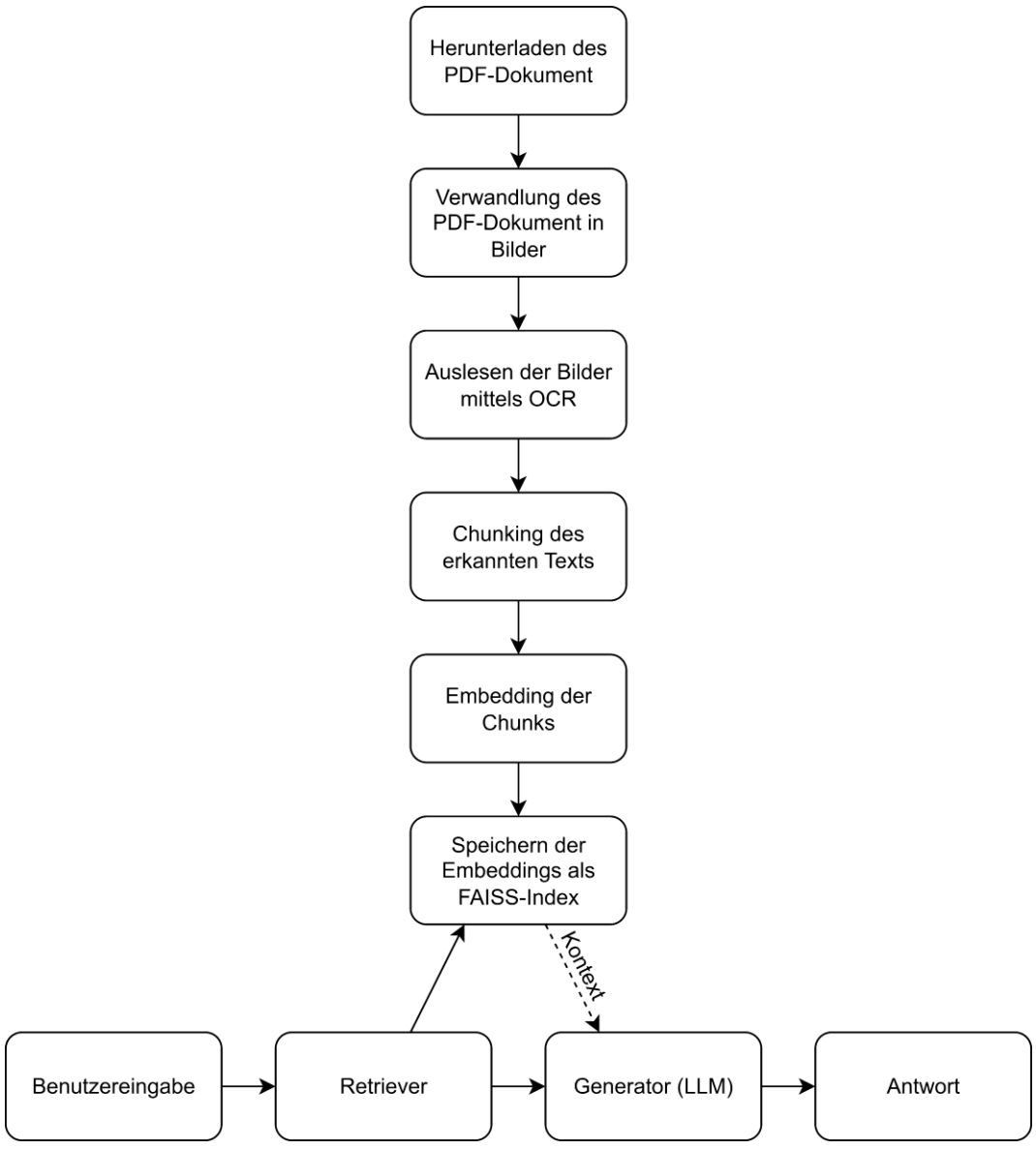
Dokumentation - RAG-System

Architektur:

Zunächst wird ein PDF-Dokument anhand einer URL heruntergeladen, in Bilder verwandelt und mit OCR eingelesen. Der durch OCR extrahierte Text wird anschließend in Token basierte Chunks zerteilt und über das OpenAI Modell text-embedding-3-small vektorisiert. Die daraus resultierenden Embeddings werden in einem FAISS-Index gespeichert. Die gesamte Prompt-Chain wurde dabei mit LangChain umgesetzt. Sobald eine Benutzerfrage eingeht, wird diese in den Prompt eingebettet, mit dem FAISS-Index abgeglichen, welcher über den Retriever die relevantesten Chunks aus dem FAISS-Index extrahiert und in den Prompt einbaut. Anschließend wird der Prompt an das gpt-4o-mini LLM von OpenAI gegeben, welches basierend auf dem Kontext, der Frage und dem bereitgestellten Template eine Antwort liefert, die durch den Parser als String ausgegeben wird. Im Folgenden ist die Architektur nochmals grafisch dargestellt.



Modellkonfiguration:

Für die Antwortgenerierung wird das Large Language Modell **gpt-4o-mini** von OpenAI mit einer **Temperatur** von **0.3** verwendet, da das LLM sachlich und fundiert antworten soll. Zur Vektorisierung kommt das Modell **text-embedding-3-small** von OpenAI zum Einsatz. Die **Chunkgröße** ist auf **300 Tokens** und der **Overlap** auf **50 Tokens** begrenzt. Dies dient dazu, die Informationslücke zu schließen und die Chunks möglichst klein zu halten. Die **LLM-Antwort** ist auf **300 Tokens** beschränkt, um unkontrolliert große Antworten zu vermeiden und somit Kosten zu sparen.

Besonderheiten bei der Vorverarbeitung:

Bei der Vorbereitung gibt es aufgrund des Einsatzes von OCR keine besonderen Bedingungen. Durch den Einsatz von OCR können auch gescannte Dokumente, die im Bildformat ohne Textinformationen vorliegen, für den Kontext des RAG-Systems verwendet werden. Es muss einzig und allein sichergestellt werden, dass das PDF-Dokument über die angegebene URL abgerufen werden kann.