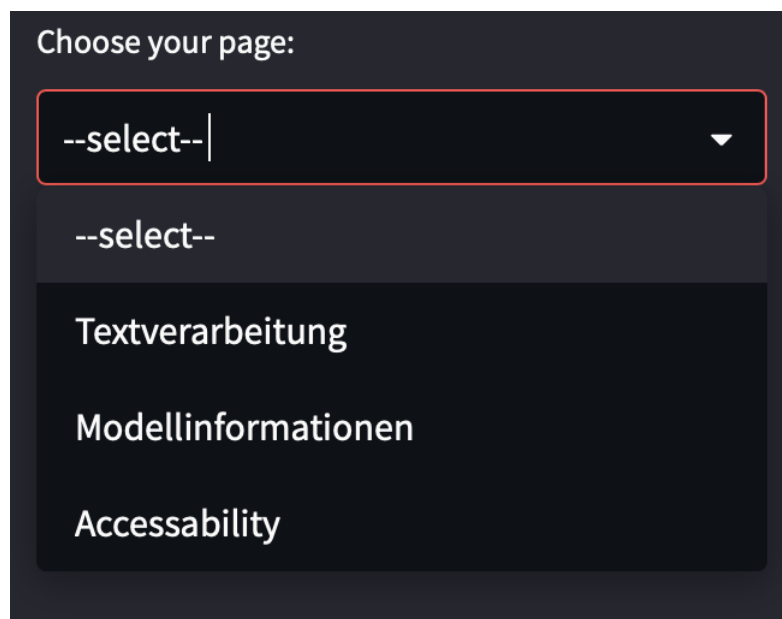


Abbildung 4.3: Drop-down Menu der Web-Applikation

Text Gegebenheit

Dateneingabe

☒ Textfeld ☐ Dateiupload ☐ Speech to Text

Your text:

Abbildung 4.4: Freitexteingabe der Benutzeroberfläche der Web-Applikation

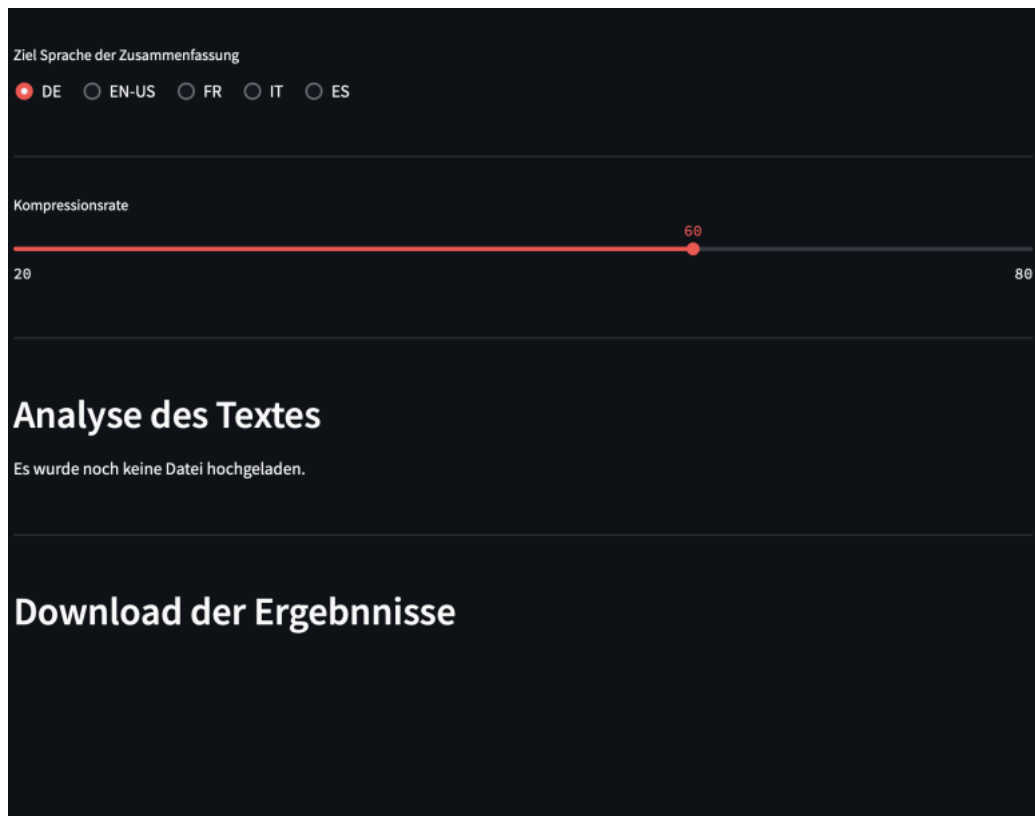


Abbildung 4.5: Anpassbarer, stufenloser Regler für die Anpassung der Kompressionsrate; Auswahl der Ziel-Sprache und Outputfeld der Web-Applikation

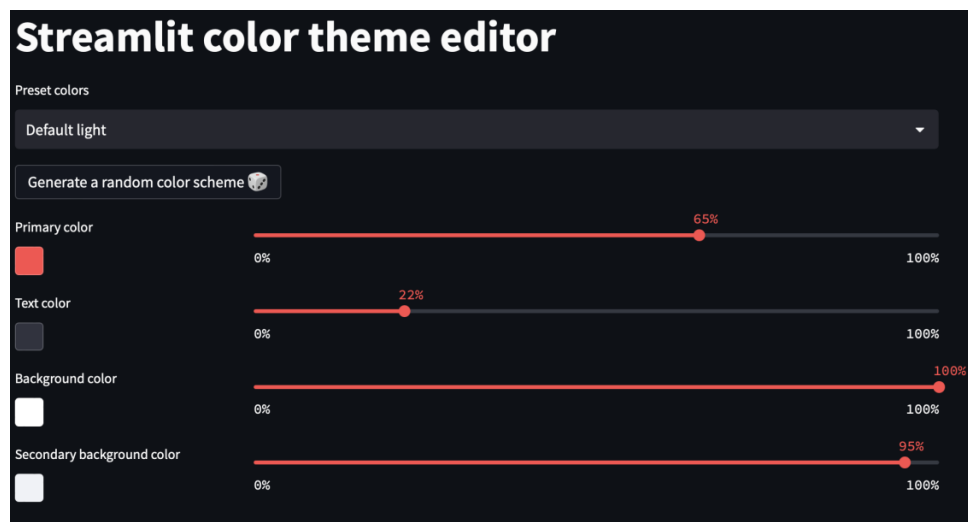


Abbildung 4.6: Kontraständerung anhand der WCAG2.0 Guidelines der Web-Applikation

4.4 Modellverwendung

Diese laufen im Hintergrund und werden automatisiert geladen und bereitgestellt. Sobald die Eingabe eines Textes, über den Datei-Upload oder die Freitexteingabe, stattgefunden hat und die Funktion des Modells beziehungsweise der Anwendung gestartet wurden, verarbeiten diese die Input-Informationen und geben diese auf der Benutzeroberfläche aus. Die Ergebnisse (Zusammenfassung, Klassifikation und Stimmungsbild) werden darüber hinaus zurückgehalten, solange die Anwendung geöffnet ist und kein neuer Input gegeben wurden und stehen zum Dateidownload in einem gewünschten Format (.docx, .txt, .pdf) zur Verfügung. Dieser Download muss ebenfalls manuell gestartet werden. Die Kompressionsrate ist anhand des stufenlosen Reglers auf dem UI manuell zwischen den Werten 20 und 80 Prozent anzupassen und entspricht der prozentualen Reduktion des Textes anhand der Wortzahl. Die Klassifizierung umfasst die Oberkategorien: emails, news, paper, poems, recipes und reviews. Das Stimmungsbild ordnet den Eingabetext in die folgenden Klassen ein: negative, neutral und positiv.

5 Fazit und Rückblick

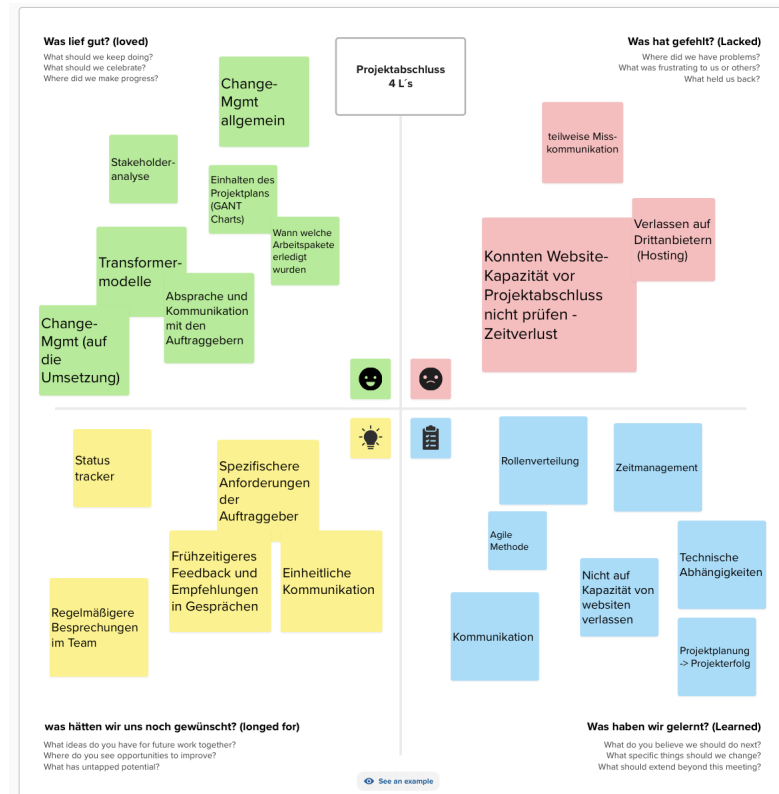


Abbildung 5.1: Auszug aus Retrospektive Session

5.1 Reflexion

Im Rückblick auf unser Projekt haben wir uns auf die zentralen Perspektiven der Scrum Retrospektive konzentriert: Was lief gut (Liked)? Was hat gefehlt (Lacked)? Was hätten wir uns gewünscht (Longed for)? Was haben wir gelernt (learned)?

Diese Ansätze ermöglichen uns, unser Vorgehen und unsere Ergebnisse tiefgehend zu reflektieren und Wege für zukünftige Verbesserungen zu identifizieren. Das vierte L für Lessons Learned, findet sich im nächsten Kapitel wieder. Im Zuge der Reflexion unseres Projekts haben wir eine Retrospektive-Session durchgeführt, um unsere Erfahrungen und Erkenntnisse nochmals zu bewerten und die 4L's – Liked, Lacked, Learned, Longed for – detailliert festzuhalten.

Was lief gut?

Die Einführung von Kanban hat sich als positiver Schritt in unserem Arbeitsprozess erwiesen. Dadurch wurde nicht nur die Kommunikation innerhalb des Teams effizienter gestaltet, sondern auch die Transparenz einzelner Arbeitsschritte erhöht. Zudem haben wir es geschafft, mit Veränderungen proaktiv umzugehen und diese als Chance zur Optimierung zu nutzen, anstatt sie als Störfaktor zu betrachten. Obwohl wir auf unerwartete technische Schwierigkeiten gestoßen sind, konnten wir uns schnell anpassen und alternative Lösungswege finden. Dies zeigt unsere Fähigkeit zur Agilität und Problemlösung. Wir erhielten wertvolles Feedback von unseren Kunden/Stakeholder, das uns half, unsere Lösungen zu verfeinern und besser auf ihre Bedürfnisse zuzuschneiden.

Was hat uns gefehlt?

Trotz unserer robusten Teamkommunikation erlebten wir Momente, in denen Informationen durchs Raster fielen oder zu Missverständnissen führten. Insbesondere unsere anfänglichen Wissenslücken bezüglich bestimmter technischer Anforderungen verursachten Verzögerungen, die durch eine detailliertere technische Einführung zu Projektbeginn hätten vermieden werden können.

Was hätten wir uns gewünscht?

Während unserer ausführlichen Projektreflexion wurde deutlich, dass trotz aller Erfolge und Erkenntnisse noch bestimmte Wünsche und Bedürfnisse im Raum standen. Mit Blick auf die Projektdurchführung und -kontrolle wäre beispielsweise ein Status Tracker ein unschätzbares Instrument gewesen, um den Projektfortschritt lückenlos nachvollziehen zu können. Dazu hätte eine höhere Frequenz an Statusupdate-Calls innerhalb des Teams maßgeblich beigetragen, die uns geholfen hätten, jederzeit alle Beteiligten synchron zu halten. Dies hätte nicht nur mögliche Missverständnisse minimiert, sondern auch Doppelarbeiten verhindert. Besonders wertvoll wäre zudem ein zeitnahes und ausgewogenes Feedback unserer Stakeholder gewesen, sowohl in positiver als auch in konstruktiv-kritischer Form. Solch ein unmittelbares Resonanzverhalten hätte es uns ermöglicht, Anpassungen agil und bedarfsgerecht in die laufende Entwicklung zu integrieren. Zuletzt wären präzisere und detailliertere Anforderungen zu Projektbeginn essenziell gewesen, um den Raum für Interpretationen zu reduzieren und so die Entwicklungsrichtung klarer zu steuern.

5.2 Lessons Learned

Während der Entwicklung und Implementierung unserer Anwendung stießen wir auf verschiedene technische und strategische bzw. organisatorische Herausforderungen, aus denen sich wertvolle Erkenntnisse für zukünftige Projekte ableiten ließen.

Organisatorisch

Im Rahmen des Projektes haben wir tiefgehende Einblicke in die Bedeutung und Anwendung effizienter Projektmanagement-Techniken gewonnen. Die zentrale Rolle, die das Zeitmanagement spielte, wurde uns insbesondere bei der Umsetzung unserer Aufgaben bewusst. Das strukturierte und gezielte Managen von Zeit ermöglichte es unserem Team, Prioritäten klar zu definieren, Engpässe in der Arbeit frühzeitig zu identifizieren und die gesteckten Ziele innerhalb der geplanten Fristen zu erreichen.

Parallel dazu hat uns die Einführung der agilen Methode, insbesondere des Kanban-Ansatzes, eine neue Perspektive auf die Aufgabenverteilung und -umsetzung eröffnet. Dies erleichterte die Kommunikation im Team erheblich und half, Transparenz bezüglich des Fortschritts einzelner Aufgaben zu gewährleisten. Dieses System hat uns nicht nur gezeigt, wo Engpässe oder Verzögerungen auftreten könnten, sondern es hat uns auch ermöglicht, proaktiv darauf zu reagieren und unsere Ressourcen effizienter zu nutzen.

In Retrospektive können wir festhalten, dass die Kombination aus gezieltem Zeitmanagement und der Verwendung des Kanban-Systems eine positive Wirkung auf unsere Arbeitsweise hatte. Durch diese Erfahrung sind wir überzeugt von der Wirksamkeit dieser Methoden und werden sie in zukünftigen Projekten mit Sicherheit wieder anwenden. Es war für uns eine eindrucksvolle "Lektion", wie mithilfe strukturierter Ansätze die Produktivität und Effizienz eines Teams gesteigert werden kann.

Im Laufe unseres Projektes stießen wir auf zahlreiche unerwartete und rapide Änderungswünsche seitens unserer Kunden und Stakeholder. Ursprünglich betrachteten wir diese als Störfaktor und Hindernis für unseren Fortschritt. Mit der Zeit haben wir jedoch erkannt, dass solche Anfragen eine Chance bieten, das Endergebnis besser an die tatsächlichen Bedürfnisse des Kunden anzupassen. Wir haben gelernt, dass es nicht ausreicht, lediglich auf Veränderungen zu reagieren; vielmehr müssen wir eine proaktive Haltung einnehmen, die uns in die Lage versetzt, Veränderungen vorwegzunehmen und flexibel darauf zu antworten. Ein weiterer wichtiger Aspekt unseres Lernprozesses betraf die Rollenverteilung innerhalb des Teams und die Bedeutung der Dokumentation. Anfangs führte eine unklare Rollenverteilung zu Überlappungen und Unstimmigkeiten bei den Zuständigkeiten. Als Reaktion darauf haben wir ein klareres Rollenkonzept entwickelt und festgelegt, wer für welche Auf-

gaben und Entscheidungen verantwortlich ist. Gleichzeitig wurde uns die immense Bedeutung der Dokumentation bewusst. Indem wir Prozesse, Entscheidungen und Begründungen konsequent festhielten, konnten wir sicherstellen, dass das gesamte Team stets auf dem gleichen Informationsstand war und mögliche Fehlerquellen durch mangelnde Information vermieden wurden.

Schließlich wurde uns im Verlauf des Projektes immer deutlicher, dass Kommunikation – sowohl innerhalb des Teams als auch im Umgang mit unseren Stakeholdern – das zentrale Element für den Projekterfolg ist. Eine offene, regelmäßige und klare Kommunikation ermöglichte es uns, Missverständnisse zu vermeiden, Erwartungen zu managen und sicherzustellen, dass alle Beteiligten ein gemeinsames Verständnis für die Projektziele und -methoden hatten. Besonders im Bereich des Stakeholder-Managements wurde uns bewusst, wie essenziell eine proaktive und transparente Kommunikation ist, um Vertrauen zu schaffen und sicherzustellen, dass alle Beteiligten in dieselbe Richtung arbeiten.

Technisch

Im Verlauf unseres Projektes erhielten wir außerdem bedeutende Erkenntnisse in Bezug auf technische Aspekte und ihre Implikationen für den Entwicklungsprozess gewinnen. Zunächst stellten wir fest, dass das Training unseres Modells effektiver und effizienter durch den Einsatz von High Performance Computing (HPC) durchgeführt werden könnte. Jedoch trafen wir auf unerwartete Schwierigkeiten mit den HPC-Servern, die zu Verzögerungen im Trainingsprozess der Modelle führten. Diese Erfahrung betonte die Notwendigkeit, die Zuverlässigkeit und Verfügbarkeit von HPC-Infrastrukturen im Voraus gründlich zu überprüfen und alternative Ressourcen in Betracht zu ziehen, um potenzielle Ausfälle zu kompensieren. Ein weiteres kritisches Problem, mit dem wir konfrontiert wurden, betraf die Verwendung bestimmter Bibliotheken. Die Durchführung lokaler Updates dieser Bibliotheken brachte das gesamte System durcheinander. Dies unterstreicht die Relevanz einer sorgfältigen Versionskontrolle und der Beachtung von Kompatibilitätsfragen bei der Aktualisierung von Softwarekomponenten. Es hat sich als unerlässlich erwiesen, regelmäßige Systembackups durchzuführen und Updates in einer isolierten Umgebung zu testen, bevor sie im Hauptsystem implementiert werden.

Bei der Implementierung einer Übersetzungsfunktion evaluierten wir die Möglichkeit einer internen Lösung. Es wurde jedoch rasch deutlich, dass die Komplexität einer solchen Implementierung den Aufwand für die beschränkte Anforderung in unserer App nicht rechtfertigt. Daher entschieden wir uns, pragmatisch die API von Deepl im kostenlosen Bezugsplan zu nutzen, wobei wir die zusätzlichen Kosten eines Upgrades im Hinterkopf behielten, falls der Übersetzungsbedarf die Grenze von 500.000 Zeichen pro Monat überschreiten sollte.

Beim Thema der PDF-Verarbeitung kristallisierte sich Optical Character Recognition (OCR) als potenzieller Lösungsansatz heraus, der bei einer weiteren Fortführung des Projekts in Erwägung gezogen werden sollte. Ein zusätzliches technisches Hindernis war die Verarbeitung von .doc-Dateien, einem älteren Format aus dem Jahr 2003. Da aktuelle Standardbibliotheken dieses Format nicht mehr unterstützen, mussten wir auf alternative Lösungsstrategien zurückgreifen. Diese Erfahrungen betonen die Wichtigkeit, stets aktuelle technologische Entwicklungen und Standards im Blick zu haben und flexibel auf die Anforderungen und Erwartungen der Endbenutzer reagieren zu können. Es wurde erneut klar, wie bedeutend sorgfältige Voruntersuchungen und adaptive Lösungsansätze für den Erfolg eines Projekts sind.

Literaturverzeichnis

- Aichele, C., Schönberger, M., Aichele, C., & Schönberger, M. (2014). Grundlagen des Projektmanagements. *IT-Projektmanagement: Effiziente Einführung in das Management von Projekten*, 3–16.
- Peter-Johann, H. (2023). *Lastenheft und Pflichtenheft - Peterjohann Consulting*. Verfügbar 20. Mai 2023 unter https://www.peterjohann-consulting.de/lastenheft-und-pflichtenheft/#211_einige_typische_fehler_und_fallstricke