|  |  |
| --- | --- |
| Knowledge BAse assistent  7.11.2023 | Abstract  Řešení RestAPI služby, která implementuje funkcionalitu pro potřebu chatbotů založených na externích zdrojích dat. Služba dokáže podporovat více chatbotů založených na různých zdrojích dat. Součástí řešení je i příprava externích dat pro potřeby této služby.  Vladimír Dědourek, Michal Stoklasa |

Obsah

[1 Knowledge Base Assistent 2](#_Toc150347085)

[1.1 Cíle řešení 2](#_Toc150347086)

[2 KBAIndex 2](#_Toc150347087)

[2.1 Model embeddings 3](#_Toc150347088)

[2.1.1 Indexování strukturovaného textu 3](#_Toc150347089)

[3 KBAQnA 3](#_Toc150347090)

[4 RestAPI server 4](#_Toc150347091)

[4.1 Dotaz odpověď – qna 5](#_Toc150347092)

[4.2 Načtení parametrů serveru – get\_srv\_par 6](#_Toc150347093)

[4.3 Načtení parametrů projektu – get\_project\_par 6](#_Toc150347094)

[4.4 Nastavení parametrů serveru – set\_srv\_par 7](#_Toc150347095)

[4.5 Nastavení parameter projektu – set\_project\_par 7](#_Toc150347096)

[5 Přílohy 8](#_Toc150347097)

# Knowledge Base Assistent

## Cíle řešení

Cílem řešení je vytvořit framework pro tvorbu webových chatbotů založených na externích zdrojíchy dat. Těmito zdroji mou být jak různé typy textových souborů, tak i webové stránky samotné.

Řešení jse složeno ze 2 částí:

1 – Úoha pro indexování z externích zdrojů dat (KBAIndex)

2 – Rest API služba která vytváří odpovědi na základě položených dotazů. Umožňuje vést konverzaci (KBAQnA)

# KBAIndex

Úloha pro indexování externích zdrojů dat a uložení do vektorové databáze pro potřeby Rest API služby. Funkcionalita je implementovaná ve třídě Pythonu KBAIndex. Funkcionalita je založena na modelu vytvoření embeddings, které jsou uloženy v Qdrant databázi.

## Model embeddings



### Indexování strukturovaného textu

Pro indexování se berou texty z externích zdrojůt. Tyto texty jsou rozděleny na menší segmenty (chunks) tak, aby maximální velikost nepřekročila maximální počet tokenu (max\_tokens).

# KBAQnA

Třída Pythonu, která zpřístupňuje služby pro generování odpovědi na základě dotazů.

# RestAPI server

Rest API služba ve formě serveru, která zpřístupňuje služby pro generování odpovědi na základě dotazů. Využívá třídu KBAQnA



Programový kód aplikace je veden v projektu PythonFlaskRestApi. Je synchronizován do Git úložiště na GitHubu. Při změně v Git repository jsou změny automaticky sestaveny a publikovány do webové služby v Azure.

A screenshot of a chatbot

Description automatically generated

Jednotlivé služby jsou popsány v následující kapitole.

Služby jsou zabezpečeny Basic autorizací (username, password).

## Dotaz odpověď – qna

/qna - Question / Answer service. It cooperates with class KBAQnA. Use embeddings in vector database Qdrant, where are prepared embeddings data for the project.

POST method.

Input:

{

"question": question,

"user\_id": user\_id,

"project": project,

[system\_msg": system\_msg,]

["api\_type": api\_type,]

["api\_base": api\_base,]

["api\_key": api\_key,]

["api\_version": api\_version,]

["api\_model": api\_model,]

}

Output:

{

"answer": answer

"error": error

}

Parameters:

question - question (is mandatory)

user\_id - unique user id (is mandatory)

project - project name (is collection name in vector db). Is mandatory.

system\_msg - partial text which will be added at the begin of the system message (can be empty)

api\_type - OpenAI type - open\_ai, azure (if empty then environment variable "OPENAI\_API\_TYPE" is used )

api\_base - URL base of the ChatGPT API (if empty then environment variable "OPENAI\_API\_BASE" is used

api\_key - API key of the ChatGPT (if empty then environment variable "OPENAI\_API\_KEY" is used)

api\_version - version of the ChatGPT API (if empty then environment variable "OPENAI\_API\_VERSION" is used)

api\_model - model of the ChatGPT API. (if empty then environment variable "OPENAI\_API\_MODEL\_GPT" is used)

For open\_ai: gpt-3.5-turbo, gpt-3.5-turbo-0613, gpt-3.5-turbo-16k, gpt-3.5-turbo-16k-0613

gpt-4, gpt-4-0613, gpt-4-32k, gpt-4-32k-0613

For azure: deployment name

answer - answer

error - normally it is empty. It contains a text error if there is a problem

## Načtení parametrů serveru – get\_srv\_par

/get\_srv\_par - Get server parameters

GET method.

Output:

{

"db\_type": db\_type,

"db\_dir": db\_dir,

"system\_msg": system\_msg,

"k\_history": k\_history,

"time\_limit\_history": time\_limit\_history,

"verbose": verbose,

"answer\_time": answer\_time,

}

Parameters:

db\_type - vector database type (if empty then unchanged):

local - local Chroma DB in db directory,

qdrant - Qdrant database.

db\_dir - directory, where is saved local vector Chroma db (only for db = local)

system\_msg - partial text which will be added at the begin of the system message (can be empty)

k\_history - the maximum length of history that is used for the conversation

time\_limit\_history - the time interval in seconds after which the history is erased

verbose - True - logging process question/answer to system output, False - without logging

answer\_time - True - the answer contains the time spent in seconds, False - answer is without spent time

## Načtení parametrů projektu – get\_project\_par

/get\_project\_par - Get project parameters

GET method.

Output:

{

"project": project,

"system\_msg": system\_msg,

"api\_model": api\_model,

"answer\_time": answer\_time,

}

Parameters:

project - project name. Is mandatory.

Parameters:

system\_msg - partial text which will be added at the begin of the system message

api\_model - model of the ChatGPT API.

For open\_ai: gpt-3.5-turbo, gpt-3.5-turbo-0613, gpt-3.5-turbo-16k, gpt-3.5-turbo-16k-0613

gpt-4, gpt-4-0613, gpt-4-32k, gpt-4-32k-0613

For azure: deployment name

answer\_time - True - answer is with elapsed time, False - answer is without elapsed time

## Nastavení parametrů serveru – set\_srv\_par

/set\_srv\_par - Set server parameters

POST method.

{

["db\_type": db\_type,]

["db\_dir": db\_dir,]

["system\_msg": system\_msg,]

["k\_history": k\_history,]

["time\_limit\_history": time\_limit\_history,]

["verbose": verbose,]

["answer\_time": answer\_time,]

["erase\_history": erase\_history]

}

Parameters (if parameter isn't used then is unchanged):

db\_type - vector database type (if empty then unchanged):

local - local Chroma DB in db directory,

qdrant - Qdrant database. Needs environment variables: QDRANT\_URL, QDRANT\_API\_KEY

db\_dir - directory, where is saved local vector Chroma db (only for db = local)

system\_msg - partial text which will be added at the begin of the system message ((if empty then it is unchanged))

k\_history - the maximum length of history that is used for the conversation

time\_limit\_history - the time interval in seconds after which the history is cleared

verbose - True - logging process question/answer to system output, False - without logging

answer\_time - True - the answer contains the time spent in seconds, False - answer is without spent time

Others:

erase\_history - True - question/answer history will be erased, False - question/answer history will not be erased

Default False.

## Nastavení parameter projektu – set\_project\_par

/set\_project\_par - Set project parameters

POST method.

{

"project": project,

["system\_msg": system\_msg,]

["api\_model": api\_model,]

["answer\_time": answer\_time,]

["erase\_history": erase\_history]

}

Parameters: (when is empty or None then are unchanged)

project - project name (is collection name in vector db). Is mandatory.

system\_msg - partial text which will be added at the begin of the system message (if is empty then is unchanged)

api\_model - model of the ChatGPT API. (if empty then environment variable "OPENAI\_API\_MODEL\_GPT" is used)

For open\_ai: gpt-3.5-turbo, gpt-3.5-turbo-0613, gpt-3.5-turbo-16k, gpt-3.5-turbo-16k-0613

gpt-4, gpt-4-0613, gpt-4-32k, gpt-4-32k-0613

For azure: deployment name

answer\_time - True - answer is with elapsed time, False - answer is without elapsed time (if is None or isn't presented then is unchanged)

# Přílohy