Projektauftrag (gemäß: DIN 69901-5:2009-01)

Projektinformation		
Projektname	TLDR AI	
Beschreibung	Entwicklung von Machine Learning (ML)-Modellen zur Textzusammenfassung und Klassifikation inklusive der Bereitstellung einer entsprechenden Schnittstelle	
Projekt Manager	Marco Zeulner (Domänenexperte)	
Ressourcen (Team)	Maximilian Graf (Software Engineer), Valentin Härdrich (Data Engineer), Ilyas Böhm (NLP-Experte)	
Dauer (von/bis)	08.05.2023	27.07.2023
Auftraggeber	DHBW Mannheim	
Budget	150 Std. * 4 = 600 Std. → 600 Std. * 96 € = 57.600 €	
Randbedingungen	Keine vorgefertigten Komplettlösungen an Software-Tools	

Projektbegründung

Angesichts der heutigen Informationsflut steht unsere Gesellschaft vor einer immer größer werdenden Herausforderung, relevante Informationen aus einer Vielzahl von Quellen zu filtern und zu verstehen. Insbesondere für Bildungseinrichtungen zeigt sich, dass diese von einer verbesserten Informationsverarbeitung und -bereitstellung profitieren können, um den Lern- und Forschungsprozess effektiver und effizienter zu gestalten.

Vor diesem Hintergrund soll im Rahmen dieses Projekts mit TLDR AI eine nutzbare Schnittstelle entwickelt werden, mit der Texte zusammengefasst und in Oberkategorien klassifiziert werden können. Diese soll es den Nutzern ermöglichen, den Inhalt von längeren Texten schnell zu erfassen und die wichtigsten Informationen in komprimierter Form zu erhalten. Des Weiteren kann der Auftraggeber DHBW Mannheim diese Schnittstelle anschließend nutzen, um die Suchfunktion der Online-Bibliothek zu verbessern, in dem Nutzer nach entsprechenden Arten von Literatur filtern können.

Projektziel

Das Ziel dieses Projekts ist die Entwicklung einer nutzbaren Schnittstelle (REST-API), mit welcher monolinguale Texte (englisch) automatisch zusammengefasst und in Oberkategorien klassifiziert werden können. Der zu übergebende Text kann eine beliebige Länge aufweisen, während die Länge des zusammengefassten Textes über eine angegebene Kompressionsrate bestimmt wird. Des Weiteren soll zwischen den folgenden Oberkategorien unterschieden werden: Nachrichtenartikel, literarische Texte, Blogbeitrag, wissenschaftliche

Arbeiten, Patente und Gesetzestexte.
Hierfür soll im Rahmen dieses Auftrags zwei ausgewählte ML-Modell für die Schnittstelle trainiert werden. Die Leistung der Modelle soll einen F1-Score von mindestens 65 % sowohl für die Textklassifikation als auch -zusammenfassung aufweisen.

Projektumfang

In-Scope

- Datenauswahl und Vorverarbeitung: Im Vorlauf zu den Entwicklungen und dem Training der Modelle soll eine passende Datengrundlage ausgewählt werden, mit welcher eine Bewältigung der definierten Aufgaben möglich ist. Des Weiteren sollen die ausgewählten Datensätze anschließend in einem zentralen Datensatz zusammengeführt werden.
- Auswahl und Anwendung geeigneter Lernalgorithmen: Basierend auf den Anforderungen des Auftrags und der ausgewählten Datengrundlage sollen geeignete Lernalgorithmen identifiziert werden. Die Wahl der Algorithmen soll auf deren Fähigkeit zur Bewältigung der definierten Aufgaben und der Effizienz beruhen. Der Lernalgorithmus wird an die Domäne angepasst und für das Training eines spezifischen Modells verwendet.
- Entwicklung einer Schnittstelle: Die Schnittstelle soll in der Lage sein, den übergebenen Text zu verarbeiten und die gewünschten Operationen (Klassifikation, Zusammenfassung) durchzuführen.
- Leistungsevaluation: Ausgehend von ausgewählten Metriken sollen die Modelle der Schnittstelle weitestgehend optimiert werden.
- Sicherung der Schnittstelle: Die Anfragen an die bereitgestellte REST-API soll entsprechend gesichert sein, sodass diese ausschließlich vom Auftraggeber angesprochen werden kann.

Out-of-scope

- Datensammlung: Die Beschaffung von Textdaten über ein (Web-)Scraping, auf denen die Modelle trainiert werden sollen, liegt außerhalb des Scopes dieses Projekts. Es werden öffentlich zugängliche, annotierte Daten verwendet.
- Lernalgorithmus: Im Rahmen dieses Projektes wird auf bestehende Lernalgorithmen zurückgegriffen. Die Entwicklung eigener, neuer Algorithmen ist nicht vorgesehen.
- Integration in weitere Systeme: Die Integration und Nutzung der Schnittstelle obliegt beim Auftraggeber und ist nicht Bestandteil des Auftrages. Die Schnittstelle wird als eigenständiges Modul entwickelt und bereitgestellt.
- Bereitstellung: Während eine grundlegende Leistung der Schnittstelle im Testbetrieb erwartet wird, liegt eine skalierbare Überführung in den Produktivbetrieb außerhalb des Projektrahmens.
- UI/UX-Design: Die Bereitstellung der Modelle erfolgt über eine Schnittstelle. Die Entwicklung einer Benutzeroberfläche ist nicht vorgesehen.

LASTENHEFT

Version: 0.1

Datum: 23.05.2023

DOKUMENTVERSIONEN

Versionsnr.	Datum	Autor	Änderungsgrund / Bemerkungen
0.1	23.05.2023	DHBW Mannheim	Ersterstellung

INHALT

1. EINLEITUNG

1.1 Allgemeines

1.1.1 Ziel und Zweck dieses Dokuments

Dieses Lastenheft enthält die Anforderungen für das Projekt TLDR AI, das von der DHBW Mannheim in Auftrag gegeben wurde. Ziel ist die Entwicklung einer nutzbaren Schnittstelle (REST-API), mit der englischsprachige Texte automatisch zusammengefasst und in Oberkategorien klassifiziert werden können.

1.1.2 Projektbezug

Das Projekt *TLDR AI* bezieht sich auf die Entwicklung einer KI-basierten Schnittstelle zur Textzusammenfassung und -klassifikation.

1.1.3 Abkürzungen

AI: Artificial Intelligence (Künstliche Intelligenz)
ML: Machine Learning (Maschinelles Lernen)
NLP: Natural Language Processing (Natürliche Sprachverarbeitung)
REST: Representational State Transfer
API: Application Programming Interface

1.1.4 Ablage, Gültigkeit und Bezüge zu anderen Dokumenten

Das Lastenheft ist während der gesamten Projektlaufzeit gültig und wird zentral beim Projektteam aufbewahrt. Es bezieht sich auf das Projektkonzept und den Projektplan.

1.2 Verteiler und Freigabe

1.2.1 Verteiler für dieses Lastenheft

Rolle	Name
Projektleiter	Marco Zeulner
NLP-Experte	Ilyas Böhm
Software Engineer	Maximilian Graf
Data Engineer	Valentin Härdrich

1.3 Reviewvermerke und Meeting-Protokolle

1.3.1 Erstes bis n-tes Review

Die Reviews des Lastenheftes werden im Verlauf des Projekts regelmäßig durchgeführt. Die wichtigsten Punkte und Entscheidungen aus diesen Reviews werden hier dokumentiert.

2. KONZEPT UND RAHMENBEDINGUNGEN

2.1 Benutzer / Zielgruppe

Die Zielgruppe aus Anwendungssicht sind Studierende und Mitarbeiter von Universitäten, die einen effizienten Zugang zu Informationen aus umfangreichen Texten benötigen. Jedoch soll nicht für jeden einzelnen Endanwender ein Konto zur Benutzung der zu entwickelten Schnittstelle erstellt und genutzt werden. Stattdessen soll ein Konto je Universität genutzt werden. Da diese Schnittstelle explizit von der DHBW Mannheim beauftragt wurde und die Implementierung im Anschluss an Letztere übergeben wird, wird nur ein einziges Konto benötigt.

Die Schnittstelle selbst soll mittels einer REST-API abgebildet werden, wodurch entsprechend fachliches Wissen zur Bedienung vorausgesetzt wird. Zur Integration in bestehende Lösungen werden deshalb IT-Administratoren und Webentwickler benötigt, welche die Zielgruppe aus Implementierungssicht widerspiegeln.

2.2 Ziele des Anbieters

Wird vom Auftragnehmer definiert (Pflichtenheft).

2.3 Ziele und Nutzen des Anwenders

Das Ziel der Anwender ist es, einen schnellen Überblick über lange Texte zu erhalten und diese nach vordefinierten Kategorien zu klassifizieren. Diese soll es den Nutzern ermöglichen, den Inhalt von längeren Texten schnell zu erfassen und die wichtigsten Informationen in komprimierter Form zu erhalten. Mit der Klassifikation soll es den Nutzern ermöglicht werden, entsprechende Arten von Literatur zu filtern.
Für den Auftraggeber DHBW Mannheim kann diese Schnittstelle anschließend genutzt werden, um die Suchfunktion der Online-Bibliothek durch eine Erweiterung zu verbessern.

2.4 Systemvoraussetzungen

Wird vom Auftragnehmer definiert (Pflichtenheft).

2.5 Ressourcen

Wird vom Auftragnehmer definiert (Pflichtenheft).

2.6 Funktionale Anforderungen

UI !

Als Anwender möchte ich die Schnittstelle mithilfe des REST-Protokolls ansprechen können, damit ich eine universelle und standardisierte Methode der Kommunikation habe.	
ID	2.6.1 L) wenn so komplex

Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	1

Als Anwender möchte ich den zu klassifizierenden oder zusammenzufassenden Text im "Body" der HTTP-Anfrage übergeben können, um eine bequeme und standardisierte Methode der Dateneingabe zu haben.	
ID	2.6.2
Vergleich zu bestehenden Lösungen	Ähnlich zu den meisten Web-Diensten, die Daten in den Body von HTTP-Requests übergeben.
Priorität	1

Als Anwender möchte ich die Antwort meiner Anfrage in standardisierten Schlüssel-Wert-Paaren erhalten, um eine leicht verständliche und weiterverarbeitbare Datenstruktur zu haben.	
ID	2.6.3
Vergleich zu bestehenden Lösungen	Ähnlich den öffentlichen REST-API's von Twitter oder Youtube.
Priorität	1

Als Anwender möchte ich, dass die Schnittstelle in der Lage ist, englischen Text zusammenzufassen und zu klassifizieren, um eine Vielzahl von Texten verarbeiten zu können.	
ID	2.6.4
Vergleich zu bestehenden Lösungen	Ähnlich zu GPT-4 (OpenAI) und PaLM-2 (Google)
Priorität	1

Als Anwender möchte ich für nicht vorgesehene Sprachen eine aussagekräftige Fehlermeldung erhalten, um über unterstützte Sprachen informiert zu werden.	
ID	2.6.5

Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	2

Als Anwender möchte ich, dass die Schnittstelle in der Lage ist, beliebig lange Texte zusammenzufassen, um nicht nur auf kurze Texte beschränkt zu sein.	
ID	2.6.6 beliebig lang wird schwierig, Bücher?
Vergleich zu bestehenden Lösungen	Ähnlich zu GPT-4 (OpenAI) und PaLM-2 (Google).
Priorität	2

Als Anwender möchte ich für fehlgeschlagene Zusammenfassungen oder Klassifikationen eine aussagekräftige Fehlermeldung erhalten, um meine Anfrage entsprechend anzupassen oder zu wiederholen.	
ID	2.6.7
Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	1

Als Anwender möchte ich eine aussagekräftige Fehlermeldung erhalten, wenn mein Authentifizierungs-Token abgelaufen ist, um einen neuen anfordern zu können.	
ID	2.6.8 Lowle Authentifizierung?
Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung
Priorität	1

Als Anwender möchte ich bei der Klassifikation eine Debug-Flag setzen können und damit nicht nur die "wahrscheinlichste" Klasse als Antwort erhalten, sondern alle Klassen inklusive der dazugehörigen Modellergebnis-Werte, um eine detaillierte Analyse durchführen zu können.	
ID	2.6.9

Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	2

Als Anwender möchte ich, dass der Text innerhalb von maximal 10 Sekunden zusammengefasst wird, um eine effiziente Nutzung des Dienstes zu gewährleisten.	
ID	2.6.10
Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	2

Als Anwender möchte ich die Länge der Zusammenfassung über eine vom Anwender wählbare Kompressionsrate konfigurieren können, um eine personalisierte Zusammenfassung zu erhalten.	
ID	2.6.11
Vergleich zu bestehenden Lösungen	Ähnlich zu Diensten wie SMMRY, das eine konfigurierbare Textkompression bietet.
Priorität	1

Als Anwender möchte ich, dass der Text ausschließlich in die Kategorien “Nachrichtenartikel”, “literarischer Text”, “Blogbeitrag”, “Paper”, “Patent” und “Gesetz” klassifiziert wird, um eine konsistente und nützliche Klassifizierung zu gewährleisten.	
ID	2.6.12
Vergleich zu bestehenden Lösungen	Ähnlich zu Systemen wie IBM Watson NLU, das Text in vordefinierte Kategorien klassifiziert.
Priorität	1

Als Anwender möchte ich, dass die Leistung der Modelle einen F1-Score von mindestens 65 % sowohl für die Textklassifikation als auch Zusammenfassung aufweisen, um zuverlässige Ergebnisse zu erhalten.	
ID	2.6.13

Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	1

Als Anwender möchte ich, dass die Zusammenfassung des Textes dem Inhalt des ursprünglichen Textes entspricht und keine neuen Informationen erfunden werden, um eine genaue und vertrauenswürdige Zusammenfassung zu erhalten.	
ID	2.6.14
Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung. → GPT?
Priorität	1

Als Anwender möchte ich, dass die Schnittstelle die Klasse “Sonstige” zurückgibt, wenn sich das Model nicht sicher ist, um nur eindeutige Klassifizierungen zu erlauben.	
ID	2.6.15 → s.o.
Vergleich zu bestehenden Lösungen	Keine vergleichbare Lösung.
Priorität	1

2.7 Nicht-funktionale Anforderungen

→ hier ohne Struktur?

ID	Beschreibung	Kategorie (ISO/IEC 25010:2011)	Priorität
2.7.1	Eine vollständige technische Dokumentation der API, einschließlich Implementierungsdetails und Statuscodes, muss erstellt und bereitgestellt werden. (ähnlich zu „Swagger“)	Benutzbarkeit	2
2.7.2	Das System muss in der Programmiersprache Python entwickelt werden. → Grund?	Funktionalität	1
2.7.3	Die API-Schnittstelle muss den REST-Designprinzipien folgen	Kompatibilität	1

2.7.4	Die Lösung muss plattformunabhängig sein und sowohl auf Windows- als auch auf Linux-Systemen laufen können.	Portabilität	1
2.7.5	Die REST-API muss sowohl Verschlüsselung für alle eingehenden Anfragen unterstützen als auch Mechanismen zur Authentifizierung von Benutzern bereitstellen.	Sicherheit	1
2.7.6	Das System muss sicherstellen, dass übergebene Textdaten nicht aufgezeichnet oder in irgendeiner Form dauerhaft gespeichert werden.	Sicherheit	1
2.7.7	Das System muss die Datenschutz-Grundverordnung (DSGVO) einhalten und alle relevanten Anforderungen erfüllen.	Sicherheit	1
2.7.8	Alle Passwörter, die im System verwendet werden, müssen sicher verschlüsselt werden.	Sicherheit	1
2.7.9	Alle Codeelemente (Datenvorverarbeitung, Modell-Trainings, Schnittstelle) müssen angemessen dokumentiert sein, um die Verständlichkeit und Wartbarkeit des Systems zu gewährleisten.	Wartbarkeit	1
2.7.10	Die API-Schnittstelle muss eine hohe Zuverlässigkeit aufweisen und bei allen Anforderungen korrekt funktionieren.	Zuverlässigkeit	1
2.7.11	Die API-Schnittstelle muss eine redundante Implementierung ermöglichen, um Ausfallzeiten zu minimieren und eine hohe Verfügbarkeit zu gewährleisten.	Zuverlässigkeit	2

3. GENEHMIGUNG

Die Genehmigung erfolgt...

Datum:	24.05.2023
Unterschrift Auftraggeber:	DHBW Mannheim
Unterschrift Projektleiter:	Marco Zeulner
Weitere Unterschriften:	-

PFLICHTENHEFT

Version: 0.1

Datum: 23.05.2023

DOKUMENTVERSIONEN

Versionsnr.	Datum	Autor	Änderungsgrund / Bemerkungen
0.1	23.05.2023	Projektteam TLDR AI	Ersterstellung

INHALT

1. Einleitung

1.1 Allgemeines

1.1.1 Ziel und Zweck dieses Dokuments

Dieses Pflichtenheft enthält die Spezifikationen für das Projekt TLDR AI, das von der DHBW Mannheim in Auftrag gegeben wurde. Ziel ist die Entwicklung einer nutzbaren Schnittstelle (REST-API), mit der englischsprachige Texte automatisch zusammengefasst und in Oberkategorien klassifiziert werden können.

1.1.2 Projektbezug

Das Projekt *TLDR AI* bezieht sich auf die Entwicklung einer KI-basierten Schnittstelle zur Textzusammenfassung und -klassifikation.

1.1.3 Abkürzungen

AI: Artificial Intelligence (Künstliche Intelligenz)
ML: Machine Learning (Maschinelles Lernen)
NLP: Natural Language Processing (Natürliche Sprachverarbeitung)
REST: Representational State Transfer
API: Application Programming Interface

1.1.4 Ablage, Gültigkeit und Bezüge zu anderen Dokumenten

Das Pflichtenheft ist während der gesamten Projektlaufzeit gültig und wird zentral beim Projektteam aufbewahrt. Es bezieht sich auf das Projektkonzept und den Projektplan.

1.2 Verteiler und Freigabe

1.2.1 Verteiler für dieses Pflichtenheft

Rolle	Name
Projektleiter	Marco Zeulner
NLP-Experte	Ilyas Böhm
Software Engineer	Maximilian Graf
Data Engineer	Valentin Härdrich

1.3 Reviewvermerke und Meeting-Protokolle

1.3.1 Erstes bis n-tes Review

Die Reviews des Pflichtenheftes werden im Verlauf des Projekts regelmäßig durchgeführt. Die wichtigsten Punkte und Entscheidungen aus diesen Reviews werden hier dokumentiert.

2. Konzept und Rahmenbedingungen

2.1 Benutzer / Zielgruppe

-/-

2.2 Ziele des Anbieters

Das Hauptziel ist die Entwicklung einer robusten, effizienten und benutzerfreundlichen KI-Schnittstelle für die Textzusammenfassung und -klassifikation. Daraus ergeben sich zwei zusätzliche Nebenziele:
Die Schnittstelle soll jährlich im Durchschnitt zu 99% verfügbar sein. Für Wartungsarbeiten und Ausfälle sollen Redundanzen und Lastverteilungsmechanismen genutzt werden, um die tatsächlichen Ausfallzeiten minimal zu halten. Damit diese Anforderung nach Projektende im Rahmen des Betriebs seitens der DHBW Mannheim erfüllt werden kann, soll die bereitgestellte Schnittstelle und die Modelle im Stil der Microservice-Architektur implementiert und vorbereitet werden, wodurch eine horizontale Skalierung ermöglicht wird.
Die Schnittstelle soll nur von autorisierten Nutzern genutzt werden können. Diese müssen hierfür initial ein Benutzerkonto anlegen lassen. Für die Benutzung müssen diese sich mit ihren Anmeldedaten authentifizieren, bevor sie einen temporär gültigen Token für nachfolgende Anfragen zugestellt bekommen. Rate Limiting via User?

2.3 Ziele und Nutzen des Anwenders

-/-

2.4 Systemvoraussetzungen

Um das System betreiben zu können, wird eine Python-Umgebung benötigt. Diese kann entweder über einen Server oder einen Container bereitgestellt werden. Die Schnittstelle soll als REST-API bereitgestellt werden, um eine einfache Integration in bestehende Systeme zu ermöglichen. Zur Nutzung der Schnittstelle wird eine Internetverbindung benötigt und ein Tool/SDK/Paket, welches die Kommunikation mit einer REST-API ermöglicht. Die angesprochene Schnittstelle stellt Rückmeldungen im JSON-Format bereit. Diese Nachrichten müssen vom Empfänger verarbeitet werden können.

2.5 Ressourcen

Das Projektteam besteht aus vier Personen mit Fachkenntnissen in den Bereichen Software-Engineering, Datenverarbeitung und NLP. Der Aufwand wurde 150 Stunden pro Person geschätzt, wodurch sich insgesamt 600 Arbeitsstunden ergeben. Mit einem Stundenlohn von 96 € pro Person ergibt sich ein Gesamtbudget von 57600 €. Die benötigte Hard- und Software ist im Budget inkludiert. Hierzu gehören ein Rechner/Server zum Trainieren der ML-Modelle, Python-Bibliotheke zur natürlichen Sprachverarbeitung und annotierte Datensätze.
↳ Libraries

3. Anforderungsbeschreibung

Im Folgenden werden die vom Lastenheft abgeleiteten Anforderungen als Pflichten aufgeführt und beschrieben. Zur besseren Übersicht und Organisation wurden die Pflichten in Komponenten zusammengefasst.

3.1 Implementierung einer Schnittstelle nach dem REST-Protokoll

Implementierung einer API nach dem REST-Standard	
ID	1
Beschreibung	Die zu implementierende Schnittstelle soll den Prinzipien einer REST-Architektur entsprechen und HTTP-Anfragen empfangen und beantworten können. Der vom Anwender im Body des HTTP-Requests übergebene Text soll entsprechend verarbeitet werden können, um eine Klassifikation und Zusammenfassung des übergebenen Textes zu ermöglichen. Die Antwort auf eine Anfrage soll in Form von standardisierten Schlüssel-Werte-Paaren an den Anwender zurückgesendet werden. Die Rückgabe enthält entweder den zusammengefassten Text oder die wahrscheinlichste Klasse. Für die wahrscheinlichste Klasse wird ein Grenzwert festgelegt, falls dieser unterschritten wird, soll erkenntlich werden, dass die Klasse nicht bestimmt werden konnte. Bei Nutzung eines speziellen Debug-Flags sollen alle Klassen mit zugehörigen Wahrscheinlichkeiten zurückgegeben werden. Die Authentifizierung für die Nutzung der Schnittstelle sollte verschlüsselt sein. Eine Persistierung oder Protokollierung des übergegebenen Textes ist nicht vorgesehen. Für den Fall der Textzusammenfassung soll zusätzlich zum Text ein Kompressionsparameter übergeben werden. Bei Fehlern der Modelle oder beim Login wird eine aussagekräftige Fehlermeldung zurückgesendet. Gleiches gilt für die Eingabe eines Textes mit nicht unterstützter Sprache. → Fehlercode?
Technische Umsetzung	Die REST-API wird mittels des Frameworks fastAPI umgesetzt. Diese soll die drei folgenden Routen bereitstellen: - /classification - /summarization - /login Die Umsetzung der Sicherheitsbestimmung erfolgt über JSON Web Tokens (JWT). Für die Zusammenfassung wird der Kompressionsparameter als *Query-Parameter* übergeben.
Wechselwirkungen	Erforderlich für die Bereitstellung der Funktionalitäten sind die beiden Modelle für die Klassifizierung und Zusammenfassung des Textes.
Risiken	Der Auftraggeber muss die Online-Bibliothek für die Integration und Nutzung der REST-API anpassen.
Betroffene funktionale Anforderungen aus Lastenheft	2.6.1; 2.6.2; 2.6.5; 2.6.7; 2.6.10; 2.6.11

Betroffene nicht-funktionale Anforderungen aus Lastenheft	2.7.3; 2.7.5; 2.7.6; 2.7.7; 2.7.8; 2.7.10; 2.7.11
Schätzung des Aufwands	100 Stunden

3.2 Datengrundlage

Sammlung und Vorverarbeitung der Daten für das Modelltraining	
ID	2
Beschreibung	Die Datengrundlage für das Training der Modelle zur Klassifikation und Zusammenfassung wird auf Grundlage öffentlich zugänglicher Daten erstellt. Diese enthalten bereits die korrespondierenden Klassen und/oder den zusammengefassten Text. Bei der Wahl der Datensätze sind folgende Kriterien ausschlaggebend: - valide und bekannte Quelle - ausreichende Qualität, insbesondere in der Annotation - ausreichende Größe
Technische Umsetzung	Die Daten werden für die Funktionalität der Klassifikation von Texten aus mehreren Datensätzen zusammengestellt und in einen einzelnen Datensatz überführt. Eine Tabelle mit den dafür verwendeten Datensätzen findet sich im Anhang dieses Dokuments.
Wechselwirkungen	Voraussetzung für das Training der Modelle und bestimmender Faktor für die Qualität des Endprodukts.
Risiken	Verfügbarkeit der Datensätze mit o.g. Anforderungen
Betroffene funktionale Anforderungen aus Lastenheft	-/-
Betroffene nicht-funktionale Anforderungen aus Lastenheft	-/-
Schätzung des Aufwands	40 Stunden

3.3 Erstellung eines Modells zur Zusammenfassung von Texten

Modell zur Zusammenfassung von Texten	
ID	3
Beschreibung	Mit der Schnittstelle soll eine Komponente bereitgestellt werden, welche mittels eines ML-Modells Texte in englischer Sprache zusammenfassen kann. Die Ausgabe des Modells soll von einer Texteingabe und einer Kompressionsrate abhängen. Das Modell soll auf einer Transformer-Architektur basieren. Es soll erkannt werden, ob die Eingabe in der erwarteten Sprache (Englisch) ist. Texte, deren Länge die maximale Eingabegröße überschreitet, werden in einzelne Textbestandteile zerlegt und anschließend unabhängig voneinander zusammengefasst. Bei Auftreten eines Fehlers wird anstelle des Ergebnisses eine aussagekräftige Fehlermeldung zurückgegeben. Die Evaluation auf Testdaten soll einen F1-Score von > 65 % erreichen. Des Weiteren soll der Inhalt des zusammengefassten Textes dem ursprünglichen Text entsprechen und keine erfundenen Informationen enthalten. → wie wird das gemessen?
Technische Umsetzung	Transformer-Modell mit angepassten Sequenzen zur Zusammenfassung (bspw. Unifying Cross-lingual Summarization and Machine Translation with Compression Rate (arxiv.org)).
Wechselwirkungen	Wichtiger Faktor für die Qualität des Endprodukts; Muss die vorgenommene Zusammenfassung an die REST-API zurückgeben
Risiken	Das Modell funktioniert nicht wie erwartet. Kompressionsrate kann nicht ausreichend akkurat umgesetzt werden.
Betroffene funktionale Anforderungen aus Lastenheft	2.6.4; 2.6.5; 2.6.7; 2.6.7; 2.6.10; 2.6.11; 2.6.13; 2.6.14
Betroffene nicht-funktionale Anforderungen aus Lastenheft	2.7.2; 2.7.6; 2.7.9
Schätzung des Aufwands	140 Stunden → realistisch!

3.4 Erstellung eines Modells zur Klassifizierung von Texten

Modell zur Klassifizierung von Texten	
ID	4
Beschreibung	Mit der Schnittstelle soll eine Komponente bereitgestellt werden, mit welcher mittels eines ML-Modells Texte in englischer Sprache einer von sechs zuvor definierten Klassen zugeordnet werden. Die möglichen Klassen sind Nachrichtenartikel, literarischer Text, Blogbeitrag, Artikel aus wiss. Zeitschrift (Paper), Patente und Gesetze. Neben den genannten Klassen soll kein anderes Ergebnis zurückgegeben werden.
Technische Umsetzung	Für die Klassifikation von Textdaten wird ein vorab trainiertes bidirektionales Sprachmodell verwendet, welches auf dem Encoder-Teil der Transformer-Architektur basiert. Bei dem vorgesehenen Modell (RoBERTa) handelt es sich um ein monolinguales Modell, welches auf den für das Projekt konsolidierten Daten fein abgestimmt (Fine-Tuning) wird. Die technische Umsetzung erfolgt unter der Verwendung der Python-Bibliotheken Pytorch und Hugging Face.
Wechselwirkungen	Wichtiger Faktor für die Qualität des Endprodukts; Muss die vorgenommene Klassifikation an die REST-API zurückgeben
Risiken	Falsche Klassifizierung, wenn Textart nicht in Kategorien.
Betroffene funktionale Anforderungen aus Lastenheft	2.6.4; 2.6.7; 2.6.9; 2.6.12; 2.6.13
Betroffene nicht-funktionale Anforderungen aus Lastenheft	2.7.2; 2.7.9
Schätzung des Aufwands	120 Stunden

also keine Berücksichtigung sonstiger Eingaben?

3.5 Benutzermanagement

Erstellung eines Benutzer-Managements mit verschlüsselten Login	
ID	5
Beschreibung	Für die Anwendung an der DHBW Mannheim wird das System mit vorerst einem Nutzer aufgesetzt. Dieser kann sich per Passwort authentifizieren. Die Autorisierungen bei konsekutiven API-Aufrufen soll über einen Token erfolgen, welcher bei der initialen Authentifizierung ausgestellt wird und eine zeitlich begrenzte Gültigkeit besitzt. Des Weiteren soll aus Sicherheitsgründen das Passwort gehashed abgespeichert werden.
Technische Umsetzung	Der Benutzer inklusive gehashtem Passwort wird in einer SQLite Datenbank gespeichert. Für das Passwort-Hashing wird die Python-Bibliothek *passlib* und der Algorithmus *Bcrypt* verwendet, welcher Passwörter mithilfe eines zusätzlichen Salts hashed. Zur Autorisierung des Benutzers wird ein vom Benutzer initial frei wählbares Passwort verwendet. Dieser muss zusammen mit dem Benutzername der Route “/login” übergeben werden, welche bei erfolgreicher Überprüfung einen 3-Tage lang gültigen Token zurückgibt. Dieser Token wird als JSON Web Token (JWT) abgebildet und muss allen anderen Routen zur Autorisierung übergeben werden.
Wechselwirkungen	Funktionierendes Benutzermanagement ist Voraussetzung für eine erfolgreiche Anmeldung an API.
Risiken	Eine unsichere Implementation könnte zu kompromittierten Passwörtern führen.
Betroffene funktionale Anforderungen aus Lastenheft	2.6.9
Betroffene nicht-funktionale Anforderungen aus Lastenheft	-/-
Schätzung des Aufwands	40 Stunden

Vermutlich komplette Add-Library sinnvoll?

3.6 Dokumentation

Dokumentation der Schnittstelle und des Quellcodes	
ID	6
Beschreibung	Erstellung einer technischen Dokumentation für IT-erfahrene Nutzer der Anwendung. Ausführliche Codedokumentation, aufgrund derer sich weitere Entwickler ohne Probleme in die technische Funktionsweise der Schnittstelle einarbeiten können.
Technische Umsetzung	Erstellung eines Dokuments, welches die technische Dokumentation enthält. Die Codedokumentation erfolgt mittels Line/Block-Comments direkt im Code.
Wechselwirkungen	Wechselwirkungen mit allen Komponenten, da alle Teile der Schnittstelle dokumentiert werden sollen.
Risiken	Bei unsachgemäßer Umsetzung können Sicherheitsrisiken und Einschränkungen in der Skalierbarkeit auftreten. Ein Risiko besteht durch mangelhafte Ausführung dieser Anforderung. Dann könnten bspw. keine Problemlösung bei Ausfall des Systems durchgeführt werden.
Betroffene funktionale Anforderungen aus Lastenheft	-/-
Betroffene nicht-funktionale Anforderungen aus Lastenheft	2.7.1; 2.7.9
Schätzung des Aufwands	120 Stunden

Code-Qualität? SCP?

3.7 Architektur

Spezifikation der Architektur	
ID	7
Beschreibung	Die zu verwendende Programmiersprache ist Python. Die Architektur wird in drei unterschiedliche Komponenten aufgeteilt. Zum einen die beiden Modelle zur Klassifikation und Zusammenfassung der Textdaten. Zum anderen das Backend der REST-API. Geliefert wird keine vollständige Architektur, sondern Container, die vom Auftraggeber selbst gehostet werden.
Technische Umsetzung	Die gewünschte Zielarchitektur im Betrieb der Softwarelösung kann dem Architekturplan im Anhang unter Punkt 4.2 entnommen werden.
Wechselwirkungen	Die Festlegung der Programmiersprache auf Python wirkt sich auf die Implementierung der Modelle und der Schnittstelle aus.
Risiken	-/-
Betroffene funktionale Anforderungen aus Lastenheft	-/-
Betroffene nicht-funktionale Anforderungen aus Lastenheft	-/-
Schätzung des Aufwands	40 Stunden

GENEHMIGUNG

Datum:	24.05.2023
Unterschrift Auftraggeber:	DHBW Mannheim
Unterschrift Projektleiter:	Marco Zeulner
Weitere Unterschriften:	-

4. Anhang

4.1 Tabellarische Übersicht der Datenquellen

4.1.1 Daten für Klassifikation

Für das Training des Modells zur Klassifikation von Textdaten ist eine Kombination der folgenden Datensätze vorgesehen:

Link	Anzahl der Daten
Scientific Papers - scientific_papers · Datasets at Hugging Face	350.000 Paper
Blog - blog_authorship_corpus · Datasets at Hugging Face	681.000 Blog-posts
News - https://lil.nlp.cornell.edu/newsroom/index.html	1.300.000 Artikel
Patent - https://huggingface.co/datasets/ccdv/patent-classification - https://huggingface.co/datasets/big_patent	35.000 Patente + 1.300.000 Patentdokumente
Literary Texts - https://github.com/dbamman/litbank - https://txtlab.org/data-sets/	100 Werke + 150 Novels + 1900 Fiction + 600.000 Fiction (+ Non-Fiction) + 75.000 Gedichte
Gesetze - https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/0EGYWY	142.000 EU-Gesetze

Um eine ausgewogene Klassenverteilung sicherzustellen, werden die Datensätze in einen einzelnen Datensatz überführt, wobei 10 000 Trainingsdaten je Klasse vorgesehen sind.

4.1.2 Daten für Zusammenfassung

Für das Training des Modells zur Zusammenfassung von Texten ist die Nutzung einer Kombination der folgenden Datensätze vorgesehen:

- News: Cornell Newsroom Dataset - NLP Hub - Metatext
- Wikipedia: WikiSummary Dataset - NLP Hub - Metatext
- Reddit: reddit · Datasets at Hugging Face

4.2 Architektur