Big Data: DBA64

Graded Assignment Part 3: Azure AI Service

Assignment

eingereicht bei

Dr. Martin Prause
AKAD

von

Justin Stange-Heiduk
Hengstrücken 132
37520 Osterode am Harz
Telefon: 015233817587
Studiengang: Data Science
3. Fachsemester
Matrikelnummer: 8149363
Datum: 26.10.2024

1. Einleitung 1
2. Schritt für Schritt Anleitung 2
2.1 Einrichten von Konten und Verbindungen 2
2.1.1 Azure Account 2
2.1.2 Ressourcengruppe 2
2.1.3 Azure OpenAI 4
2.1.3.1 Text-Embedding-Modell 6
2.1.4 Azure AI Search 8
2.1.5 Speicherkonto 9
2.1.5.1 Datei in Speicherkonto ablegen 11
2.2 Definition von Vektorräumen und Suchindizes 12
2.2.1 Import und Vektorisierung der Daten mit Azure AI Search 12
2.2.2 Projekt in Azure AI Studio 16
2.2.3 Sprachmodell deployen 19
2.3 Erstellen eines Promts, eines Endpunkts und Zugreifen auf den Endpunkt 20
2.3.1 Playground 21
2.3.1.1 Daten Hinzufügen 22
2.3.2 Endpoint erreichen 24
3. Zusammenfassung 27

Abbildung 1: Suche Ressourcengruppe 3
Abbildung 2: Ressourcengruppe erstellen 3
Abbildung 3: Ressourcengruppe Erstellung 3
Abbildung 4: Azure OpenAI erstellen 4
Abbildung 5: Azure OpenAI Erstellung 5
Abbildung 6: Azure AI Service Network Type 6
Abbildung 7: Überblick Azure OpenAI 7
Abbildung 8: Azure OpenAI-Studio Überblick 7
Abbildung 9: Azure AI Search Erstellung 8
Abbildung 10: Azure AI Search alternative Region 9
Abbildung 11: Speicherkonto Erstellung 10
Abbildung 12: Speicherkonto Container erstellen 12
Abbildung 13: Azure AI Search Importieren und Vektorisieren von Daten erstellen 13
Abbildung 14: Azure AI Search Importieren und Vektorisieren von Daten: Mit Ihren Daten verbinden 13
Abbildung 15: Azure AI Search Importieren und Vektorisieren von Daten: Ihren Text vektorisieren 14
Abbildung 16: Azure AI Search Importieren und Vektorisieren von Daten: Überprüfen und erstellen 14
Abbildung 17: Datenvektor 15
Abbildung 18: JSON Ergebniss der Vektorisierung 15
Abbildung 19: Azure AI Studio Projekt erstelen 17
Abbildung 20: Azure AI Studio Projekt anpassen 17
Abbildung 21: Azure AI Studio Projekt Erstellung 18
Abbildung 22: Azure AI Studio fertig erstelltes Projekt 19
Abbildung 23: Azure AI Studio Modell bereitstellen 20
Abbildung 24: Playground öffnen 21
Abbildung 25: Chatbot Test ohne RAG 22
Abbildung 26: Azure AI Studio Daten hinzufügen 23
Abbildung 27: Azure AI Studio Daten hinzufügen: Datenfeld Zuordnung 23
Abbildung 28: Chatbot Test mit RAG 24
Abbildung 29: Beispielcode Chatbot 25
Abbildung 30: Azure AI Search Endpoint 26

1. Einleitung

Diese Anleitung bietet eine detaillierte, schrittweise Einführung in die Erstellung eines Retrieval-Augmented Generation (RAG)-Endpunkts in Microsoft Azure. Sie richtet sich an Anwender, die eine praxisorientierte Herangehensweise an die Implementierung und Nutzung dieser fortschrittlichen Technologie suchen. Dabei werden sämtliche Aspekte der notwendigen Infrastruktur, von der Einrichtung bis zur Produktion, abgedeckt. Die Dokumentation umfasst drei zentrale Bereiche:

- Einrichten von Accounts und Verbindungen: Hier wird erklärt, wie Azure-Konten und Ressourcengruppen erstellt und konfiguriert werden, um eine stabile Grundlage für die Integration von Azure OpenAI und Azure Cognitive Search zu schaffen. Dabei spielt insbesondere das Text-Embedding-Modell eine Schlüsselrolle, da es die Umwandlung von Text in Vektordaten ermöglicht.
- Definition von Vektorräumen und Suchindizes: Im Fokus dieses Abschnitts steht die Erklärung, wie Daten in Azure AI Search importiert, vektorisiert und als Suchindizes organisiert werden. Diese Vektorräume ermöglichen es dem RAG-System, relevante Informationen effizient zu durchsuchen und bereitzustellen. Ein besonderes Augenmerk wird auf die Rolle des Azure AI Studios gelegt, das als zentrale Plattform die Verwaltung des Projekts sowie der Workflows erleichtert.
- Erstellen eines Prompts und Endpoints: Abschließend wird beschrieben, wie ein geeigneter Prompt erstellt und der RAG-Endpunkt eingerichtet wird, sodass auf das Sprachmodell zugegriffen werden kann. Dieser Endpunkt ermöglicht es, datenbasierte Antworten zu generieren und die Funktionalität des RAG-Systems in Echtzeit zu nutzen.

Diese Anleitung zeigt detailliert auf, wie ein RAG-Endpunkt eigenständig in Azure aufgesetzt, konfiguriert und für die Generierung von KI-gestützten Antworten genutzt werden kann. Sie führt Anwender sicher durch alle notwendigen Schritte und legt den Grundstein für den erfolgreichen Einsatz von RAG in produktiven Umgebungen.

2. Schritt für Schritt Anleitung

In diesem Kapitel werden alle notwendigen Schritte aufgeführt und erklärt, von der Erstellung eines Azure-Accounts bis hin zur Nutzung des RAG-Endpoints.

2.1 Einrichten von Konten und Verbindungen

Zum Beginn wird ein Azure-Account und anschließend daran eine Ressourcengruppe erstellt um alle relevanten Ressourcen zu organisieren. Danach werden die Azure-Dienste wie Azure OpenAI und AI Search konfiguriert, um die Textgenerierung und die KI-basierte Suche zu ermöglichen. Wenn das erledigt ist werden die Verbindungen zu den Datenquellen hergestellt, die für die Suche verwendet werden sollen.

2.1.1 Azure Account

Azure Accounts sind Benutzerkonten, die benötigt werden, um auf die Cloud-Dienste von Microsoft Azure zuzugreifen und diese zu verwalten. Sie ermöglichen es Nutzern, Ressourcen wie virtuelle Maschinen, Datenbanken, Speicherlösungen und KI-Dienste in der Azure-Cloud zu erstellen, zu konfigurieren und zu überwachen. Ein Azure-Account ist die Grundlage für die Nutzung aller Dienste und die Verwaltung der zugehörigen Abrechnung und Sicherheitsrichtlinien.

Unter folgendenen Link kann ein neuer Azure Account angelegt werden: Konto erstellen (live.com)

Hier sind die wichtigsten Punkte, auf die man beim Erstellen eines Azure-Accounts achten sollte:

- E-Mail-Adresse: Verwende eine gültige E-Mail-Adresse, die für die Kontoerstellung benötigt wird.
- Identitätsverifizierung: Bereite dich auf die Verifizierung deiner Identität vor, z. B. durch das Senden eines Codes an deine E-Mail oder Telefonnummer.
- Passwortsicherheit: Wähle ein sicheres Passwort, das den Sicherheitsanforderungen entspricht.
- Abrechnungsinformationen: Gib gegebenenfalls Abrechnungsinformationen an, auch wenn ein kostenloses Konto erstellt wurde.
- Tarifauswahl: Entscheide dich für den geeigneten Tarif, z. B. kostenloses Konto, Studententarif oder kostenpflichtiges Abonnement.
- Nutzungsbedingungen: Lies und akzeptiere die Nutzungsbedingungen von Microsoft Azure.

2.1.2 Ressourcengruppe

Eine Ressourcengruppe in Azure ist eine Einheit, die verschiedene Ressourcen wie Speicher, Datenbanken und Dienste zusammenfasst, um sie gemeinsam zu verwalten und zu organisieren. Sie

erleichtern dadurch die Verwaltung und Überwachung von Azure-Ressourcen, indem sie eine zentrale Struktur für Zugriffsrechte und Abrechnungen bieten.

In Azure Portal angekommen erstellen wir die Ressourcengruppe indem zuerst die Suchen und dann darauf klicken

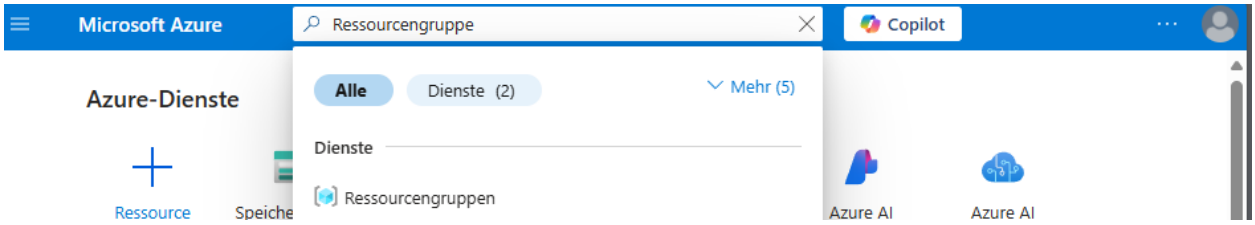

Abbildung 1: Suche Ressourcengruppe

Dann auf **erstellen** drücken.

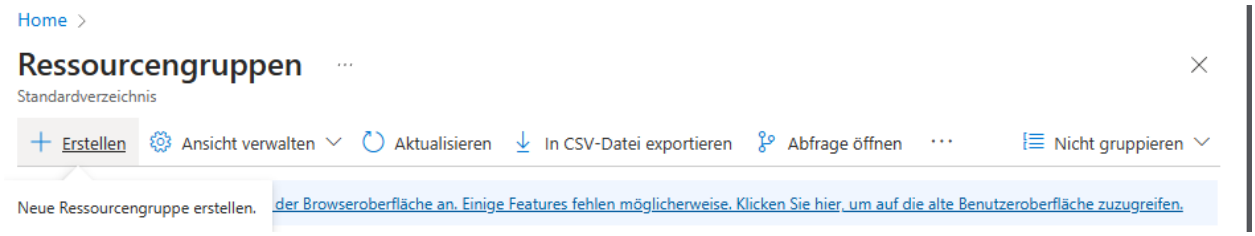

Abbildung 2: Ressourcengruppe erstellen

Im nächsten Schritt wird die Ressourcengruppe erstellt indem das Abonnement ausgewählt das beim Schritt Account erstellen angelegt wurde. Dann muss ein eindeutiger Ressourcengruppenname angegeben werden und die Region angegeben werden in welcher Server-Region es erledigt werden soll. Bei der Auswahl einer Region sollte eine aus Europa gewählt werden, um die Latenz zu minimieren, da der physische Standort der Datenverarbeitung näher am Nutzer ist. Darüber hinaus erfüllt die Wahl einer europäischen Region die Compliance-Anforderungen der Datenschutz-Grundverordnung (DSGVO) und anderer relevanter Vorschriften, die den Schutz personenbezogener Daten sicherstellen.

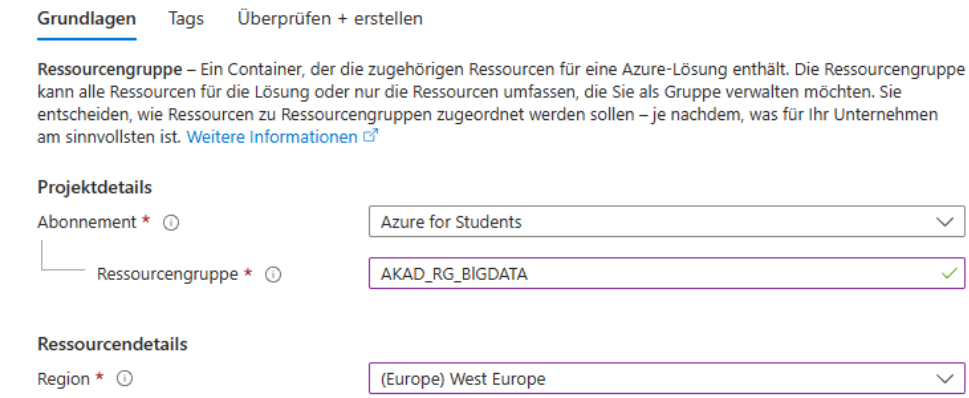

Abbildung 3: Ressourcengruppe Erstellung

Danach auf **Überprüfen & erstellen** drücken, bei Bedarf können auch Tags hinzugefügt werden um Informationen dazu zu speichern, wie: Wer hat die ressource wann erstellt etc.

2.1.3 Azure OpenAI

Azure OpenAI ermöglicht den Zugriff auf fortschrittliche KI-Modelle wie GPT, die für die Generierung natürlicher Sprache und komplexe Textverarbeitung genutzt werden. Nach der Erstellung der Ressourcengruppe ist dies der nächste Schritt, da Azure OpenAI die zentrale Komponente für die Generierung des KI-gestützten Inhalts im RAG-Endpoint ist.
Azure OpenAI ist ein Teil der Azure AI Services und bietet Zugang zu leistungsstarken Sprachmodellen wie GPT, die auf maschinellem Lernen basieren. Es ermöglicht die Integration generativer KI-Funktionen in Anwendungen, z. B. für automatische Textgenerierung, Übersetzungen oder Fragenbeantwortung. Innerhalb der Azure AI Services ist Azure OpenAI speziell auf natürliche Sprachverarbeitung (NLP) fokussiert und spielt eine Schlüsselrolle bei der Umsetzung eines RAG-Endpoints, indem es die generative Komponente für die Ausgabe von Antworten bereitstellt.

Um Azure OpenAI anzulegen suche diesen Azure-Dienst und drücke dann auf **erstellen.**

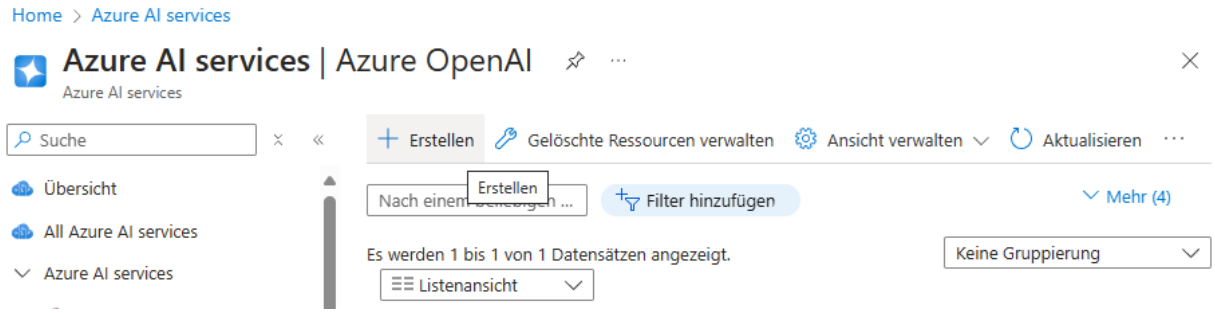

Abbildung 4: Azure OpenAI erstellen

Als nächstes die relevanten Infos eingeben wie bekannt.

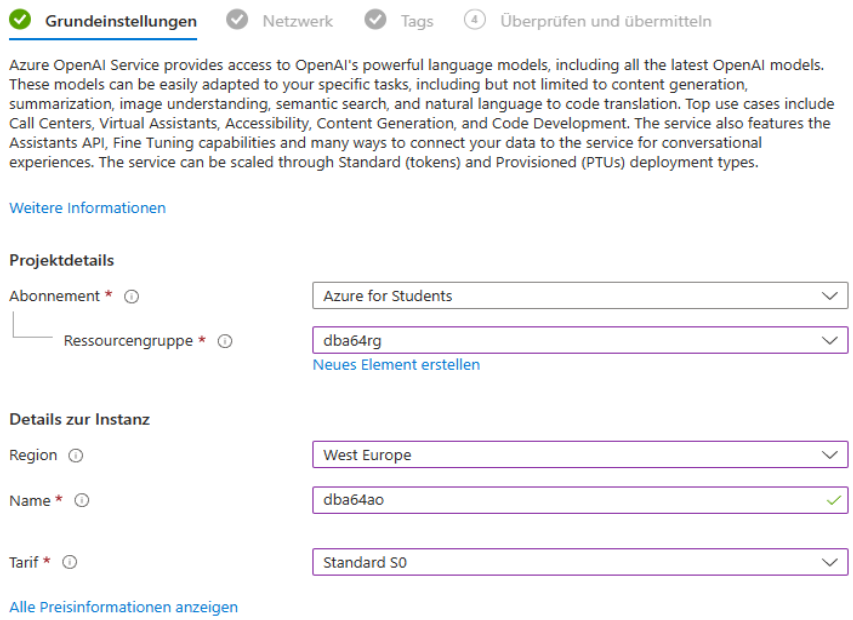

Abbildung 5: Azure OpenAI Erstellung

Danach auf **Weiter** drücken um zu den Netzwerk Einstellungen zu gelangen.

Eine kurze Übersicht was die angezeigten Netzwerk-Typen bedeuten:

- All networks, including the internet, can access this resource:

  Diese Einstellung ermöglicht es, dass alle Netzwerke, einschließlich des Internets, auf die Azure AI Service-Ressource zugreifen können.

  Nutzung: Diese Option ist nützlich, wenn eine öffentliche API oder einen Dienst benötigt wird, der von überall zugänglich sein soll.

- Selected networks, configure network security for your Azure AI services resource:

  Hier kann ein spezifisches Netzwerk ausgewählt werden, der Zugriff auf die Ressource haben. Hier kann das Subnetze in Azure oder IP-Adressen angegeben werden, die autorisiert sind.

  Nutzung: Diese Einstellung ist ideal, wenn die Sicherheit erhöht werden soll, indem nur bestimmten Netzwerken oder IP-Adressen Zugriff gewähren, beispielsweise innerhalb eines Unternehmensnetzwerks.

- Disabled, no networks can access this resource:

  Mit dieser Option wird der Zugriff auf die Ressource vollständig deaktiviert, sodass keine Netzwerke darauf zugreifen können.

  Nutzung: Diese Einstellung kann verwendet werden, um die Ressource zu schützen, während sie noch konfiguriert wird oder während Wartungsarbeiten.

Aus einfachheit soll hier und auch bei den folgenden: All networks, including the internet, can access this resource genutzt werden.

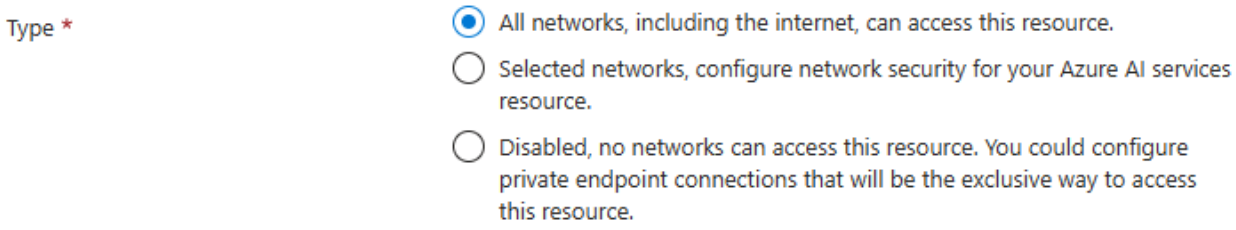

Abbildung 6: Azure AI Service Network Type

Danach kann dieser Azure-Dienst erstellt werden.

2.1.3.1 Text-Embedding-Modell

Ein Text-Embedding Modell wandelt Text in numerische Vektoren um, die die Bedeutung des Textes in einer maschinenlesbaren Form darstellen. Diese Vektoren fassen wichtige semantische Informationen zusammen und ermöglichen es, Texte effizient zu vergleichen und zu durchsuchen.

In einem RAG-Endpoint wird das Text-Embedding benötigt, um Daten in einer Weise zu strukturieren, dass sie durch einen Suchindex schnell gefunden und mit generativen Modellen kombiniert werden können. Dadurch lassen sich relevante Informationen aus großen Datenmengen extrahieren und präzise Antworten generieren.

Um solch ein Modell anzulegen, geht man auf die gerade angelegten Azure-Dienst Azure OpenAI.

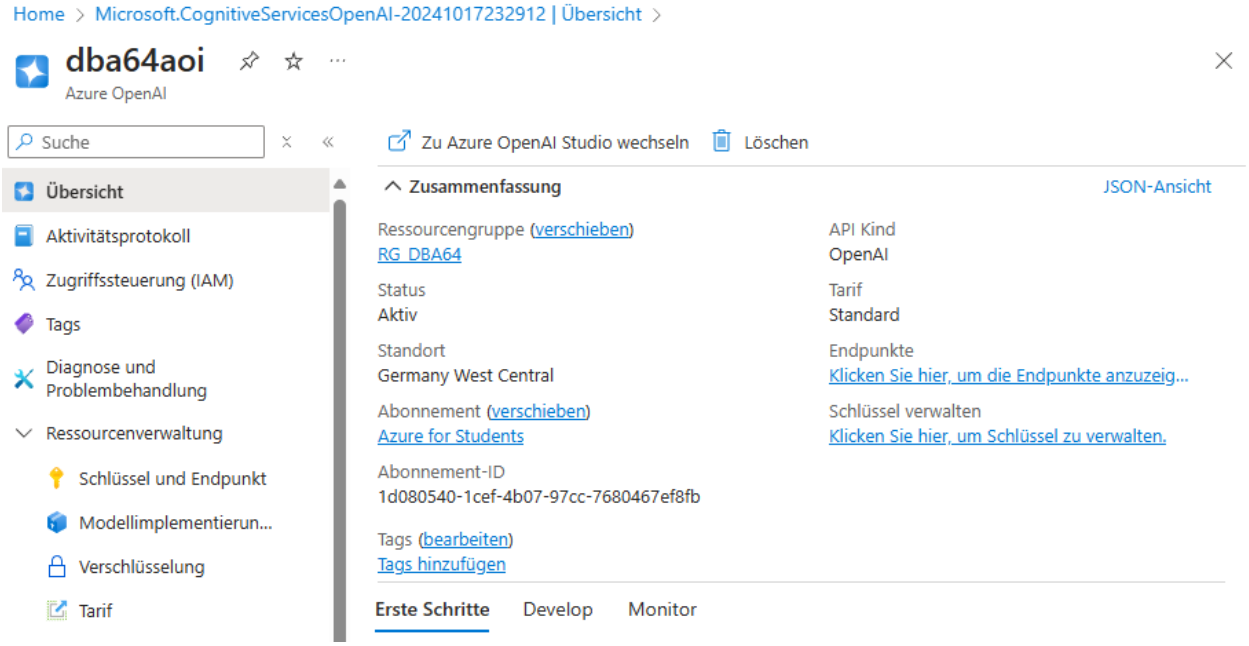

Abbildung 7: Überblick Azure OpenAI

Danach drückt man links unter Ressourcenverwaltung auf **Modellimplementierung** und dann auf **Bereitstellungen verwalten**. Anschließend öffnet sich das Azure OpenAI Studio. Das Azure OpenAI Studio ist eine Plattform, die es Nutzern ermöglicht, die KI-Modelle von OpenAI, wie GPT und Embedding-Modelle, in ihren Anwendungen zu nutzen. Es bietet eine benutzerfreundliche Oberfläche, um Modelle zu trainieren, zu testen und anzupassen, sowie verschiedene Konfigurations- und Management-Tools, um die Integration von generativer KI in Azure-Dienste zu erleichtern.

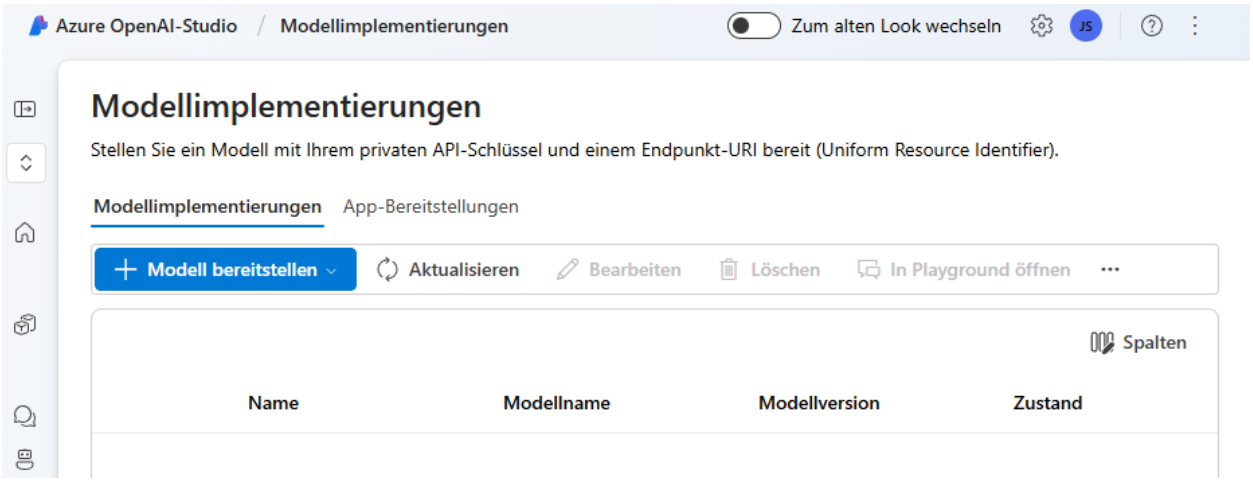

Abbildung 8: Azure OpenAI-Studio Überblick

Dann gehe auf **Modell bereitstellen**. Beim Bereitstellen eines Modells im Azure OpenAI Studio gibt es drei Optionen:

- Basismodell bereitstellen: Dies ermöglicht die Verwendung von vortrainierten Modellen von OpenAI, wie GPT-3 oder Embedding-Modelle, ohne Anpassungen.

- Optimiertes Modell: Hier wird ein Modell bereitgestellt, das auf spezielle Aufgaben oder Daten angepasst und optimiert wurde, um bessere Ergebnisse für bestimmte Anwendungsfälle zu erzielen.
- Importiertes Modell aus AzureML: Mit dieser Option kann ein Modell verwendet werden, das in Azure Machine Learning entwickelt oder trainiert wurde, um es nahtlos in Azure OpenAI zu integrieren.

Die Wahl hängt davon ab, ob ein Standardmodell genutz werden soll, es angepasstes, oder ein eigenes, bereits trainiertes Modell. In diesen Fall reicht ein Basismodell, deswegen wähle diese Option **Bereitstellen eines Basismodells**.

Für diese Zwecke wird das Modell text-embedding-ada-002 verwendet. Für Infos zu diesem Modell und Alternativen folge diesen Link: [New embedding models and API updates | OpenAI](#).

Im nächsten Fenster alle Standarteinstellungen behalten und auf **Bereitstellen** drücken.

2.1.4 Azure AI Search

Azure AI Search ist ein leistungsstarker Suchdienst, der es ermöglicht, große Mengen an Daten effizient zu durchsuchen und relevante Informationen präzise zu finden. In dieser Arbeit spielt Azure AI Search eine zentrale Rolle, da es die Grundlage für die semantische Suche in der RAG Architektur bildet. Nach der Einrichtung der Ressourcengruppe und der KI-Modelle wird Azure AI Search genutzt, um durchsuchbare Indizes zu erstellen, die es dem RAG-System ermöglichen, relevante Daten zur Generierung von Antworten zu nutzen.

Dieser Azure-Dienst wird gesucht unter „Suchdienst“ und dann muss dieser erstellt werden.

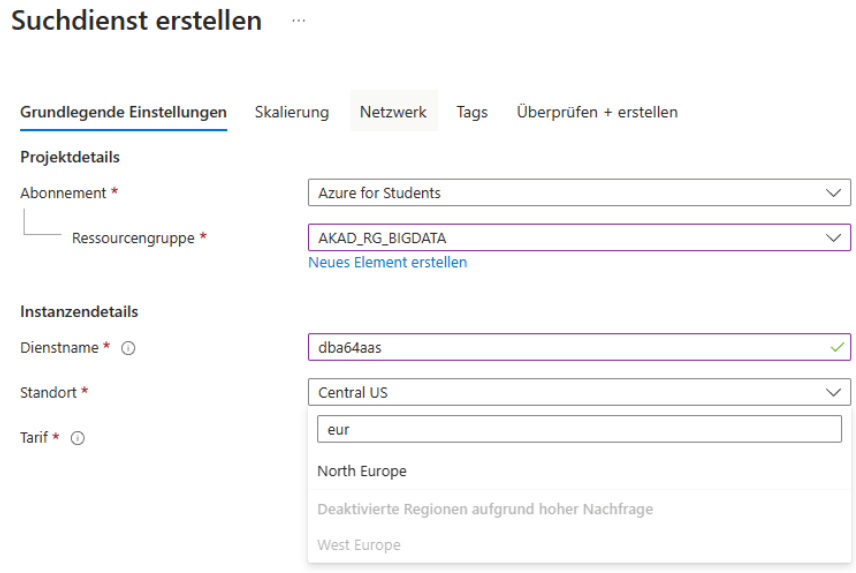

Abbildung 9: Azure AI Search Erstellung

Wie in Abbildung 9 zu sehen, kann die Region West Europa nicht gewählt werden. Es kann bei Azure vorkommen, dass bestimmte Azure-Regionen aufgrund hoher Nachfrage vorübergehend nicht verfügbar sind, was die Auswahl der üblichen Region für die Erstellung einer Ressource verhindert. Dies ist jedoch kein großes Problem, da Azure flexible Optionen bietet, um die Ressource in einer anderen Region bereitzustellen. Für das Projekt bedeutet dies lediglich, dass eine alternative Region ausgewählt werden musst. Es ist sinnvoll, eine Region in geografischer Nähe zu wählen, um die Latenz niedrig zu halten und weiterhin hohe Leistung zu gewährleisten, besonders wenn Compliance- oder Datenschutzanforderungen berücksichtigt werden müssen.

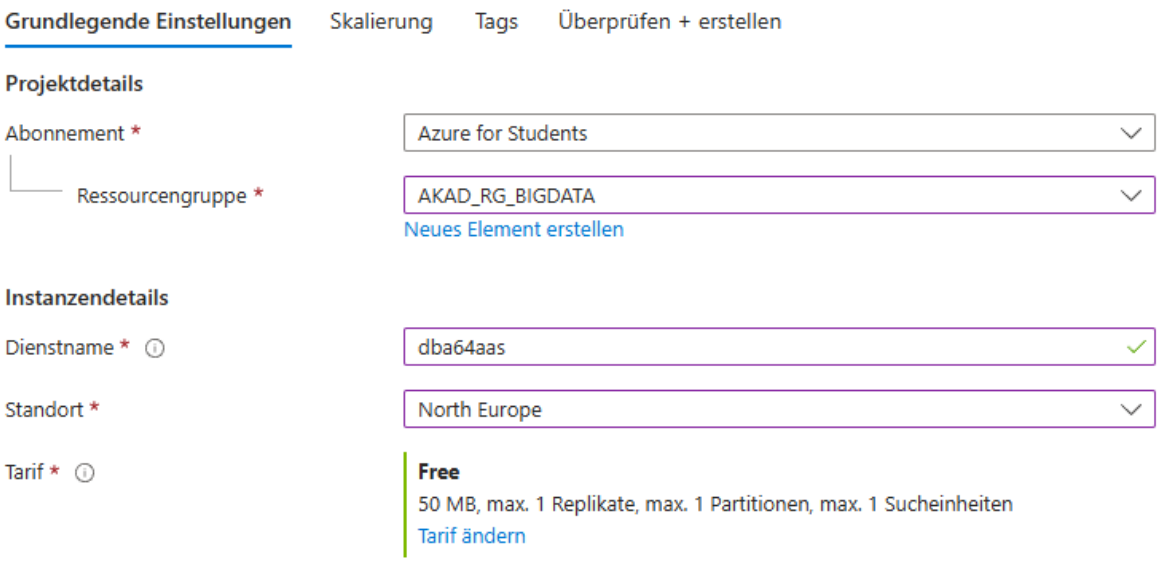

Abbildung 10: Azure AI Search alternative Region

Als Tarif kann diese gewählt werden die den Anforderungen des Projekts entspricht.

Beim Erstellen des Azure AI Suchdienstes gibt es die Option, die Skalierung festzulegen. Hier kann angegeben werden, wie viele Partitionen und Replikate verwendet werden, um die Leistung und Verfügbarkeit deines Suchdienstes zu optimieren. Mehr Partitionen ermöglichen es, größere Datenmengen zu verwalten, während zusätzliche Replikate die Ausfallsicherheit und Abfrageleistung erhöhen.

Für detaillierte Informationen zur Skalierung von Azure Cognitive Search kann dieser Link genutzt werden: Estimate capacity for query and index workloads - Azure AI Search | Microsoft Learn.

Für diese Dokumenation wird der Free Tarif genutzt und es können dort auch keine Einstellungen zu der Skalierung getroffen werden. Deswegen als nächstes **Überprüfen + Erstellen** drücken.

2.1.5 Speicherkonto

Ein Speicherkonto in Azure ist ein grundlegender Baustein für die Speicherung von Daten in der Cloud. Es bietet eine sichere, skalierbare und hochverfügbare Umgebung zum Speichern verschiedene Arten von Daten. Ein Speicherkonto in Azure spielt eine wichtige Rolle im Kontext von RAG, da es die

Grundlage für die Speicherung und den Zugriff auf die Daten bietet, die für das Abrufen und Generieren von Informationen verwendet werden.

Es wird wieder der Azure-Dienst gesucht und dann erstellt.

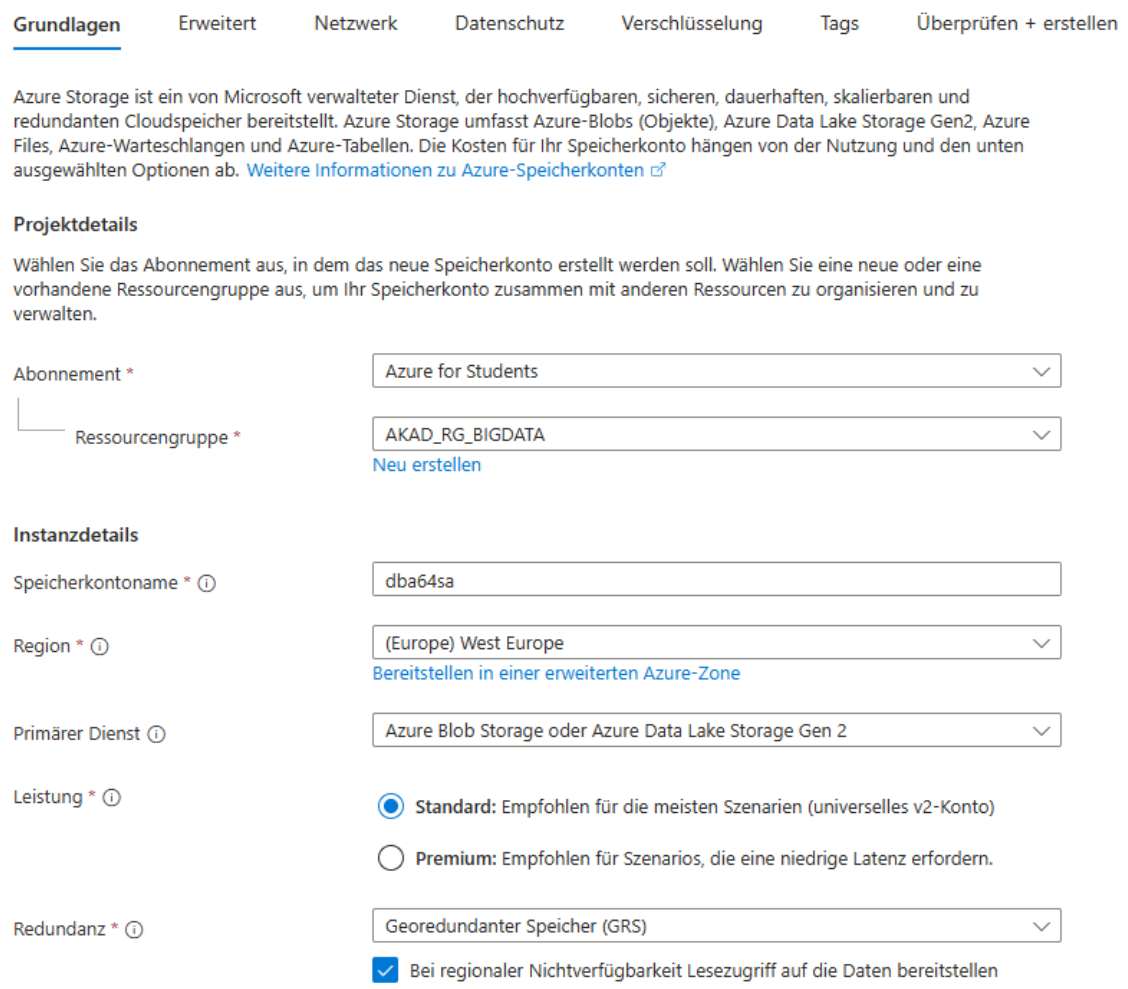

Abbildung 11: Speicherkonto Erstellung

Je nach Anforderungen des Projekts und die Art der Daten muss der Primäre Dienst, Leistung und Redundanz konfiguriert werden. In diesem Schritt wird nicht näher darauf eingegangen, sondern nehmen die Konfiguration wie in Abbildung 11 dargestellt. Die restlichen Einstellungen der anderen Reiter werden aus einfachshalber Gründen übernommen. Deswegen auf **Überprüfen + erstellen** drücken um das Speicherkonto zu erstellen.

2.1.5.1 Datei in Speicherkonto ablegen

Als nächstes wird ein Textdokument mit einer kurzen Geschichte in das Speicherkonto hinterlegt. Kurz ein Einblick in das Textdokument:

Titel: Der verschwundene Schlüssel

Autor: Max Müller

In einem kleinen, beschaulichen Dorf lebte ein alter Mann namens Herr Schmidt. Er war bekannt für seine Leidenschaft für das Gärtnern und hatte den schönsten Garten weit und breit. Eines Tages, als die Sonne strahlte und die Vögel sangen, beschloss Herr Schmidt, seine Blumen zu gießen.

Er ging in sein Haus, um den Schlüssel zur Gartentür zu holen. Doch als er in die Schublade griff, war der Schlüssel verschwunden! Er suchte überall – in der Küche, im Wohnzimmer, sogar unter seinem Bett – aber der Schlüssel blieb verschwunden.

Frustriert beschloss er, die Nachbarn um Hilfe zu bitten. Frau Meier, die immer ein offenes Ohr hatte, kam sofort vorbei. „Lass uns gemeinsam suchen!“, sagte sie. So durchkämmten sie das ganze Haus. Plötzlich entdeckte Herr Schmidt eine kleine Katze, die mit etwas Glänzendem im Maul spielte. Es war sein Schlüssel!

„Du freches Kätzchen!“, rief Herr Schmidt lachend und nahm den Schlüssel an sich. Er dankte Frau Meier und ging endlich in seinen geliebten Garten, um seine Blumen zu gießen.

Von diesem Tag an ließ er die Gartentür offen, damit die kleine Katze jederzeit hereinkommen konnte – schließlich war sie der Grund, warum er seinen Schlüssel wiedergefunden hatte.

Um dieses Textdokument mit diesem Inhalt in ein Speicherkonto zu speichern gibt es folgende Schritte.

Als erstes gehe auf den gerade erstellten Azure-Dienst Speicherkonto dann gehe links unter Datenspeicher auf **Container.** Anschließend gehe auf **+ Container** und gebe einen Namen an und drücke auf **Erstellen**.

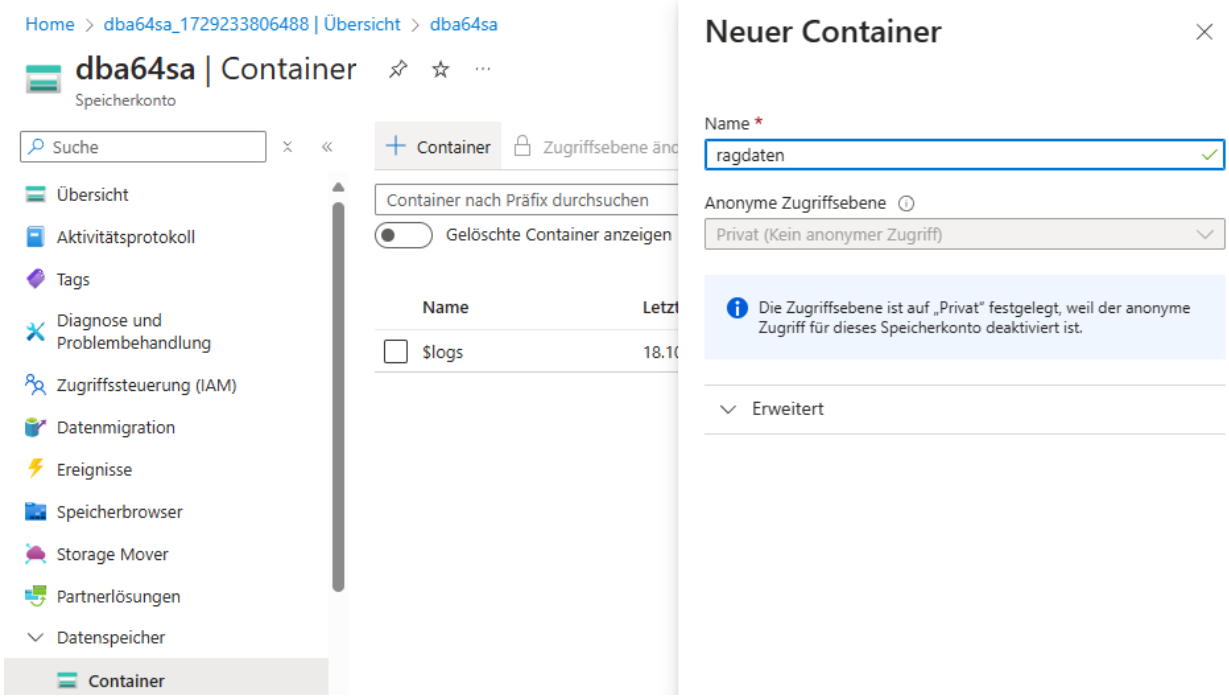

Abbildung 12: Speicherkonto Container erstellen

Anschließend drücke auf den erstellen Container und lade eine Datei hoch dir für das RAG genutzt werden soll.

2.2 Definition von Vektorräumen und Suchindizes

Nachdem alle relevanten Azure-Dienste angelegt wurden, werden jetzt die Vektorräume und Suchindizes erstellt.

2.2.1 Import und Vektorisierung der Daten mit Azure AI Search

In diesem Abschnitt wird der Prozess des Imports und der Vektorisierung von Daten mithilfe von Azure AI Search erläutert, um eine effiziente Suche und Analyse der Datenmengen zu ermöglichen. In Azure AI Search ist der Import von Daten der erste Schritt zur Erstellung einer durchsuchbaren Wissensbasis. Hierbei können verschiedene Datenquellen wie Datenbanken, Dokumente oder APIs integriert werden. Hier verwenden wir das Textdokument was in einen Container in dem Speicherkonto hinterlegt wurde.

Dazu gebe auf den Azure-Dienst Suchdienst und drücke dann oben rechts auf **Importieren und Vektorisierung von Daten.**

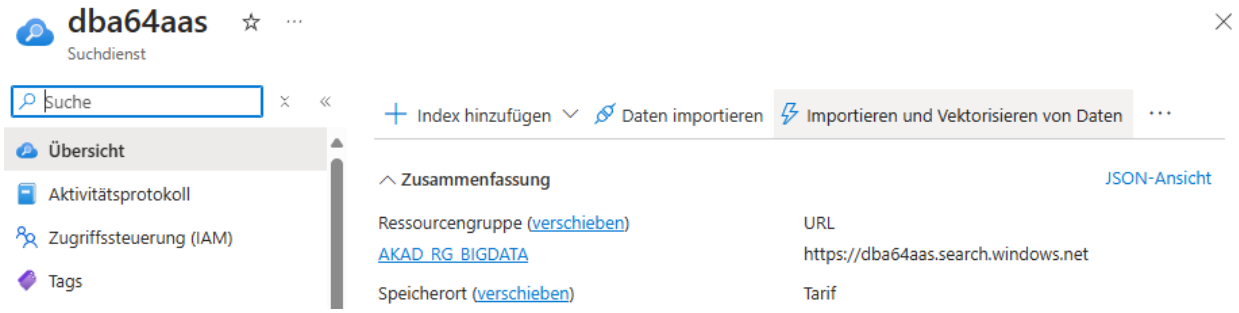

Abbildung 13: Azure AI Search Importieren und Vektorisieren von Daten erstellen

Als nächstes müssen die relevanten Daten angebunden werden.

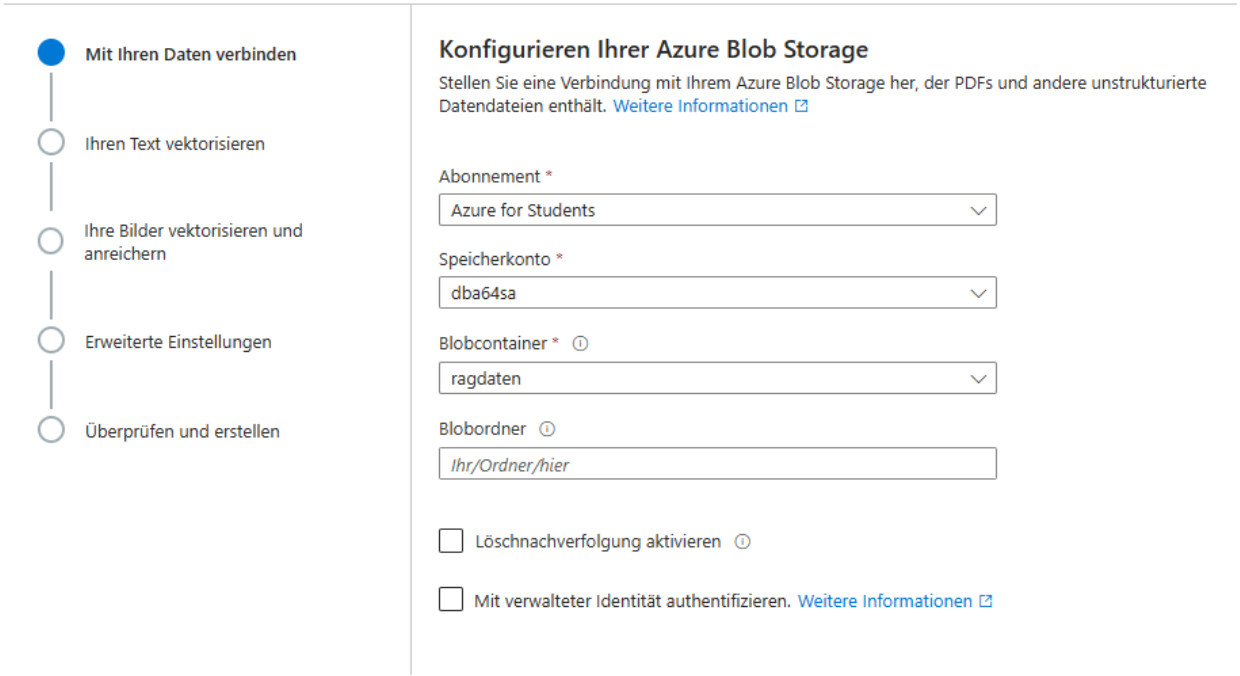

Abbildung 14: Azure AI Search Importieren und Vektorisieren von Daten: Mit Ihren Daten verbinden

Dazu wähle die Daten aus dem vorher abgelegten Daten aus dem Container im Speicherkonto.

Im nächsten Schritt müssen die Daten vektorisiert werden, in diesem Fall ein Textdokument. Vektorisierung eines Textes im Kontext von RAG bedeutet, dass der Text in eine mathematische Darstellung umgewandelt wird, die von Maschinen verstanden und verarbeitet werden kann. Genauer gesagt, wird der Text in einen numerischen Vektor transformiert. Dieser Vektor ist eine Folge von Zahlen, die die semantische Bedeutung des Textes erfassen, sodass ähnliche Texte ähnliche Vektoren haben.

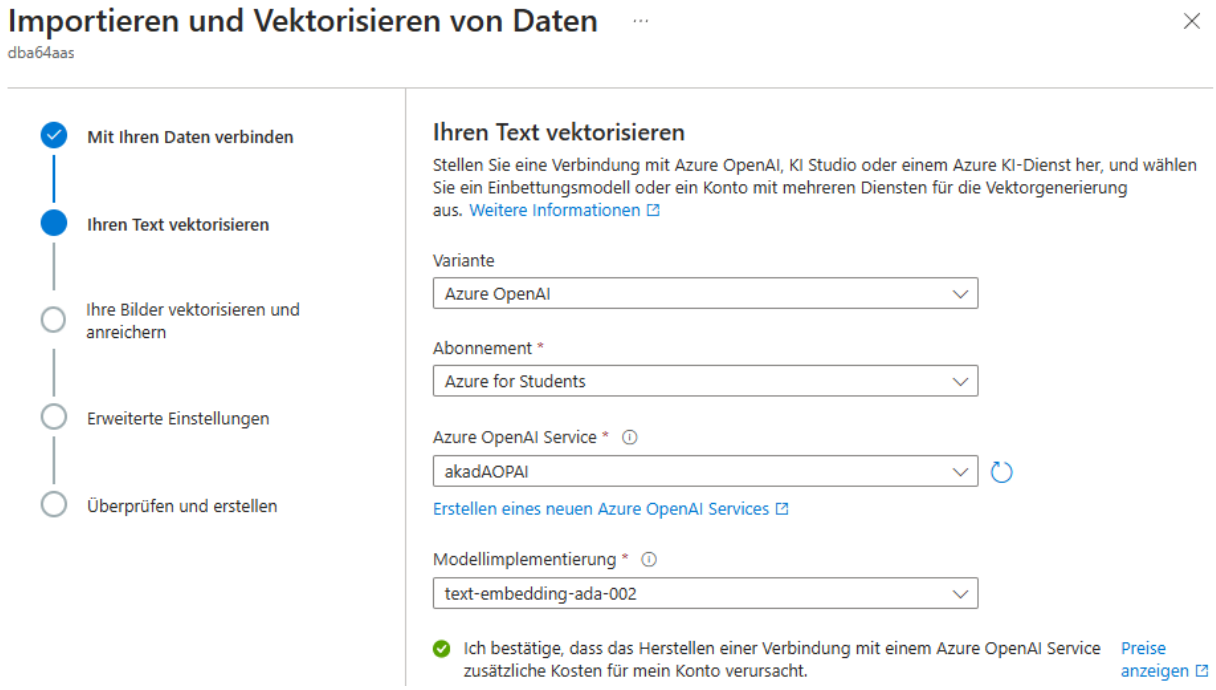

Abbildung 15: Azure AI Search Importieren und Vektorisieren von Daten: Ihren Text vektorisieren

Hier wird der Azure-Dienst Azure OpenAI und das Modell text-embedding-ada-002 verwendet, was bereits vorher angelegt wurde.

Danach kann bereits auf Überprüfen und erstellen gegangen werden. Die anderen Konfigurationen sind für diesen Zweck nicht relevant. Da weder Bilddaten genutzt werden noch Daten die ständig aktualisiert werden müssen.

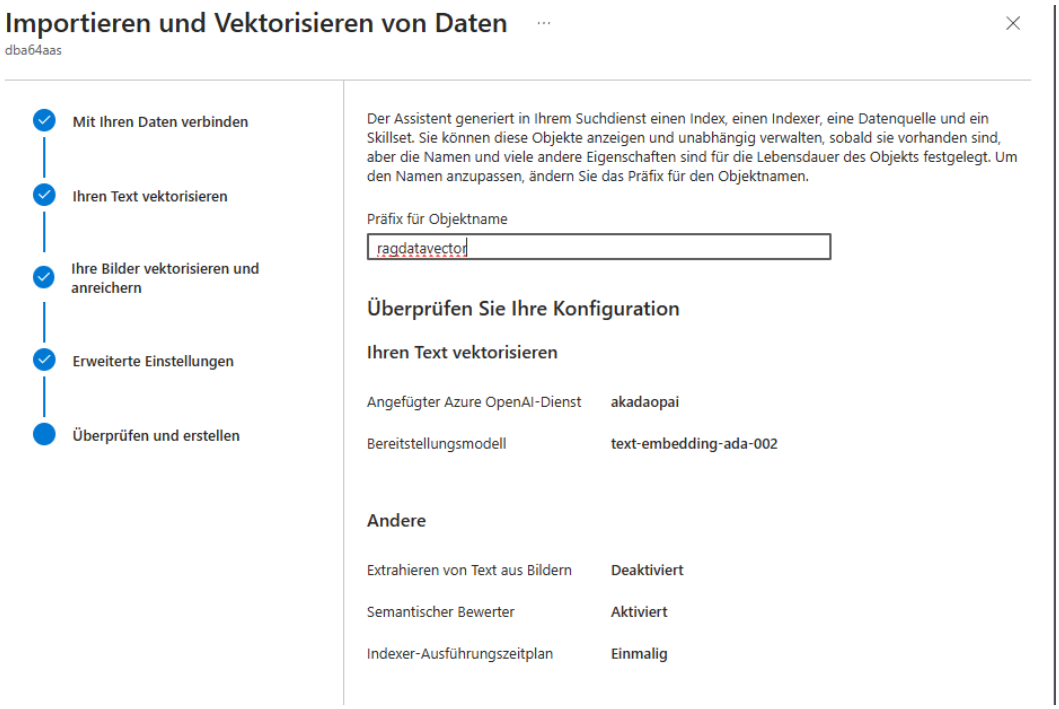

Abbildung 16: Azure AI Search Importieren und Vektorisieren von Daten: Überprüfen und erstellen

Es muss noch ein Objektname für den Index festgelegt werden und dann auch **Erstellen** drücken.

Als nächstes auf den eben erstellten Index, hier *ragdatavector, gehen*. Das ist dann der erstellte Index des Textdokuments das wir in den Speicherkonto ladeten.

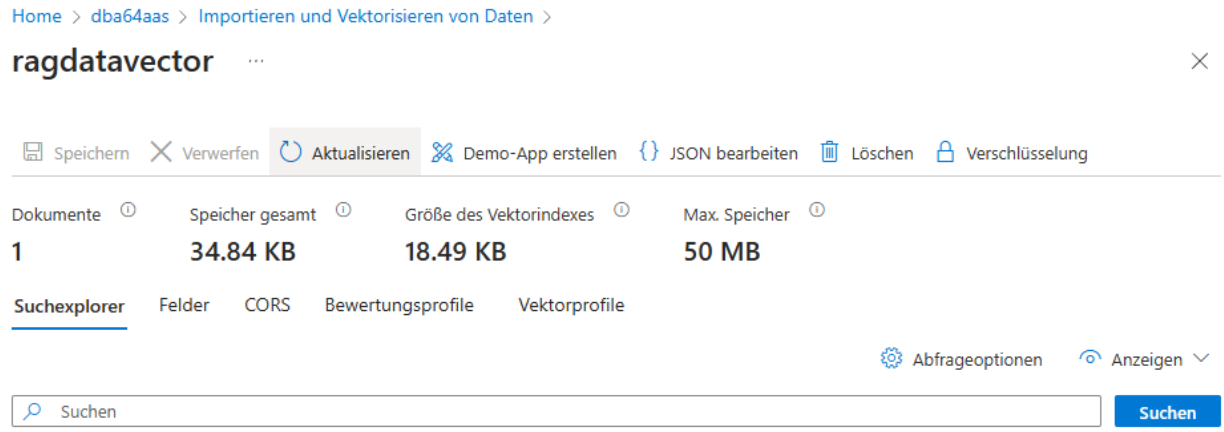

Abbildung 17: Datenvektor

Gegenfalls muss hier einmal **Aktualisieren** gedrückt werden um die neuen Informationen zu laden.

Jetzt einmal links auf **Suchen** drücken um folgendes zu erhalten:

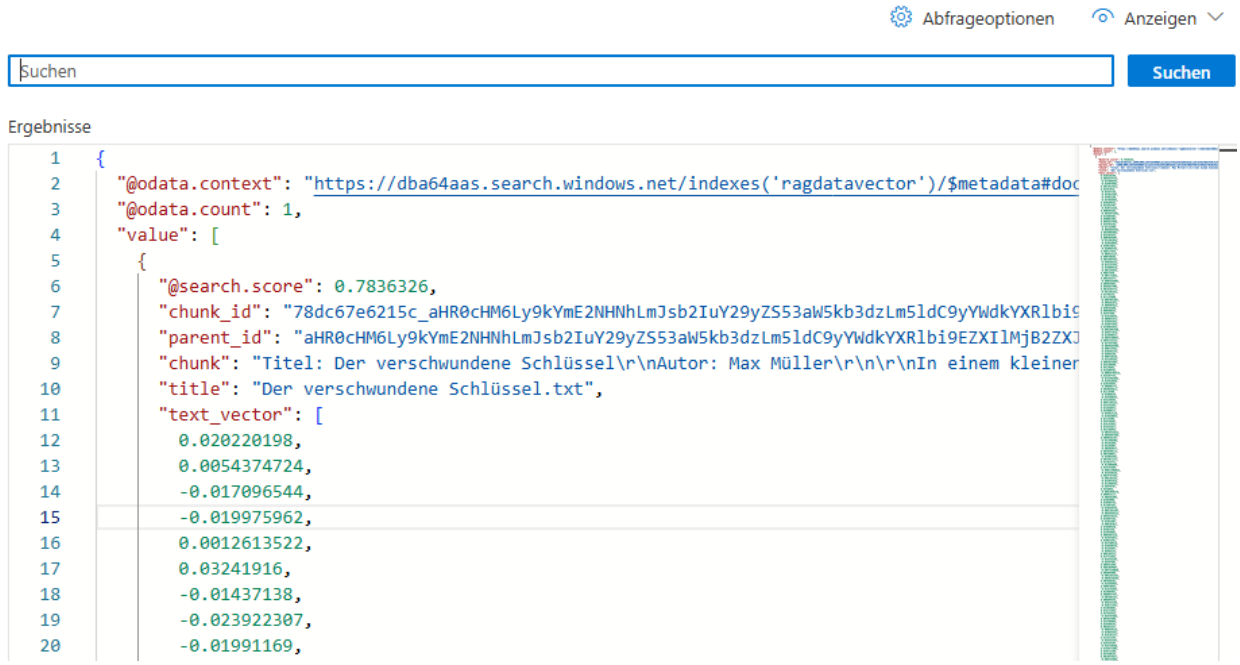

Abbildung 18: JSON Ergebniss der Vektorisierung

Der gezeigte JSON-Auszug repräsentiert das Ergebnis einer Vektorisierung eines Textes bei Azure AI Search, die typischerweise in einem RAG System verwendet wird.

Hier ist eine beschreibung der einzelnen Teile:

- @odata.context:
    - Dies zeigt den Kontext der Antwort. Es gibt an, dass die Daten aus einem bestimmten Index, hier ragdatavector stammen, der in Azure AI Search verwendet wird.
- @odata.count:
    - Diese Zahl zeigt an, dass eine Übereinstimmung (in diesem Fall nur 1 Ergebnis) für die Abfrage gefunden wurde.
- value (Array mit Ergebnissen):
    - Hier befindet sich das eigentliche Ergebnis. In diesem Fall enthält es die Details zum Text, der vektorisiert wurde.
- @search.score:
    - Dies ist der Relevanzwert der Suche. Er gibt an, wie stark das abgerufene Textstück mit der Abfrage übereinstimmt. Ein höherer Wert bedeutet eine höhere Übereinstimmung.
- chunk_id und parent_id:
    - chunk_id: Diese eindeutige ID identifiziert den spezifischen Textabschnitt (auch "Chunk" genannt), der vektorisiert wurde.
    - parent_id: Dies identifiziert das übergeordnete Dokument oder die URL, von der dieser Textabschnitt stammt.
- chunk:
    - Dies ist der eigentliche Textabschnitt, der in Vektorform umgewandelt wurde. In diesem Beispiel handelt es sich um die Geschichte „Der verschwundene Schlüssel“.
- title:
    - Der Titel der Datei, die den Text enthält. In diesem Fall ist es die Datei "Der verschwundene Schlüssel.txt".
- text_vector:
    - Dies ist der vektorisierte Text. Der Text wurde in eine numerische Repräsentation umgewandelt (Vektoren). Diese Liste von Zahlen codiert die semantische Bedeutung des Textes und ermöglicht es Azure AI Search, den Text effizient mit anderen vektorisierten Texten zu vergleichen.
    - Wofür ist der Vektor gut? Der Vektor dient dazu, semantische Ähnlichkeiten zwischen Texten zu erkennen. Ähnliche Texte haben ähnliche Vektoren, wodurch das System relevante Dokumente effizient abrufen kann.

2.2.2 Projekt in Azure AI Studio

Ein Projekt in Azure AI Studio dient als zentrale Arbeitsumgebung für die Entwicklung, Verwaltung und Bereitstellung von KI-Anwendungen. Es stellt die notwendigen Werkzeuge und Ressourcen zur Verfügung, um den gesamten Lebenszyklus eines KI-Projekts effizient zu gestalten. Ein Projekt in Azure AI Studio ist speziell dafür konzipiert, komplexe KI-Workflows zu steuern, von der Datenverarbeitung über das Modelltraining bis zur Bereitstellung der Lösung. Für RAG System spielt das Projekt in Azure AI Studio eine zentrale Rolle bei der Verwaltung der verschiedenen Phasen des Modelltrainings und der Datenverarbeitung. Es unterstützt die Vektorisierung großer Datenmengen und erleichtert die

Integration von Suchindizes, welche für die effektive Informationsretrieval-Phase von RAG entscheidend sind.

Um ein Projekt in Azure AI Studio zu erstellen gehe auf folgenden Link um zum Azure AI Studio zu gelangen: http://ai.azure.com. Wenn man beim Azure AI Studio über den Link auf die Website gelangt ist, meldet man sich mit dem gleichen Azure-Account an wie auch beim Azure Portal.

Als nächstes drücke oben Links auf **Neues Projekt**.

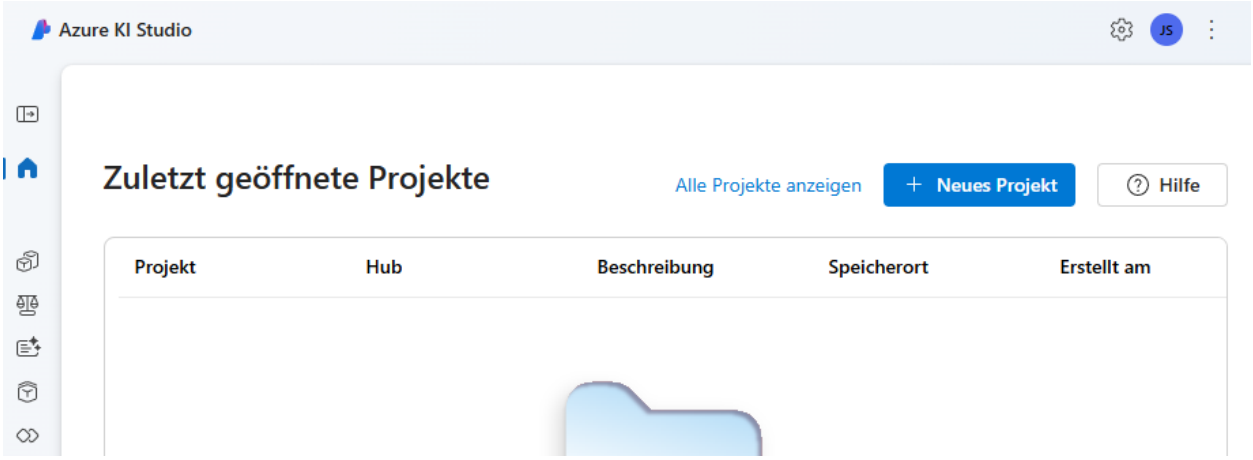

Abbildung 19: Azure AI Studio Projekt erstelen

Wird jetzt ein Projektname vorgeschlagen gehe bitte auf **Anpassen.**

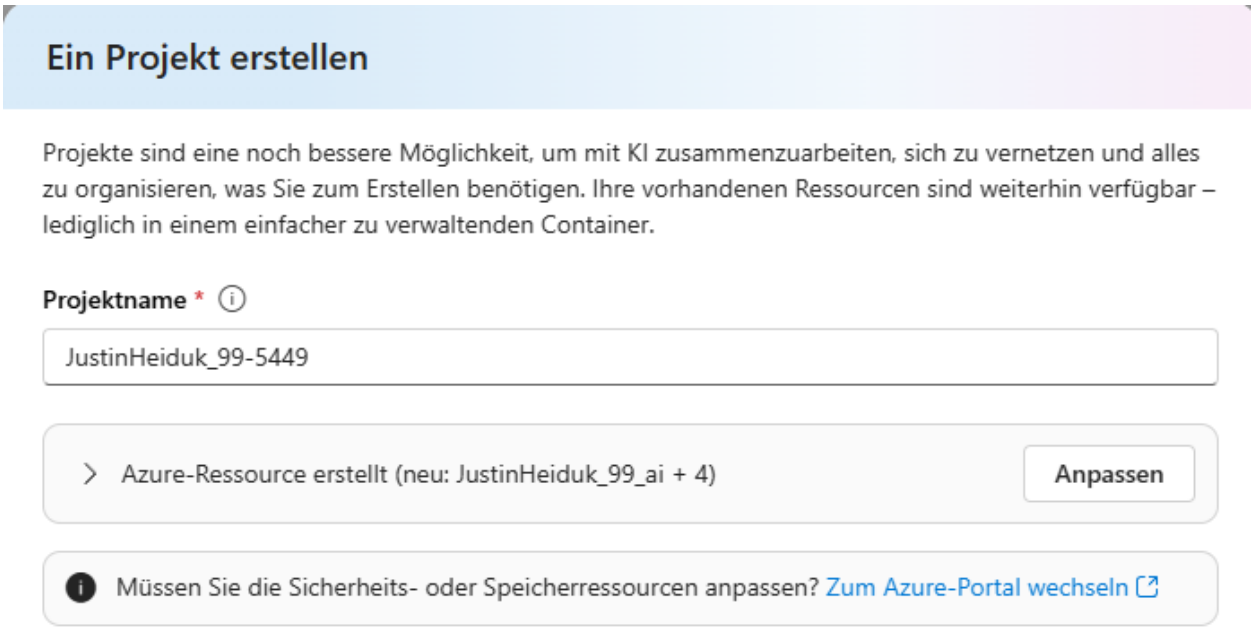

Abbildung 20: Azure AI Studio Projekt anpassen

Jetzt ist genau aufzupassen was alles anzugeben ist.

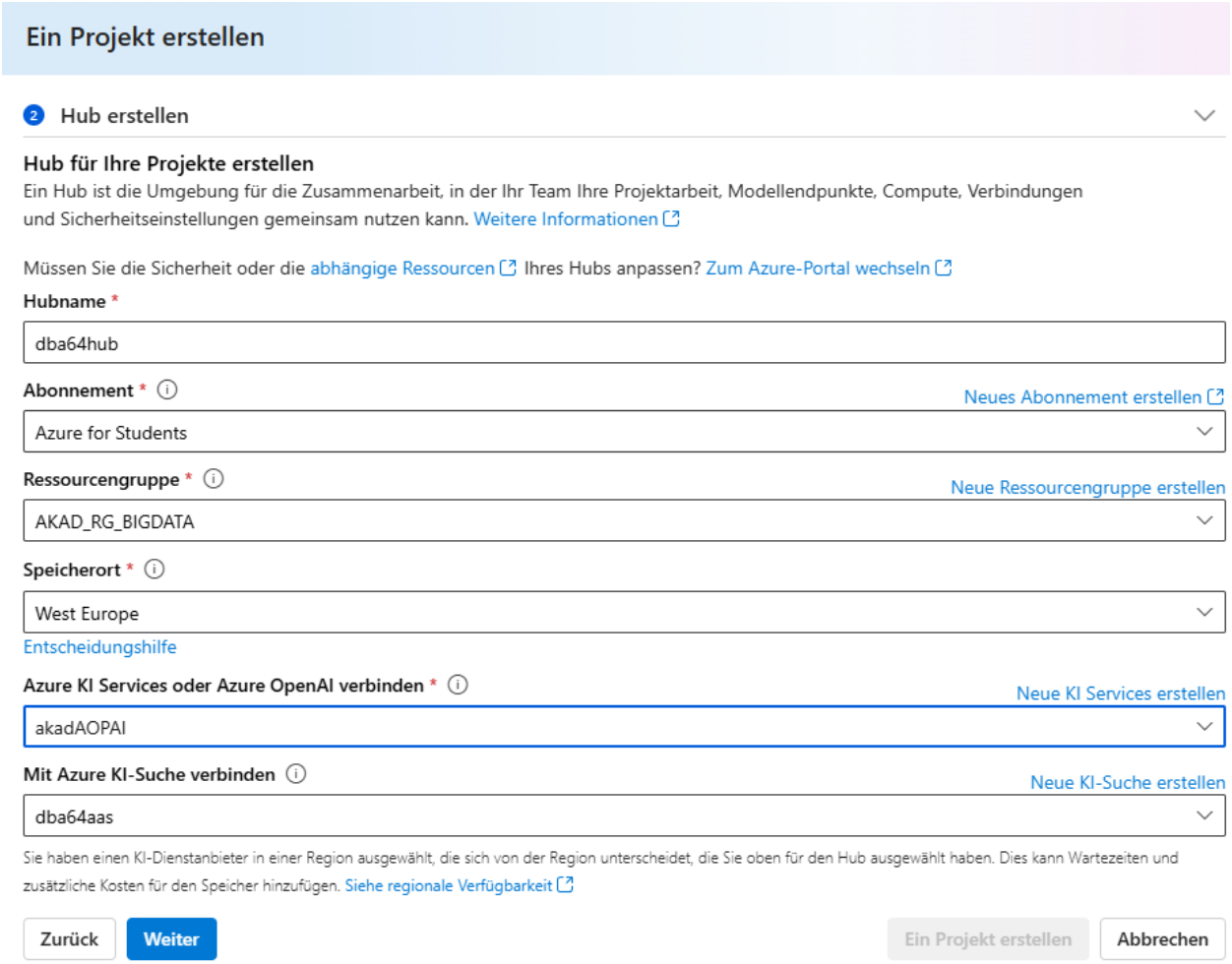

Abbildung 21: Azure AI Studio Projekt Erstellung

Hier wird innerhalb des Projekts ein Azure AI Hub erstellt. Ein Azure AI Hub ist eine zentrale Plattform innerhalb der Microsoft Azure-Infrastruktur, die verschiedene KI-Dienste, -Modelle und -Tools bereitstellt. Es besteht auch die Möglichkeit diesen Dienst wie gehabt in Azure Portal anzulegen indem Azure AI Studio gesucht wird daraufklickt und dann kann der Azure AI Hub angelegt werden.

Gebe wie in Abbildung 21 ein Hubnamen ein wähle das richtige Abonnement, die Ressourcengruppe und die Region. Als nächstes müssen die vorab erstellten Azure Dienste verbunden werden und zwar Azure Open AI, hier akadAOPAI, aus dem Kapitel 2.1.3 und Azure AI Search, hier dba64aas, aus dem Kapitel 2.1.4.
Nachdem alles richtig eingegeben wurde auf **Weiter** und danach auf **Ein Projekt erstellen** drücken. Danach wird für dieses Projekt automatisch ein Speicherkonto und ein Schlüsseltresor angelegt.

Es sollte wie folgt nun aussehen:

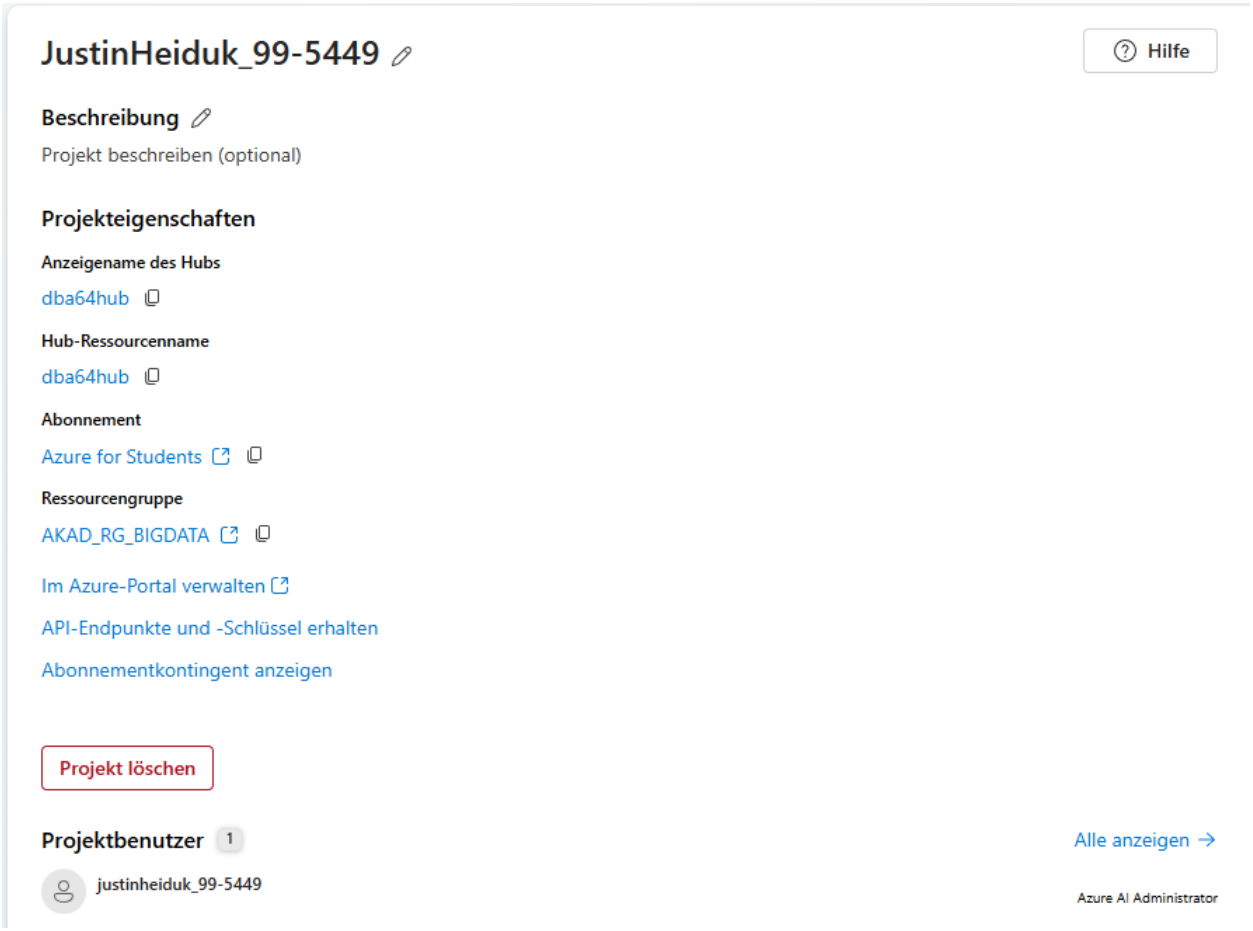

Abbildung 22: Azure AI Studio fertig erstelltes Projekt

2.2.3 Sprachmodell deployen

Das Deployen eines Sprachmodells ist ein entscheidender Schritt, um die Funktionalität eines RAG Systems voll nutzbar zu machen. Dabei wird ein vortrainiertes oder angepasste Sprachmodell in eine produktive Umgebung überführt, sodass es aktiv auf Anfragen reagieren und menschenähnliche Antworten generieren kann.

Um ein Sprachmodell zu deployen bleibe im Azure AI Studio und auf das gerade erstellte Projekt wie im Abbildung 22. Jetzt muss ganz links in der Leiste unten auf Bereitstellungen gedrückt werden.

Diese Ansicht kennt man von Kapitel 2.1.3.1, dort wurde das Text-Embedding Model bereitgestellt. Genau dieses sollte man auch in dieser Ansicht sehen können.

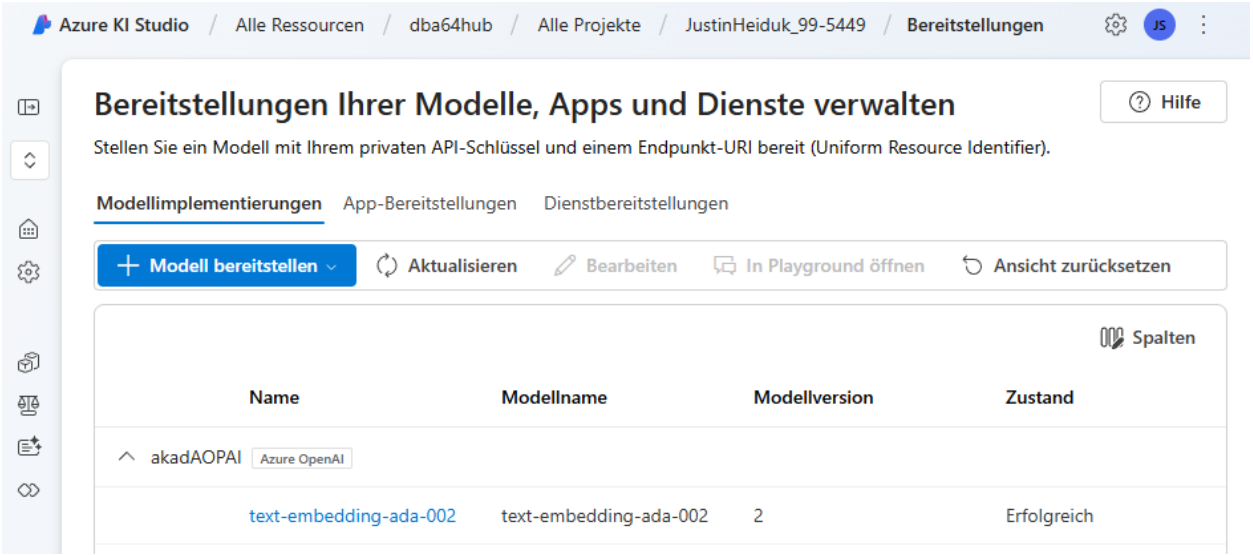

Abbildung 23: Azure AI Studio Modell bereitstellen

Danach gehe auf **Modell bereitstellen** und dann wieder auf **Bereitstellen eines Basismodells.** Nun kann ein entsprechendes Sprachmodell ausgesucht werden, hier nehmen wir *gpt-35-turbo.* Die Auswahl des Modells spielt hier keine Rolle sollte aber in einem echten Projekt gut überlegt sein. Unter diesen Link kann sich ein Bild gemacht werden welche Modelle die besseren sind und die Unterschiede zwischen diesen: 12 Best Large Language Models (LLMs) in 2024 | Beebom , The 15 Best Large Language Models (LLMs) in 2024 und Comparative Analysis of Top Large Language Models | Baeldung on Computer Science.

Nachdem das Modell ausgesucht wurde gehe auf **Bestätigen.** Im Anschluss kann alles auf die Standart Konfiguration gelassen werden, es sollte nur darauf geachtet werden das die verbundene KI-Ressourece, der Azure OpenAI Dienst, die aus Kapitel 2.1.3 ist, hier *akadOPAI*.
Zum Schluss auf **Bereiststellen** gehen um das Sprachmodell final bereitzustellen.

2.3 Erstellen eines Promts, eines Endpunkts und Zugreifen auf den Endpunkt

Es wurden alle relevanten Azure-Dienste erstellt und miteinander verbunden, sowie eine Textdatei in einen Vektor umgewandelt mit einem Embedding-Modell und zur Verfügung gestellt mit Azure AI Search.

Zum Schluss wird dieses bereitgestellten Sprachmodell aus Kapitel 2.2.3 mit unseren eigenen Daten, sprich das Textdokument mit einer Geschichte verbunden. Danach wird kurz experimentiert indem Promts diesen gestellt werden und analysiert werden. Danach wird kurz gezeigt wie der Endpunkt erstellt wird und dieser genutzt werden kann.

2.3.1 Playground

Im Playground werden alle die gerade beschriebenen Schritte durchgeführt. Darüber hinaus kann dort viel mehr gemacht werden sowie auch Modellparamenter angepesst werden wie Temperatur**,** Maximalanzahl an Tokens oder Antwortlänge.

Um zum Playground zu gelangen muss man im Azure AI Studio auf das bereitgestellte Sprachmodell gehen, hier *gpt-35-turbo*, danach auf **In Playground öffnen** drücken.

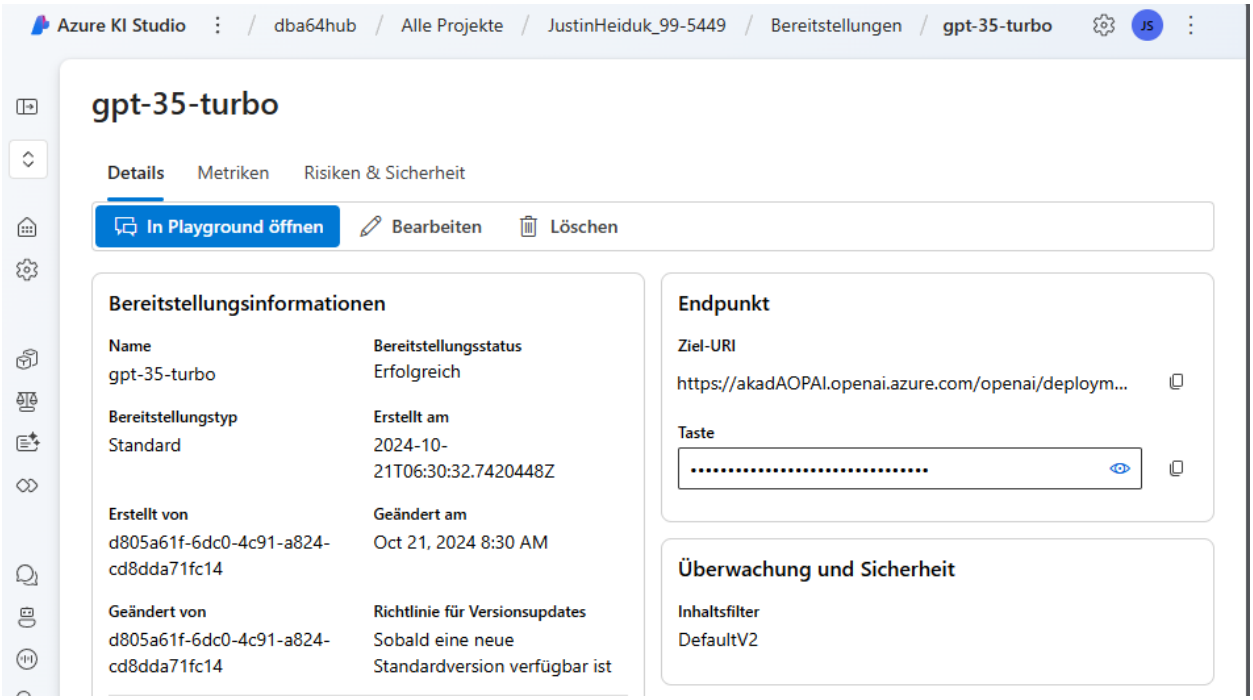

Abbildung 24: Playground öffnen

Bitte jederzeit drauf achten, dass das richtige Projekt geöffnet ist.

Nun wird getestet ob der Chatbot auf unsere Daten stellen kann indem gefragt wird: *Was ist die Geschichte: der verschwundene Schlüssel*. Zur Erinnerung diese Geschichte ist frei erfunden bzw. das Sprachmodell wurde nicht auf Basis dieser spezifischen Geschichte trainiert und kennt diese normalerweise nicht.

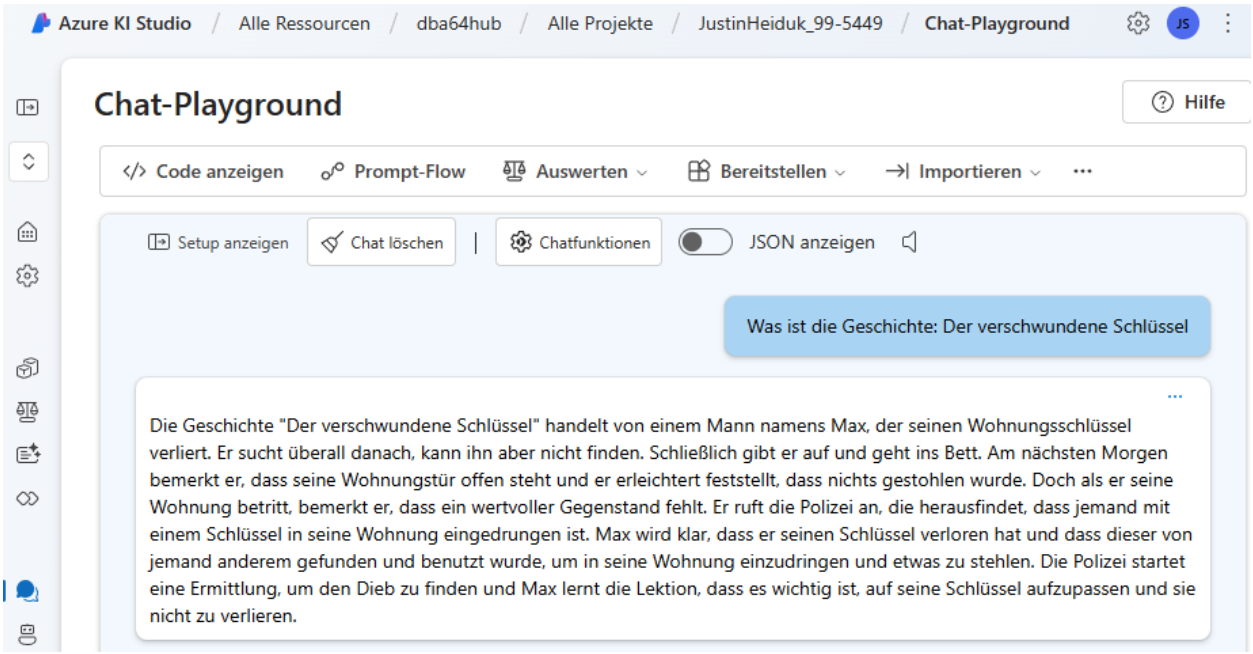

Abbildung 25: Chatbot Test ohne RAG

Es wird nicht unsere Geschichte zurückgegeben, sondern eine andere. Hier wird darauf verzichtet zu untersuchen was für eine Geschichte es dann ist, sondern nur zur Demostrations Zwecke gezeigt wird das es nicht die ist die im Textdokument beschrieben ist.

2.3.1.1 Daten Hinzufügen

Um nun den Chatbot mit unseren Daten bzw. den dazugehörigen Index zu verbinden dann gehe wie in Abbildung 25 ansonsten nochmal kurz beschrieben:
Gehe wieder ins Azure AI Studio auf das erstellt Projekt, dann unten Links wieder auf Bereitstellungen, dort dann wieder auf das bereitgestellte Sprachmodell, hier *gpt-35-turbo*, schließlich dann auf **In Playground öffnen** drücken. Dann sollte es wie in Abbildung 25 aussehen nur wieder ohne Chatverlauf.

Als nächstes drücke auf **Setup anzeigen** neben *Chat löschen*. Dort gibt es im Setup ein Reiter **Daten hinzufügen**, dort draufgeklickt, gehe auf **Neue Datenquelle hinzufügen.** Als Datenquelle wähle *Azure AI Search*, dann auf **Weiter.** Als nächstes muss der Azure AI Search ausgewählt werden und der Azure AI Search Index aus Kapitel 2.2.1, hier *ragdatavector*, ausgewählt werden.

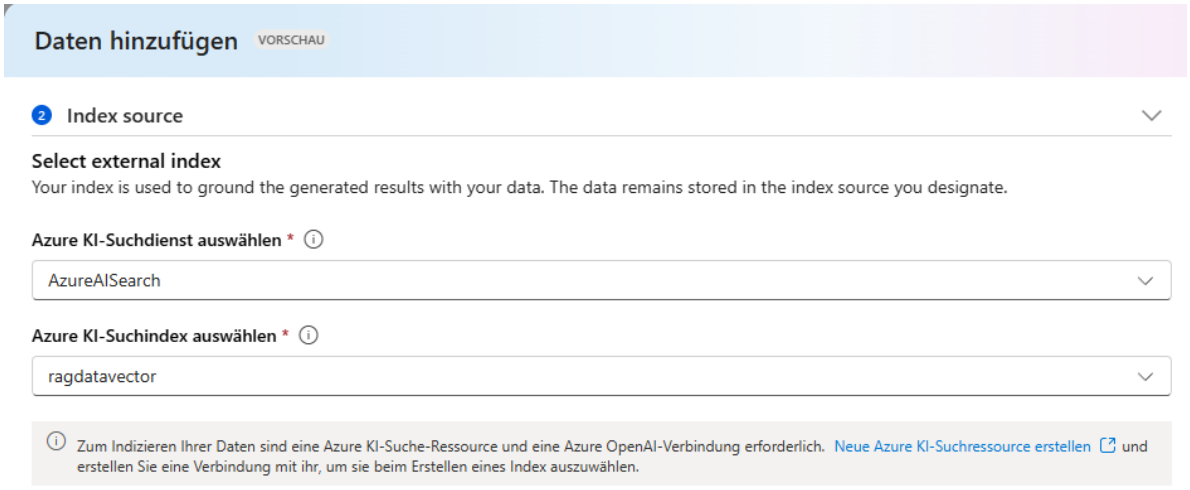

Abbildung 26: Azure AI Studio Daten hinzufügen

Dann auf **Weiter** drücken und als Azure OpenAI-Ressource, den Azure OpenAI Dienst aus Kapitel 2.1.3, hier *akad*AOPAI, hinzufügen und anschließend auf **Weiter** drücken. Als nächstes kommen die Indize Einstellungen, dort sollte der Indexname aus dem vorherigen Schritt schon drinstehen, außerdem sollte das Häckchen **an Benutzerdefinierte Feldzuordnung verwenden** gewählt werden da dies eine gezielte Anpassung der Datenstruktur und eine effiziente Zuordnung der Eingabedaten ermöglicht. Dann drücke auf **Weiter** um zum Datenfeld Zuordnung zu kommen. Wähle die Konfiguratio wie folgt:

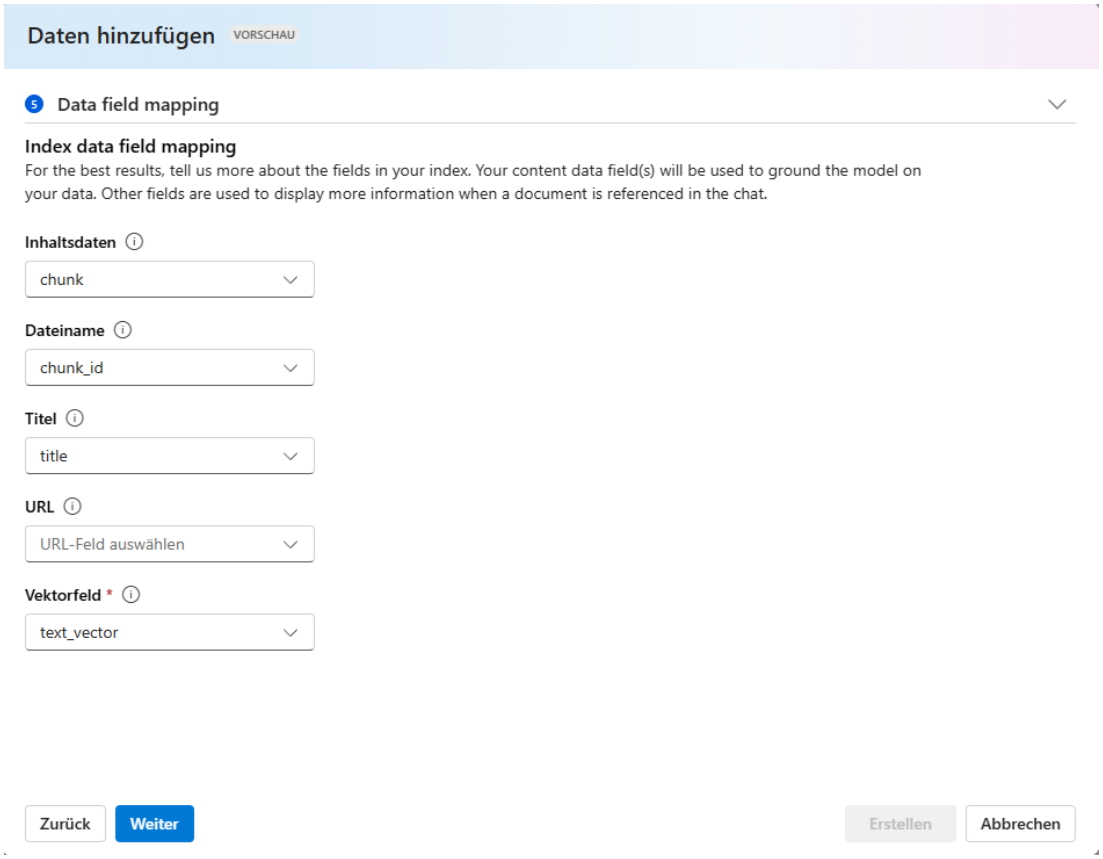

Abbildung 27: Azure AI Studio Daten hinzufügen: Datenfeld Zuordnung

Drücke dann auf **Weiter** und schließlich auf **Erstellen.** Nun wurden unsere Daten als Index dem Chatbot zur Verfügung gestellt.

Um zu überprüfen ob die Datenverbindung funktionierte wird den Chatbot die Frage gestellt: Was ist die Geschichte: Der verschwundene Schlüssel

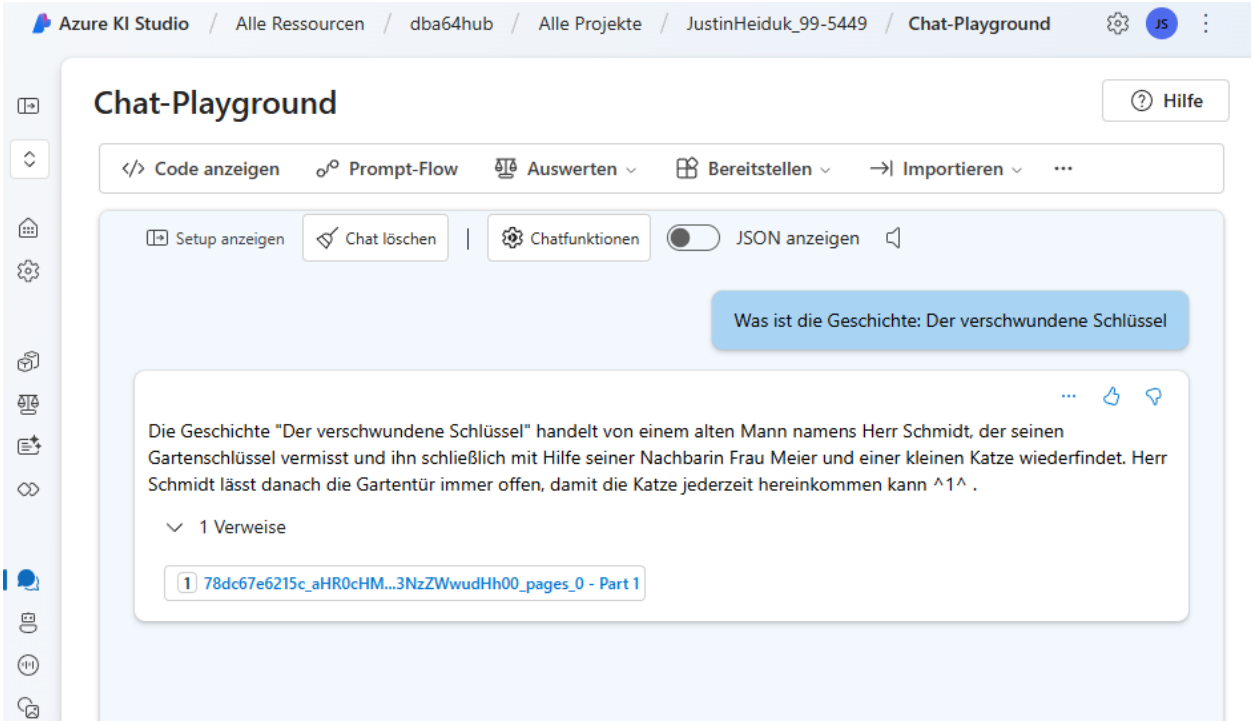

Abbildung 28: Chatbot Test mit RAG

Wie auf Abbildung 28 zu sehen ist, kann jetzt der Chatbot die Frage auf Basis der eigenen Daten beantworten und gibt die Geschichte aus dem Textdokument wieder. Damit ist erfolgreich ein Sprachmodell mit eigenen Daten über ein RAG System verbunden.

2.3.2 Endpoint erreichen

Zum Schluss soll noch kurz gezeigt werden wie das Spachmodell mit der RAG System Anbindung über einen Endpoint erreicht werden kann. Zum einen statt nur bei der GUI in Azure AI Studio den Chatbot weiter anzupassen kann das ganze auch per Code in mehreren beliebigen Sprachen angepasst werden und mit dafür erstellten Frameworks wie Langchain weiter verbessert werden zu können. Zum anderen den Chatbot den Endnutzer zur Verfügung zu stellen.

Im Playground kann zum einen oben Links auf **Code anzeigen** gedrückt werden.

Beispielcode

Sie können den folgenden Code verwenden, um mit der Integration Ihrer aktuellen Eingabeaufforderung und der Einstellungen in Ihre Anwendung zu beginnen.

https://akadAOPAI.openai.azure.com/ python

EntraID-Authentifizierung Schlüsselauthentifizierung

```python
import os
from openai import AzureOpenAI
from azure.identity import DefaultAzureCredential, get_bearer_token_provider

endpoint = os.getenv("ENDPOINT_URL", "https://akadAOPAI.openai.azure.com/")
deployment = os.getenv("DEPLOYMENT_NAME", "gpt-35-turbo")

# Initialize Azure OpenAI client with Entra ID authentication
cognitiveServicesResource = os.getenv('AZURE_COGNITIVE_SERVICES_RESOURCE', 'YOUR_COGNITIVE_SERVICES_RESOURCE')
token_provider = get_bearer_token_provider(
    DefaultAzureCredential(),
    f'{cognitiveServicesResource}.default'
)

client = AzureOpenAI(
    azure_endpoint=endpoint,
    azure_ad_token_provider=token_provider,
    api_version='2024-05-01-preview',
)

completion = client.chat.completions.create(
    model=deployment,
    messages=[
{
    "role": "system",
    "content": "Sie sind KI-Assistent und helfen Personen, Informationen zu finden."
},
{
    "role": "user",
    "content": "Was ist die Geschichte: Der verschwundene Schlüssel"
},
{
    "role": "assistant",
    "content": "Die Geschichte \"Der verschwundene Schlüssel\" handelt von einem alten Mann namens Herr Schmidt, der seinen Gartenschlüssel vermis
}
],
    past_messages=10,
    max_tokens=800,
    temperature=0.7,
    top_p=0.95,
    frequency_penalty=0,
    presence_penalty=0,
    stop=None,
    extra_body={
        "data_sources": [
            {
                "type": "azure_search",
                "parameters": {
                    "endpoint": os.environ["AZURE_AI_SEARCH_ENDPOINT"],
                    "index_name": os.environ["AZURE_AI_SEARCH_INDEX"],
                    "authentication": {
                        "type": "azure_ad"
                    }
                }
            }
        ]
    }
)
```

Abbildung 29: Beispielcode Chatbot

Zum einen ist ganz oben der Endpoint um den Sprachmodell Anfragen zu schicken. In den Code darunter sieht man wie mit Python Code den Endpoint erreicht, die Authentifizierung erledigt und wie dem

Sprachmodell über dessen Endpoint Anfragen gestellt werden. Hier wird auch die Frage: Was ist die Geschichte: Der verschwundene Schlüssel gestellt und die Antwort geliefert. Unten im Code stehen noch die Parameter für das Sprachmodell sowie welcher Index verwendet werden soll. In den Code Beispiel wurde nicht festgelegt was der *AZURE_AI_SEARCH_ENDPOINT ist.* Um diesen zu finden gehe ins Azure Portal suche dein *Azure AI Search Dienst* klicke darauf und dort findet man die URL von diesem Dienst und das ist der Endpoint der im Codebespiel angegeben werden muss.

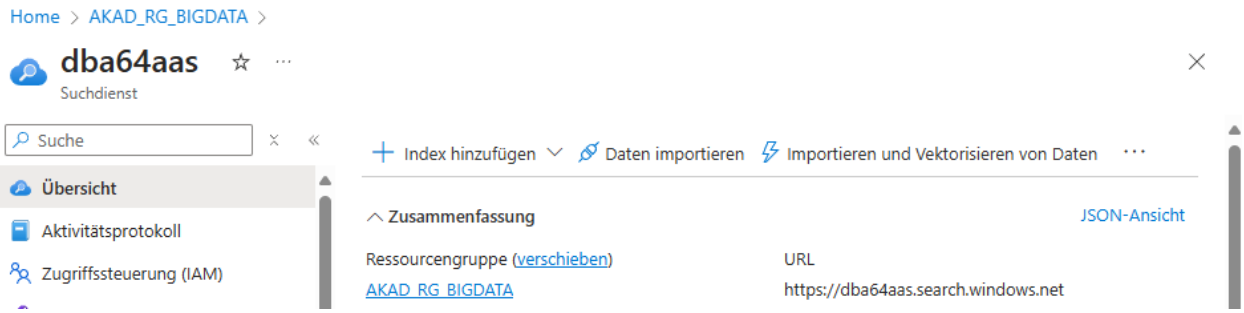

Abbildung 30: Azure AI Search Endpoint

Eine Zeile weiter drunter muss der *Index_name* angegeben werden, hier *ragdatavektor*. Auf den Azure AI Search Dienst gibt es in der linken Leiste unter *Indizes* eine übersicht über alle vorhandenen Indizes.

Zurück im Playground des Sprachmodells gibt es in der gleichen Leiste mit *Code anzeigen* auch den Reiter *Bereitstellen*. Dort kann das Sprachmodell mit den RAG-System als Web-App, neue Teams-App oder in einen neuen Copiloten in Copilot Studio bereitgestellt werden. In einer Web-App bereitstellen, kann ein Server mit Azure deployt werden, somit der Chatbot unter einer URL erreicht werden kann und Endnutzer diesen nutzen können.

3. Zusammenfassung

Diese Arbeit stellt eine schrittweise Anleitung zur Implementierung eines Retrieval-Augmented Generation (RAG)-Systems in Azure bereit. Der Prozess beginnt mit der Erstellung eines Azure Accounts, einer Ressourcengruppe und der Konfiguration von Azure OpenAI. Im Rahmen der OpenAI-Integration wird das Text-Embedding-Modell eingerichtet, welches für die Umwandlung von Texten in Vektordaten benötigt wird.

Im nächsten Schritt wird die Verbindung zu Azure AI Search aufgebaut, um einen Suchindex für die Vektorisierung der Daten zu erstellen. Ein Speicherkonto wird eingerichtet, um Dateien für die Suche bereitzustellen, bevor die Daten mit Azure AI Search importiert und vektorisiert werden. Die Definition von Vektorräumen und Suchindizes bildet die Grundlage für das RAG-System.

Ein Projekt im Azure AI Studio wurde erstellt, um als zentrale Plattform die verschiedenen KI-Modelle und Workflows zu verwalten. Im Anschluss daran erfolgt das Deployment des Sprachmodells, das Antworten auf Grundlage dieser indexierten Daten generiert. Indem die Daten als Index dem Sprachmodell zur Verfügung gestellt wurden. Schließlich wird der Prozess des Erstellens eines Prompts sowie eines Endpunkts beschrieben, um auf das Sprachmodell zugreifen zu können.

Diese Arbeit fasst somit die wesentlichen Schritte zusammen, die für den Aufbau eines RAG-Systems in Azure erforderlich sind, und zeigt, wie sich KI-Modelle und Suchtechnologien effizient kombinieren lassen, um fortschrittliche Informationserfassung und -verarbeitung zu ermöglichen.