Retrieval-Augmented Generation für den Mittelstand
KI-Innovationswettbewerb – Generative KI für den Mittelstand

Projekt ID: 01MK250104 **Projektstart:** 01.02.2025 **Laufzeit:** 36 Monate

Ergebnis 4.1: Implementierung der Basisversion

Publikationslevel	Öffentlich / Projektintern
Zieldatum	Monat 9, 31.10.2025
Abschlussdatum	Monat 9, 31.10.2025
Arbeitspaket	AP4 – Platform
Ergebnis	E4.1
Typ	Report
Status	Final
Version	1.0

Kurzzusammenfassung: Dieser Bericht gibt einen Überblick über die Basisversion (v0.1) der Learn2RAG Plattform.

Gefördert durch:

Bundesministerium
für Forschung, Technologie
und Raumfahrt

Dieses Projekt wurde finanziert vom Bundesministerium für Wirtschaft und Klimaschutz (01MK250104).

History

Version	Datum	Änderung	Author
0.0	29.08.2025	Struktur erstellt	Michael Röder
0.1	29.09.2025	Erster Entwurf abgeschlossen	Michael Röder
0.2	14.10.2025	Erster Entwurf gegengelesen	Tobias Seidenberg
0.3	15.10.2025	Feedback eingearbeitet	Michael Röder
1.0	3.10.2025	Finale Version abgeschlossen und veröffentlicht	Michael Röder

Zusammenfassung

Dieser Bericht gibt einen Überblick über die Basisversion (v0.1) der Learn2RAG Plattform. Dies beinhaltet eine kurze Beschreibung der einzelnen Komponenten, wie diese zusammengeführt werden um einen einzelnen, einfach auszuführenden Prototypen zu erstellen und wie dieser die zuvor erhobenen Anforderungen adressiert. Für eine detaillierte technische Dokumentation verweisen wir auf die Onlinedokumentation der Plattform.

List of Abbreviations

AP	Arbeitspaket
API	Application Programming Interface
RAG	Retrieval Augmented Generation

Table of Contents

Zusammenfassung **2**
1 Einleitung **5**
2 Systemübersicht **6**
Datenimport 7
RAG-Pipeline 7
Sprachmodell 7
Vector Store 7
Chat UI 8
Konfigurator 8
Build- und Deployment-Prozess 9
3 Anforderungen **9**

1 Einleitung

Das Ziel des Arbeitspakets 4 ist die Entwicklung, Wartung und Evaluation der Learn2RAG Plattform. Dies umfasst die Integration der in AP2 und 3 entwickelten Komponenten sowie alle weiteren für die Plattform notwendigen Funktionen. Als erster Schritt wurde in AP4 eine Basisversion (v0.1) der Learn2RAG Plattform entwickelt. Dabei bauen wir auf die durch AP1 erhobenen Anforderungen an diese Version auf.

Dieses Dokument beschreibt die entwickelte Basisversion der Learn2RAG Plattform. In den folgenden Abschnitten geben wir eine Übersicht über die Plattform und geben eine kurze Beschreibung ihrer einzelnen Komponenten. Für eine detailliertere Beschreibung verweisen wir auf die Onlinedokumentation der Plattform.[1] Abschließend werden die durch AP1 erhobenen Anforderungen mit den Eigenschaften der Plattform abgeglichen.

2 Systemübersicht

Abbildung 1 gibt einen Überblick über die Basisversion der Learn2RAG Plattform. Dabei werden ihre Komponenten sowie deren Zusammenarbeit dargestellt.

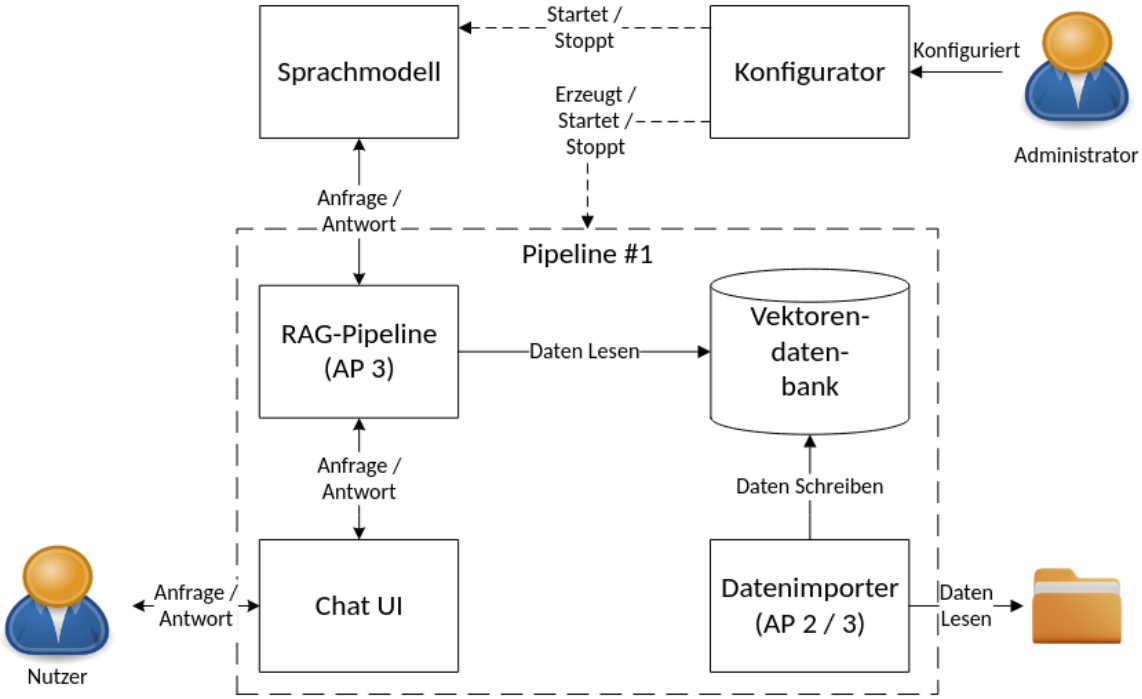


Abbildung 1 - Übersicht der Komponenten der Basisversion und Zuordnung zu den Arbeitspaketen

Eine der in AP4 entwickelten Komponenten ist der Konfigurator. Er ist das zentrale Werkzeug des Administrators zur Konfiguration der Plattform und orchestriert die anderen Komponenten. Er startet und stoppt das Sprachmodell und die einzelnen vom Administrator konfigurierten Pipelines. Eine einzelne Pipeline besteht aus 4 Komponenten. Der Datenimporter aus den APs 2 und 3 lädt die vom Administrator selektierten Daten ein und schreibt sie in die Vektordatenbank. Nach dieser Importphase kann der Benutzer seine Anfragen über einen Chat mit der in AP 3 entwickelten RAG-Pipeline interagieren. Diese reichert die Benutzeranfrage mit Daten aus der Vektordatenbank an, bevor sie die Anfrage an das Sprachmodell schickt. Die Antwort auf die Anfrage wird dann dem Nutzer in der Benutzeroberfläche angezeigt. Abbildung 2 zeigt ein Sequenzdiagramm der zuvor beschrieben Schritte inklusive des Startens und Stoppens der Pipeline durch den Konfigurator.

[1] https://docs.learn2rag.de/

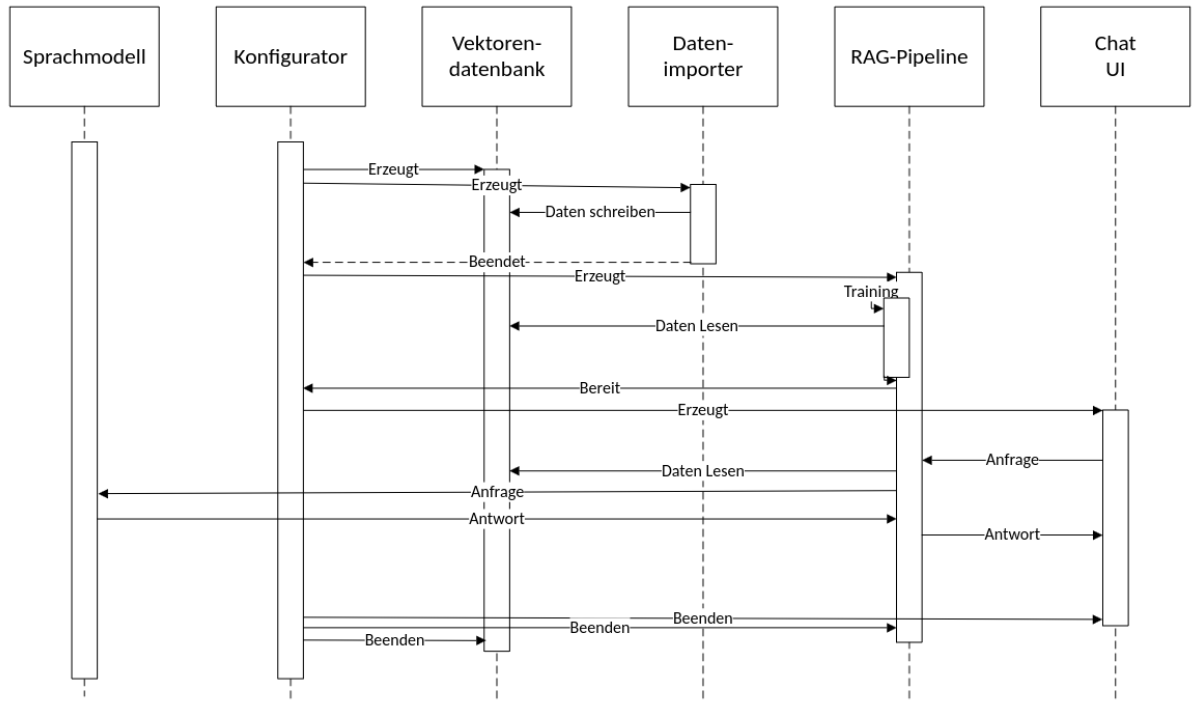


Abbildung 2 - Sequenzdiagramm der Basisversion

Datenimport

In der Basisversion unterstützt der Importprozess 2 generische Datenquellen. Zum einen können lokale Dateien bzw. ganze Ordnerstrukturen importiert werden. Zum anderen können HTML-Webseiten heruntergeladen werden. Der Import dieser Quellen wurde in AP2 entwickelt und in E2.1 näher beschrieben. In die Basisversion wurde eine Importpipeline integriert, die die Verarbeitung der Daten aus AP2 und die Vorverarbeitung der Daten aus AP3 integriert.

RAG-Pipeline

In der Basisversion ist die RAG-Pipeline als Webservice implementiert und stellt eine optimierte Version einer naiven RAG-Lösung dar. Sie besteht also aus den klassischen Schritten Retrievel, Augmentation und Generation. Eine detaillierte Beschreibung der Pipeline findet sich in E3.1.

Sprachmodell

Das Einbinden des Sprachmodells ist auf 2 Arten möglich. Es kann ein Sprachmodell heruntergeladen werden, dass dann lokal ausgeführt wird. Alternativ kann ein Sprachmodell eingebunden werden, dass auf einer anderen Maschine betrieben wird. Für die Testbereitstellung wird das Ollama Framework verwendet.[2]

Vector Store

Die Vektordatenbank speichert die indexierten Informationen, auf die zur Beantwortung

[2] Das Ollama Projekt (https://ollama.com/) ist open source (https://github.com/ollama/ollama).

einer Nutzeranfrage zurückgegriffen werden sollen. Während der Entwicklung der Basisversion wurden verschiedene Implementierungen in Erwägung gezogen, bevor wir uns für die Verwendung von Qdrant entschieden.[3] Diese Vektordatenbank bietet 3 Vorteile:

1. Die Möglichkeit sowohl eine vektorbasierte Suche als auch eine Schlagwortsuche ausführen zu können.
2. Sie ist open-source.
3. Sie lässt sich über eine Datei konfigurieren.
4. Sie kann sowohl auf Windows als auch Linux-distributionen ausgeführt werden.

Chat UI

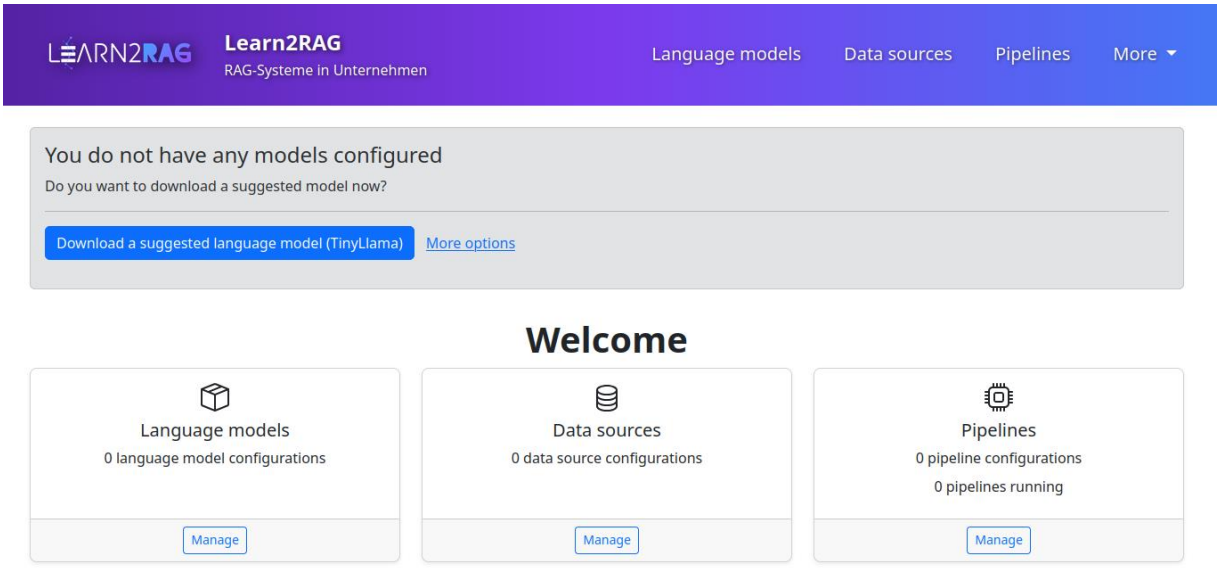
Als Benutzeroberfläche wurde Open Web UI verwendet.[4] Diese kann einfach vom Benutzer über den eigenen Browser geöffnet werden. Im Hintergrund werden Anfragen an den Webservice der RAG-Pipeline geschickt und deren Antwort angezeigt.

Konfigurator

Der Konfigurator ist eine der zentralen Komponenten, die von AP4 erstellt wurden. Er bietet dem Nutzer die Möglichkeit

- Sprachmodelle herunterzuladen bzw. einzubinden,
- Datenquellen zu definieren, und
- RAG-Pipelines zu definieren, trainieren und starten.

Der Konfigurator bietet auf einer Weboberfläche zunächst eine Übersicht über die aktuelle Konfiguration (Siehe Abbildung 3). Für alle zuvor beschriebenen Komponenten gibt es jeweils eine Unterseite, auf der sie angelegt und gelöscht werden können.



[3] Qdrant kann unter https://github.com/qdrant/qdrant gefunden werden.
[4] Das Open Web UI Projekt (https://openwebui.com/) ist open source (https://github.com/open-webui/open-webui).

Abbildung 3 - Weboberfläche des Konfigurators

Build- und Deployment-Prozess

Ein zentrales Ziel des Projekts ist einen möglichst einfach auszuführenden Prototyp zu entwickeln, der auf verschiedenen Systemen ausgeführt werden kann. Hierfür wurde in AP4 ein Build-Prozess entwickelt, der genau diese Problemstellung adressiert. Alle zuvor aufgeführten Komponenten des Prototyps werden im Build-Prozess in eine einzelne Datei verpackt, die von einem Nutzer einfach entpackt und ausgeführt werden kann. Qdrant und Ollama werden als ausführbare Binärcodedateien eingebunden. Die anderen Komponenten sind in Python geschrieben und werden während des Verpackungsprozesses gebaut. Das heißt, es werden für jede Komponenten zunächst die benötigten Bibliotheken und Umgebungen heruntergeladen. Danach werden für jede Python-Komponente die Pythonapplikation gebaut. Dann wird pyapp verwendet, um die einzelnen Komponenten mit ihren kompletten Umgebungen zu verpacken.[5] Danach werden die erstellten Dateien in eine einzelne Archivdatei verpackt und komprimiert.

Der Build-Prozess ist für eine Ausführung auf einem Debian-System konzipiert. Die erstellte Datei kann aber sowohl auf Linux als auch Windows Distributionen verwendet werden.

3 Anforderungen

Im Folgenden vergleichen wir die Anforderungen an die Basisversion aus E1.1, die mindestens die Priorität COULD hatten und wie diese in der Basisversion adressiert wurden. Von 21 Anforderungen setzt die Basisversion 14 um – darunter alle MUST-Anforderungen. Lediglich 7 Anforderungen stehen noch aus, wobei einige bereits teilweise adressiert wurden. Tabelle 1 enthält die einzelnen Anforderungen und deren Umsetzung.

Tabelle 1: Anforderungen an die Basisversion.

Geschäftliche und strategische Anforderungen			
Anforderung	**Priorität**	**Status**	**Adressierung**
Vendor-Lock-in vermeiden -offene Standards und Self-Hosting	MUST	Erledigt	Das Projekt verwendet intern nur offene Standards wie JSON oder Schnittstellen anderer Open-Source Projekte (bspw. Ollama).
Wachstumsfähigkeit und skalierbares Datenhandling -skalierbar von Pilot-Use-Case (1 Tsd. Dokumente) bis Unternehmensbreite (>10 Mio Dok.) ohne Neuarchitektur	COULD	Ausstehend	Die Unterstützung großer Datenquellen durch die Einbindung mehrerer Vektordatenbanken und einer föderierten Suche sind in dieser Basisversion noch nicht adressiert, aber im Konzept als zukünftige Erweiterung bereits vorgesehen.
Recht, Compliance und Governance			
Anforderung	**Priorität**	**Status**	**Adressierung**
EU-AI-Act Konformität	MUST	Erledigt	Die Basisversion wurde unter Berücksichtigung des EU-AI-Act entwickelt.
GDPR/DSGVO -z.B. personenbezogene Daten sind erkennbar, Pseudonymisierung und Lösch-Workflows	MUST	Erledigt	Die Basisversion erlaubt den Import, die Verarbeitung und das Löschen von Daten. Die Art der Daten ist dabei für die Basisversion unerheblich.
Datenschutz der Anwender -Die Eingaben der Nutzer sollen geschützt und nicht einsehbar sein	MUST	Erledigt	Die Basisversion nimmt keiner Speicherung der Eingaben vor.
Security und Datenschutz			

[5] pyapp ist open-source (https://github.com/ofek/pyapp)

Anforderung	Priorität	Status	Adressierung
Data-in-Transit und -at-Rest Verschlüsselung -TLS 1.3, AES-256 oder ähnlich	SHOULD	Ausstehend	Bisher teilweise adressiert. So kann die Kommunikation durch die Verwendung von HTTPS verschl
Zugriffsrechtemanagement und Berechtigungskonzept -Der Zugriff auf Dokumente im RAG-System soll konsistent mit den Zugriffsrechten auf Informationen einstellbar sein -Metadaten für Kontext- und Berechtigungsfilter	COULD	Ausstehend	Eine Einbindung von Benutzerauthentifizierung ist in der Basisversion noch nicht adressiert, aber im Konzept als zukünftige Erweiterung bereits vorgesehen.
Betrieb und IT-Service-Management			
Anforderung	**Priorität**	**Status**	**Adressierung**
One-Click-on-Deploy on-prem und Cloud -Docker-Compose/K8s Helm Chart inkl. Default Konfiguration	COULD	Ausstehend	Bisher nicht adressiert.
Self-Service Konnektorverwaltung -Fachanwender können neue Datenquellen konfigurieren ohne Dev-Eingriff	COULD	Erledigt	Als Teil des Konfigurators implementiert.
Speicherortverwaltung -verschiedene Speicherorte sollen technisch zugänglich und verwaltbar sein	COULD	Erledigt	Als Teil des Konfigurators implementiert.
Funktionale Anforderungen			
Anforderung	**Priorität**	**Status**	**Adressierung**
RAG-Pipeline Die Basisversion muss eine RAG-Pipeline-basierte Lösung implementieren	MUST	Erledigt	Als Teil von E3.1 integriert.
Nachverfolgbarkeit Die Basisversion könnte bereits eine Nachverfolgbarkeit unterstützen (Herkunft von Daten bzw. Datenschnipseln als Teil der Antwort)	COULD	Erledigt	Als Teil von E3.1 integriert.
Modell-Flexibilität -Umschaltbar zwischen lokalen Open-Weight-Modellen und externen API-Modellen	COULD	Erledigt	Als Teil des Konfigurators implementiert.
Mehrsprachige Suche und Antwortgenerierung -mind. Deutsch und Englisch	SHOULD	Erledigt	Als Teil von E3.1 integriert.
Kontrolle über Quellen -System soll Kontrolle darüber ermöglichen, welche Informationsquellen einbezogen werden (z.B. Ausschluss einzelner Dokumente oder Websuche deaktivieren)	MUST	Erledigt	Als Teil des Konfigurators implementiert.
Kontextspezifisches Retrieval -Filtern möglich	COULD	Ausstehend	Bisher nicht adressiert.
Usability und Change Management			
Anforderung	**Priorität**	**Status**	**Adressierung**
Grafisch unterstützter Setup-Assistent - möglichst unter acht Eingaben bis zur lauffähigen Pipeline	MUST	Erledigt	Nach dem Entpacken und Starten der Learn2RAG Software kann eine fertige RAG-Pipeline durch 5 weitere Klicks erstellt und gestartet werden.
Mehrsprachiges User Interface - Deutsch und Englisch - konsistente Terminologie	COULD	Ausstehend	Die Benutzeroberfläche des Konfigurators existiert in Englisch und Deutsch.

- Übersetzungsfiles extern editierbar			
Qualität, Wartbarkeit und Erweitbarkeit			
Anforderung	**Priorität**	**Status**	**Adressierung**
Modularer Aufbau -klare standardisierte API-Schnittstellen pro Schicht (bspw. Ingestion, Embedding, Retrieval, Generation)	COULD	Erledigt	Siehe Übersichtsbeschreibung
Konfiguration-als-Code -gesamte Pipeline in YAML/JSON, versionierbar in Git	COULD	Ausstehend	Alle Konfigurationen (Sprachmodelle, Datenquellen, Pipelines) werden als YAML-Dateien abgelegt. Diese werden allerdings noch nicht für einen Export zu einer einzelnen Datei zusammengefügt.
Dokumentation in geringen Umfängen (für Basisversion) -Entwicklungs-, Admin- und Benutzerhandbuch, stets synchron mit dem Release	MUST	Erledigt	Die Onlinedokumentation ist in 2 Sprachen unter https://docs.learn2rag.de/ verf