FOM Hochschule für Oekonomie & Management

Hochschulzentrum Siegen

Seminararbeit

im Rahmen der Lehrveranstaltung
IT-Infrastruktur

über das Thema

Dialogbasierter Zugriff auf eine KI über eine Django REST API

von

Henning Beier

Betreuer :	Daniel Bitzer
Matrikelnummer :	517370
Abgabedatum :	27. August 2022

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

API Application Programming Interface
HTTP/S Hypertext Transfer Protocol (Secure)
KI Künstliche Intelligenz
REST Representational State Transfer

1 Einleitung und Zielsetzung

In dieser Seminararbeit soll eine Künstliche Intelligenz (KI)-Anwendung über eine REST[1] API[2] aufrubar gemacht werden. Eine KI-Anwendung muss dazu in einer Art eingebunden werden, die es erlaubt auf diese selbst in geeigneter Weise über die Endpunkte einer API zuzugreifen. Das bedeutet, dass die KI-Anwendung als Webservice zur Verfügung gestellt wird und damit auch einen diaglobasierten Zugriff auf die KI-Anwendung und dessen Funktionen selbst ermöglicht.

Verwendet werden dazu soll Django, ein quelloffenes Web-Framework das weit verbreitet im produktiven Einsatz ist und ebenso wie viele KI-Anwendungen in der Programmiersprache Python geschrieben wurde. Dieses zusammenführen und bereitstellen als eine Lösung im Softwarebereich, bzw. hier als Dienst, stellt dabei einen typischen Arbeitsbereich der IT-Infrastruktur dar. Die KI-Anwendung muss also in eine geeignete IT-Infrastruktur eingebunden werden. Ein sicherer und belastbarer Webservice, ist das angestrebte Ziel dieser Seminararbeit.

1.1 Praktische Relevanz und Problemstellung

Aktuell werden in vielen Bereichen neue Projekte und Lösungen mit KI entworfen, entwickelt und getestet. Dazu werden KI-Frameworks wie TensorFlow, PyTorch oder Keras benutzt. Für die vorgenannten, aber ebenso wie auch für andere KI-Frameworks, hat sich Python als Programmiersprache durchegsetzt. Dies, da sich mit Pyhton große Datenmengen vergleichweise einfach aufnehmen, verarbeiten und ausgeben lassen. Große Datenmengen sind für die KI-Entwicklung, häufig z.B. als Trainingsdatensätze, unverzichtbar.

Entwürfe und Testentwicklungen von KI-Anwendungen erfolgen insbesondere in frühen Phasen meist lokal in einer Entwicklungsumgebung. Ergebnisse zu präsentieren oder ein Austausch mit Dritten darüber ist damit nicht immer einfach möglich. Zumeist müssten Dritte erst einen Python-Interpreter sowie umfangreiche weitere Äbhänigkeiten, wie z.B. das verwendete KI-Framework selbst, installieren um die KI-Anwendung überhaupt erst Ausführen zu können. Das fällt nicht jedem interessieren leicht. Auch gibt es, durch die dynmaische Entwicklung, meist verscheidenste Versionen der Software und Komponenten die zu unbeabsichtigen und unerwarteten Problemen und Hürden führen können.

[1] Representational State Transfer, API-Architekturansatz, konzipiert und entwickelt für Webservices
[2] Application Programming Interface, Programmierschnittstelle

1.2 Methodik

Die Seminararbeit soll das gewünschte Ziel, durch exploratives, vertikalens Prototyping erreichen. Vertikal bedeutet dabei, dass von der einfachen Konsolen KI-Anwendung bis hin zum dialogbasierten Aufruf über den Webserver, alle Ebenen durchdrungen werden sollen. Dabei sollen alle Schritte, jede eingesetzte Software und Lösung gründlich betrachtet und alle Schritte und Entscheidungen nachvollziehbar dokumentiert werden.

Zuerst soll eine einfache KI-Anwendung, die als Konsolenanwendung lokal in einer Entwicklungsumgebung, lauffähig ist erstellt werden. Hier ist ein üblicher Prototyp einer KI-Anwendung, z.B. aus einem einfach Tutorial, aus. Zum erreichen des Ziels, reicht es dabei aus, wenn nur wenige einfache Funktionen des verwendeten KI-Frameworks zum Einsatz kommen.

Im Anschluss daran, soll ermittelt werden auf welche Arten die Einbindung einer solchen KI-Anwendung in das Django-Framework möglich ist. Mögliche Lösungsansätze sollen ermittelt, betrachtet und bewertet werden.

Die schrittweise Einrichtung einer Django-Installation wird analog durchgeführt und alle gemachten Einstellungen, eingerichteten Erweiterungen und Anpssungen nachvollziehbar dokumentiert. Dies umfasst dabei auch die notwendigen Module für Bereitstellung der API sowie den Einsatz von Django in einem produktiven Umfeld.

Im Schlussschritt werden die Django-Installation und die, für die Einbindung vorbereitete, KI-Anwendung zusammengeführt. Der dann mögliche, dialogbasierte Zugriff auf die KI-Anwendung als Webservice, die Django REST API, soll durch den Einsatz eines dynamischen Frontends, belegt werden.

2 Theorie: Begrifflichkeiten und Grundlagen

In diesem Abschnitt werden einige für das weieter Verständnis hilfreiche Begrifflichkeiten und Grundlagen kurz erklärt und eingeordnet. Dabei wird als Zielgruppe der Seminararbeit, eine zumindest in IT-Grundlagen versierte, Leserschaft angenommen.

2.1 REST API

Die Abkürzung API bedeutet Application Programming Interface, ins deutsche übersetzt also „Schnitstelle zur Programmierung von Anwendungen“. Eine API ist damit eine Schnittstelle zu einem Programm oder zu einer Programmierung und erlaubt es, meist anhand detailierten Dokumentation, den Zugriff auf die bzw. die Verarbeitung von Daten aus der Schnittstelle. Im heutigen Sprachgebrauch ist mit API meist der Zugriff auf einen Webservice gemeint. Grundsätzlich aber, umfasst der Begriff API nicht nur die vorgenantnen Webservices, die eine Protokollorientierte Schnittstelle sind, sondern auch API anderer Typenklassen wie Datei-, Funktions- oder Objektorientierte APIs.

Mit Representational State Transfer (REST) beschreibt Fielding erstmals im Jahr 2000 in seiner Arbeit *Architectural styles and the design of network-based software architectures* einen hybriden Architekturansatz aus der Vermischung verschiedener netzwerkbasierter Architekturansätze für Schnittstellen. Dieser Ansatz hat sich zwischenzeitlich durchgesetzt und wird vielfach in modernen Webservices verwendet. Fielding beschreibt in seiner Arbeit umfassend wie eine zustandslose Kommunikation zwischen Client und Server über eine einheitliche Schnittstelle (API) mittels bekannter und bewährter Hypertext Transfer Protocol (Secure) (HTTP/S)-Methoden, wie GET, POST, PUT und DELETE, umgesetzt wird.

Heute werden REST APIs vielfälig eingesetzt: Ob zum Datenaustausch zwischen Unternehmen oder auch der Realisierung verschiedener Frontends bzw. Endgeräte für eine Anwendung – Aus der Praxis sind Schnittstellen dieser Art heute nicht mehr wegzudenken.

Anhang

Anhang 1: Beispielanhang

Dieser Abschnitt dient nur dazu zu demonstrieren, wie ein Anhang aufgebaut seien kann.

Anhang 1.1: Weitere Gliederungsebene

Auch eine zweite Gliederungsebene ist möglich.

Literaturverzeichnis

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. Irvine: University of California.

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit von mir selbstständig und ohne unerlaubte Hilfe angefertigt worden ist, insbesondere dass ich alle Stellen, die wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen sind, durch Zitate als solche gekennzeichnet habe. Ich versichere auch, dass die von mir eingereichte schriftliche Version mit der digitalen Version übereinstimmt. Weiterhin erkläre ich, dass die Arbeit in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde/Prüfungsstelle vorgelegen hat. Ich erkläre mich damit nicht einverstanden, dass die Arbeit der Öffentlichkeit zugänglich gemacht wird. Ich erkläre mich damit einverstanden, dass die Digitalversion dieser Arbeit zwecks Plagiatsprüfung auf die Server externer Anbieter hochgeladen werden darf. Die Plagiatsprüfung stellt keine Zurverfügungstellung für die Öffentlichkeit dar.

Wilnsdorf, 27. August 2022
(Ort, Datum)

(Eigenhändige Unterschrift)