Assignment 4: Retrieval-Augmented Generation with LangChain

2025 NTHU Natural Language Processing

Hung-Yu Kao

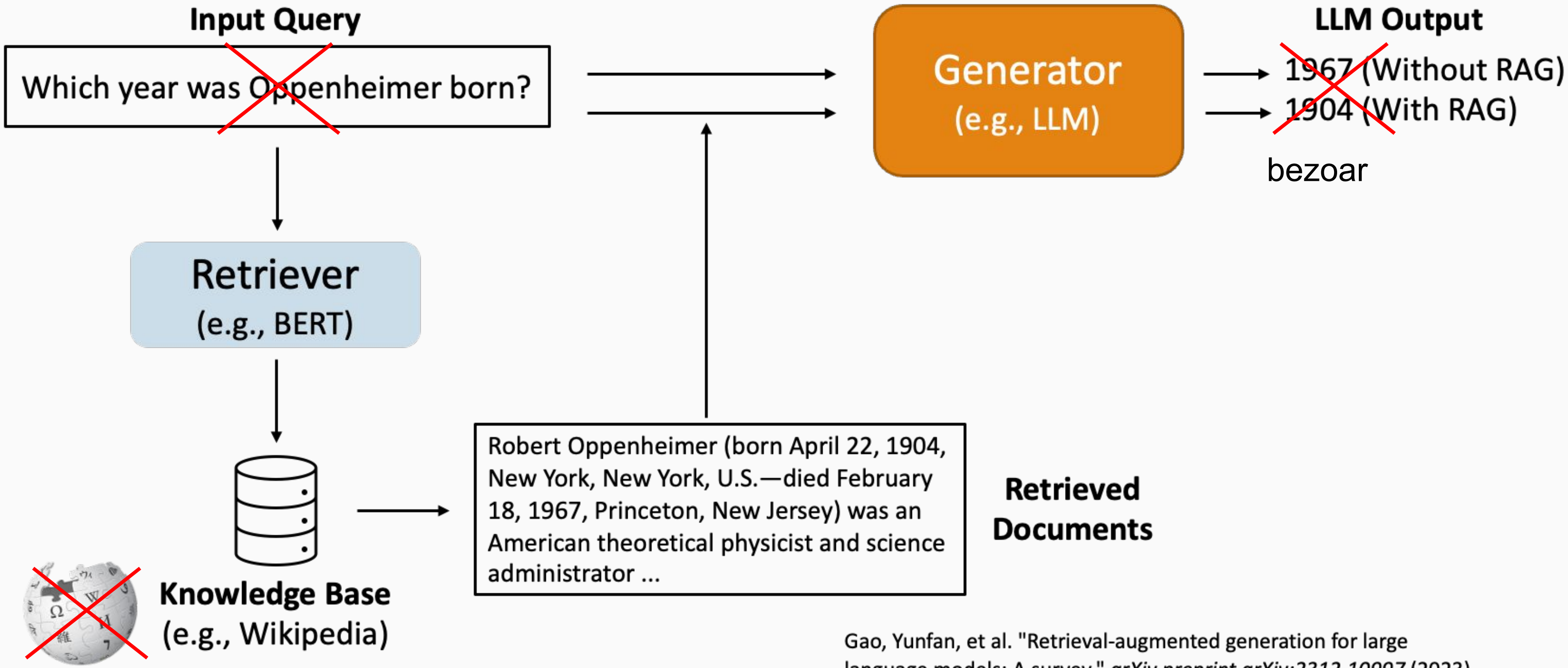
IKM Lab TAs

Assignment Description

- Task: Retrieval-Augmented Generation (RAG) for **Question Answering**
- Database: Cats’ knowledge in texts
- Inputs: **Ten questions** about cats’ knowledge
- Answer: Short answers for the ten questions (evaluation: exact matching)
- Restricted models
    - Generator: Llama-3.2-1B (freezed)

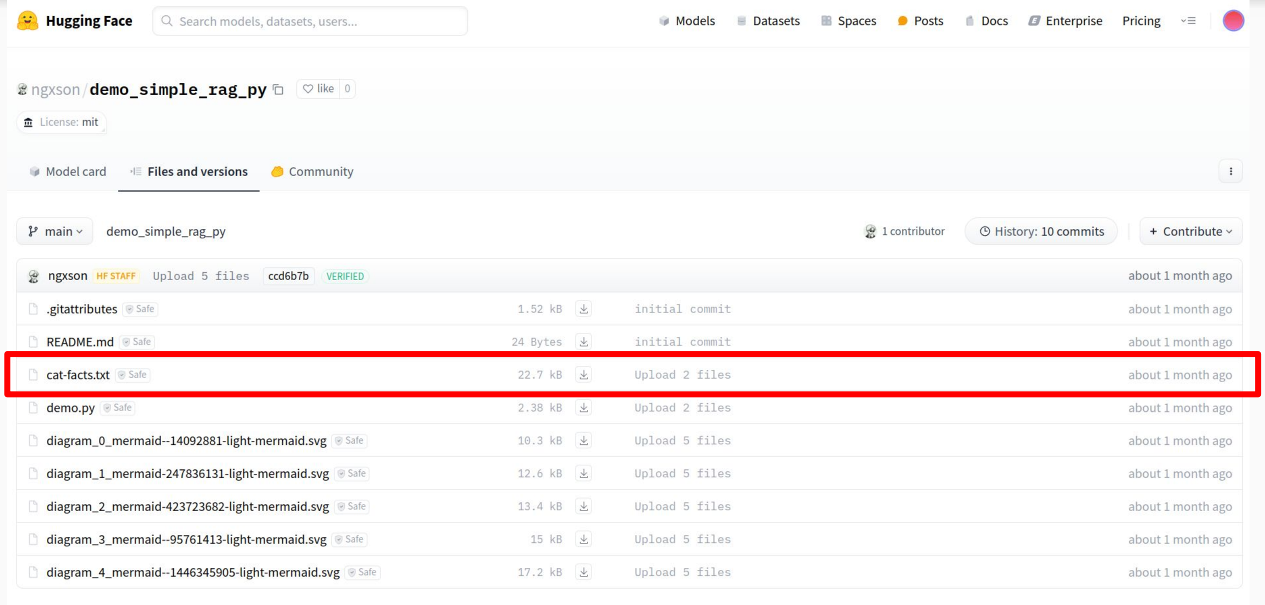
Retrieval-Augmented Generation (RAG)

🐈What is the technical term for a cat's hairball?


Input Query
~~Which year was Oppenheimer born?~~
Retriever
(e.g., BERT)
Knowledge Base
(e.g., Wikipedia)
Robert Oppenheimer (born April 22, 1904, New York, New York, U.S.—died February 18, 1967, Princeton, New Jersey) was an American theoretical physicist and science administrator ...
Retrieved Documents
Generator
(e.g., LLM)
LLM Output
~~1967 (Without RAG)~~
~~1904 (With RAG)~~
bezoar

Gao, Yunfan, et al. "Retrieval-augmented generation for large language models: A survey." *arXiv preprint arXiv:2312.10997* (2023).

Dataset: Cat-facts



ngxson/demo_simple_rag_py

License: mit

demo_simple_rag_py

ngxson HF STAFF Upload 5 files ccd6b7b VERIFIED about 1 month ago

File	Size	Commit	Date
.gitattributes Safe	1.52 kB	initial commit	about 1 month ago
README.md Safe	24 Bytes	initial commit	about 1 month ago
cat-facts.txt Safe	22.7 kB	Upload 2 files	about 1 month ago
demo.py Safe	2.38 kB	Upload 2 files	about 1 month ago
diagram_0_mermaid--14092881-light-mermaid.svg Safe	10.3 kB	Upload 5 files	about 1 month ago
diagram_1_mermaid-247836131-light-mermaid.svg Safe	12.6 kB	Upload 5 files	about 1 month ago
diagram_2_mermaid-423723682-light-mermaid.svg Safe	13.4 kB	Upload 5 files	about 1 month ago
diagram_3_mermaid--95761413-light-mermaid.svg Safe	15 kB	Upload 5 files	about 1 month ago
diagram_4_mermaid--1446345905-light-mermaid.svg Safe	17.2 kB	Upload 5 files	about 1 month ago

The dataset is presented in ngxson/demo_simple_rag_py

Dataset

Cat-facts dataset

- Database size: 150 facts
- Test QA: 150 QA pairs (generated by GPT-5)
- Each fact is represented as a sentence.
    - E.g., When a cat chases its prey, it keeps its head level. Dogs and humans bob their heads up and down.
    - E.g., The technical term for a cat's hairball is a "bezoar."
    - …

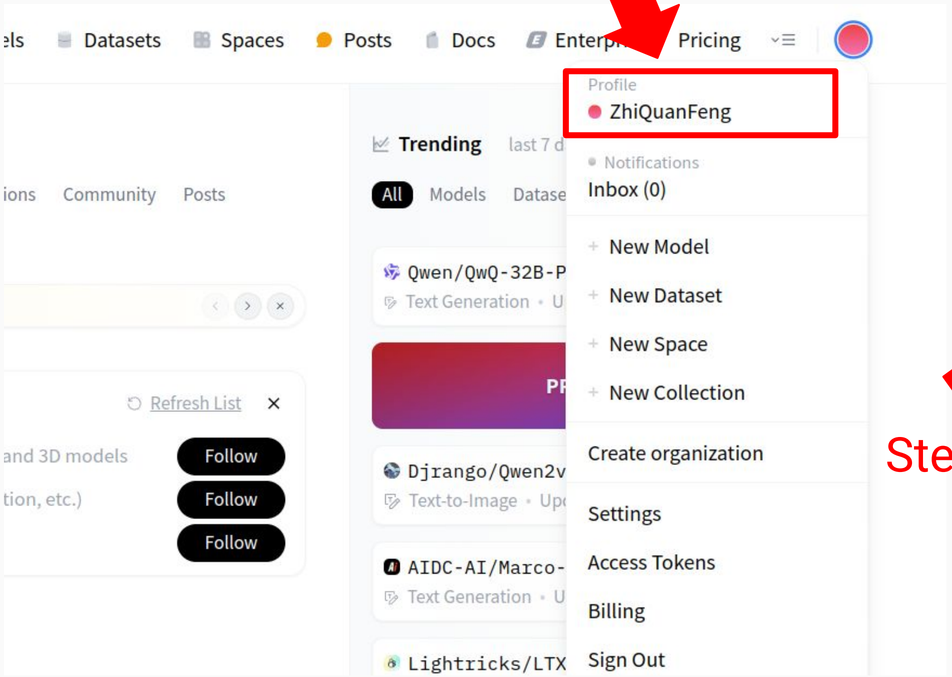
Construction for Questions and Answers

Index	Question	Answer Sentence in the database	Answer we set
1	Why don't cats have a sweet tooth?	Unlike dogs, cats do not have a sweet tooth. Scientists believe this is due to a mutation in a key taste receptor.	Taste mutation
2	How many sounds do cats make?	Cats make about 100 different sounds. Dogs make only about 10.	About 100
3	How many domestic cats exist worldwide?	There are more than 500 million domestic cats in the world, with approximately 40 recognized breeds.	500 million
4	How many people are bitten by cats annually in the U.S.?	Approximately 40,000 people are bitten by cats in the U.S. annually.	40,000
5	What is a cat's top running speed?	A cat can travel at a top speed of approximately 31 mph (49 km) over a short distance.	31 mph

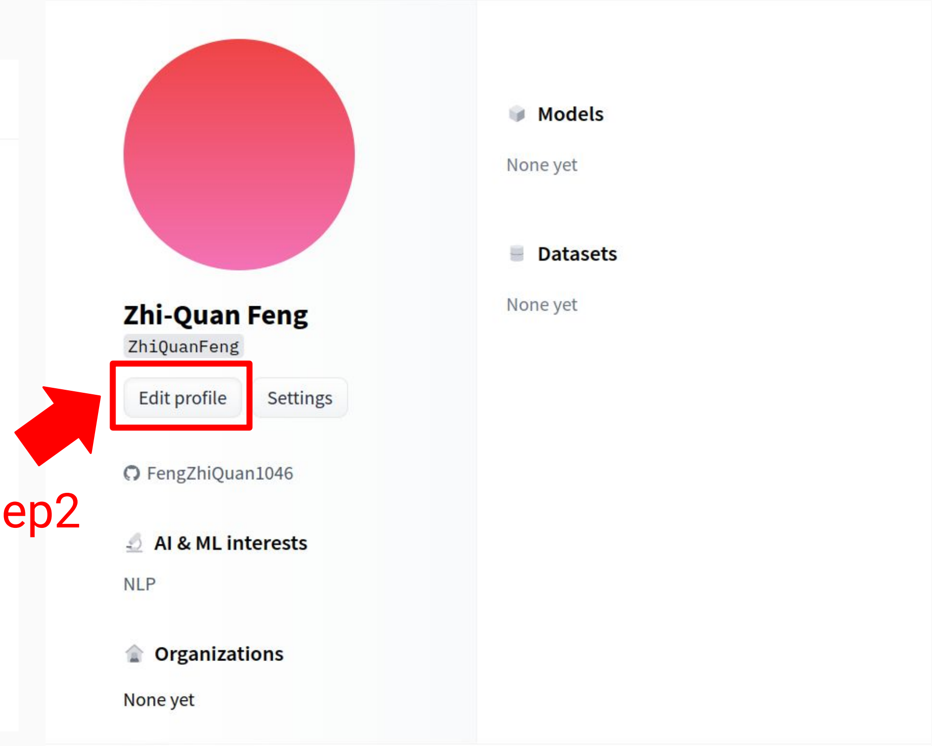
Code (hints only)

Connect to Huggingface (1/4)

Step1



Step2



Zhi-Quan Feng
ZhiQuanFeng

FengZhiQuan1046

AI & ML interests

NLP

Organizations

None yet

Models

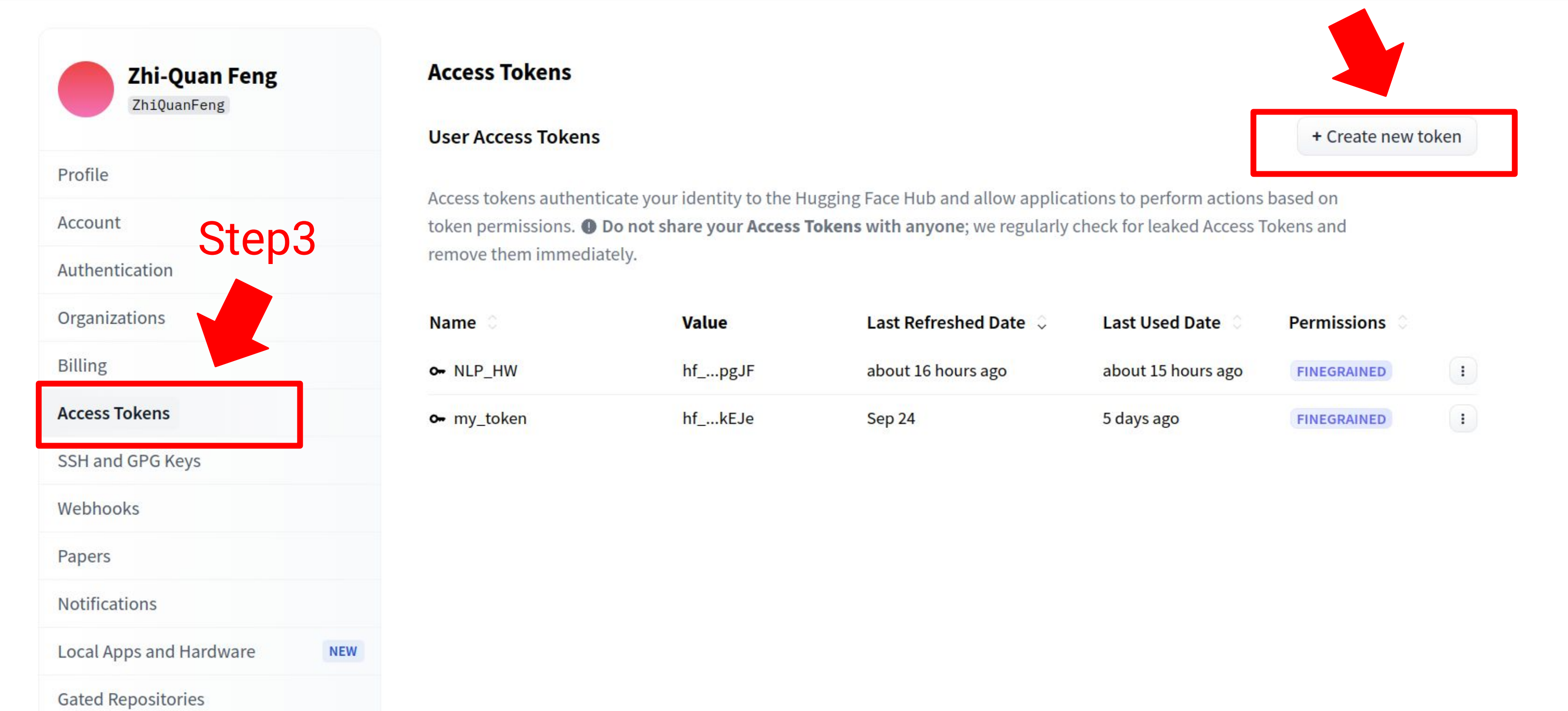
None yet

Datasets

None yet

Connect to Huggingface (2/4)

Step4



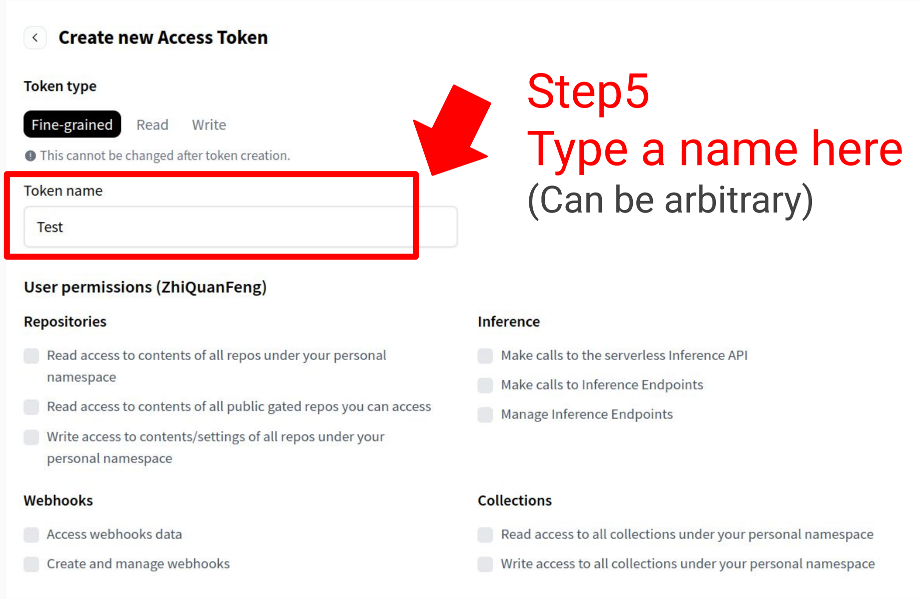
Access Tokens

User Access Tokens

Access tokens authenticate your identity to the Hugging Face Hub and allow applications to perform actions based on token permissions. **Do not share your Access Tokens with anyone**; we regularly check for leaked Access Tokens and remove them immediately.

Name	Value	Last Refreshed Date	Last Used Date	Permissions
NLP_HW	hf_...pgJF	about 16 hours ago	about 15 hours ago	FINEGRAINED
my_token	hf_...kEJe	Sep 24	5 days ago	FINEGRAINED

Connect to Huggingface (3/4)



Create new Access Token

Token type

Fine-grained Read Write

This cannot be changed after token creation.

Token name

Test

User permissions (ZhiQuanFeng)

Repositories

- ☐ Read access to contents of all repos under your personal namespace
- ☐ Read access to contents of all public gated repos you can access
- ☐ Write access to contents/settings of all repos under your personal namespace

Inference

- ☐ Make calls to the serverless Inference API
- ☐ Make calls to Inference Endpoints
- ☐ Manage Inference Endpoints

Webhooks

- ☐ Access webhooks data
- ☐ Create and manage webhooks

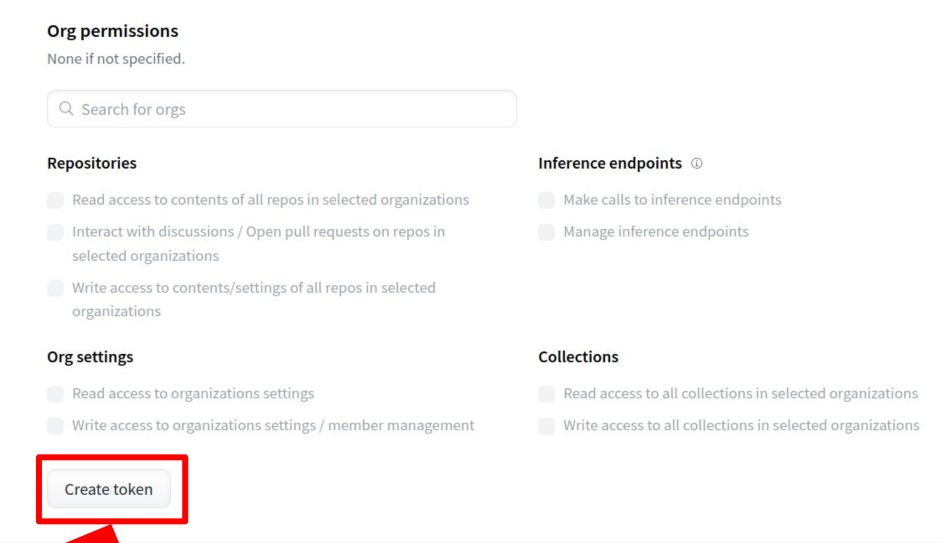
Collections

- ☐ Read access to all collections under your personal namespace
- ☐ Write access to all collections under your personal namespace

Step5

Type a name here

(Can be arbitrary)



Org permissions

None if not specified.

Search for orgs

Repositories

- ☐ Read access to contents of all repos in selected organizations
- ☐ Interact with discussions / Open pull requests on repos in selected organizations
- ☐ Write access to contents/settings of all repos in selected organizations

Inference endpoints

- ☐ Make calls to inference endpoints
- ☐ Manage inference endpoints

Org settings

- ☐ Read access to organizations settings
- ☐ Write access to organizations settings / member management

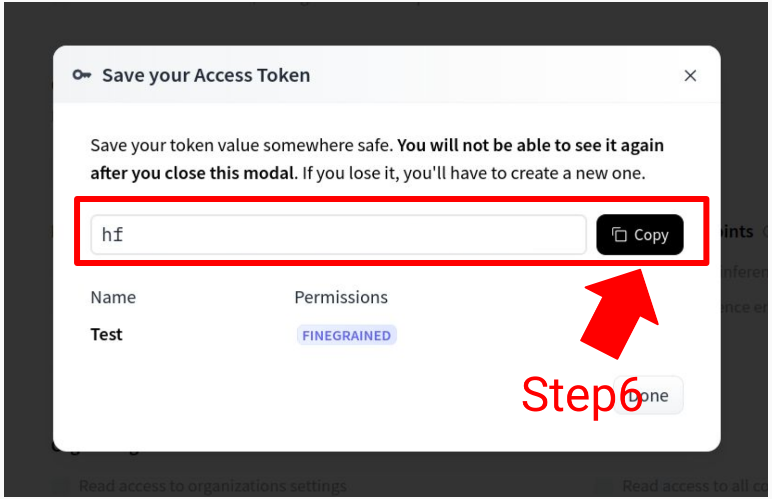
Collections

- ☐ Read access to all collections in selected organizations
- ☐ Write access to all collections in selected organizations

Create token

Step6

Connect to Huggingface (4/4)



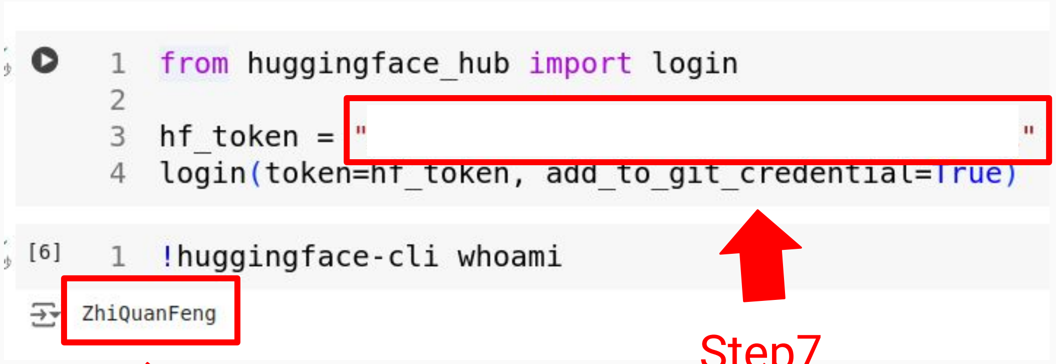
Save your Access Token

Save your token value somewhere safe. **You will not be able to see it again after you close this modal.** If you lose it, you'll have to create a new one.

hf

Name	Permissions
Test	FINEGRAINED

Step6



```
from huggingface_hub import login

hf_token = ""
login(token=hf_token, add_to_git_credential=True)
```

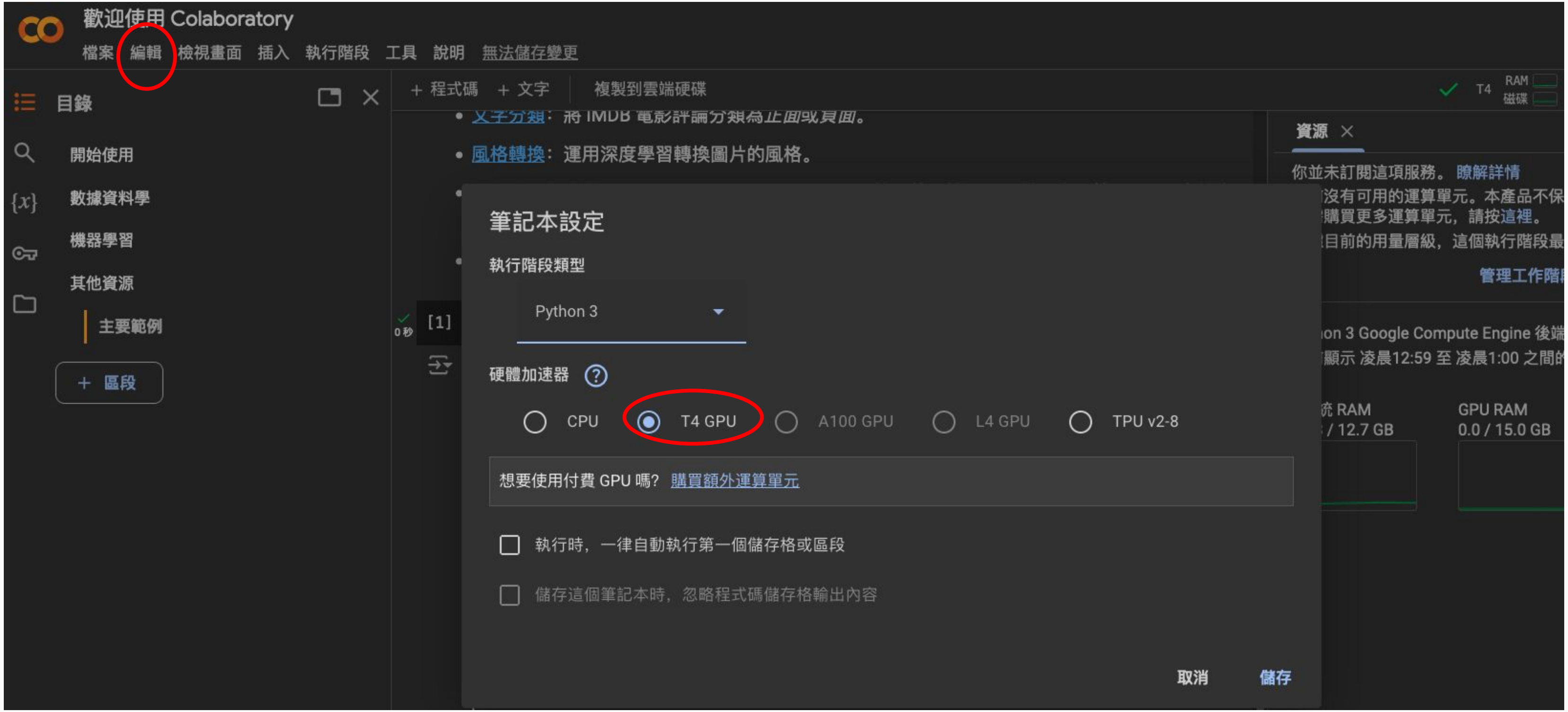
```
[6] !huggingface-cli whoami
```

```
ZhiQuanFeng
```

Step7
Your access token here

If you succeed, your Hugging Face ID will be displayed here.

How to use GPU on Colab?



- 文字分類：將 IMDB 電影評論分類為正面或負面。
- 風格轉換：運用深度學習轉換圖片的風格。

你並未訂閱這項服務。瞭解詳情

沒有可用的運算單元。本產品不保

購買更多運算單元，請按這裡。

目前的用量層級，這個執行階段最

管理工作階

on 3 Google Compute Engine 後端

顯示 凌晨12:59 至 凌晨1:00 之間的

統 RAM / 12.7 GB

GPU RAM 0.0 / 15.0 GB

筆記本設定

執行階段類型

Python 3

硬體加速器 ?

○ CPU ● T4 GPU ○ A100 GPU ○ L4 GPU ○ TPU v2-8

想要使用付費 GPU 嗎? 購買額外運算單元

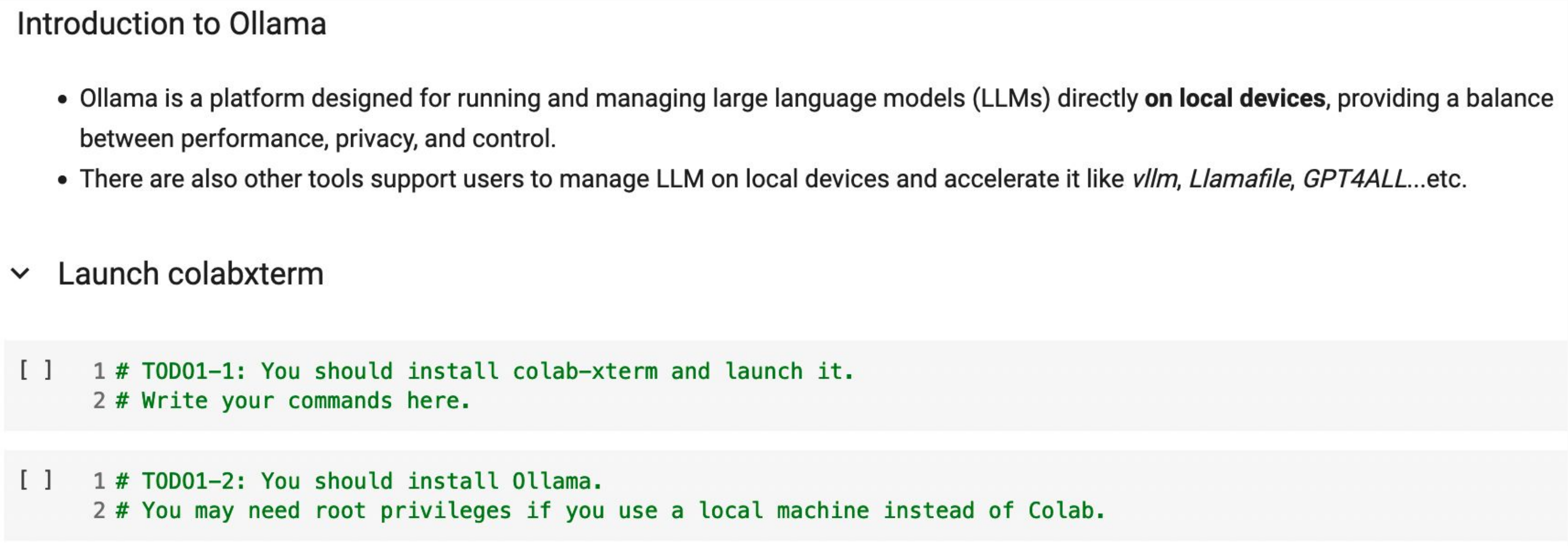
□ 執行時，一律自動執行第一個儲存格或區段

□ 儲存這個筆記本時，忽略程式碼儲存格輸出內容

Type in a code block:

```
!nvidia-smi
```

TODO1: Set up the environment of Ollama (1/3)



Introduction to Ollama

- Ollama is a platform designed for running and managing large language models (LLMs) directly **on local devices**, providing a balance between performance, privacy, and control.
- There are also other tools support users to manage LLM on local devices and accelerate it like *vllm*, *Llamafile*, *GPT4ALL*...etc.

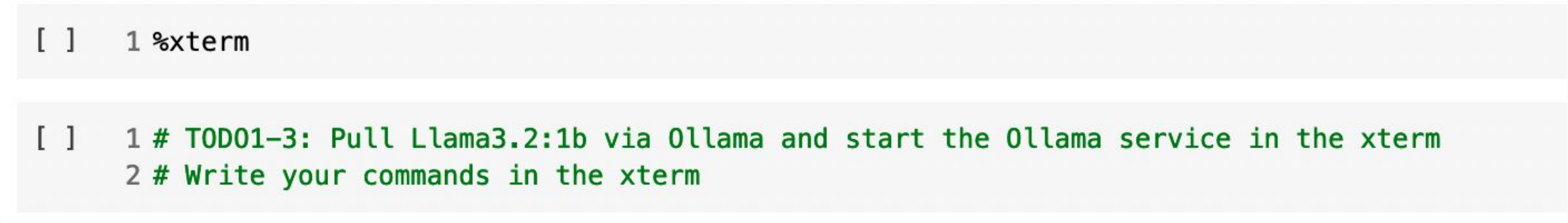
Launch colabxterm

```
# TODO1-1: You should install colab-xterm and launch it.
# Write your commands here.
```

```
# TODO1-2: You should install Ollama.
# You may need root privileges if you use a local machine instead of Colab.
```

Install the colab-xterm package and use “load_ext” to use it.

TODO1: Set up the environment of Ollama (2/3)



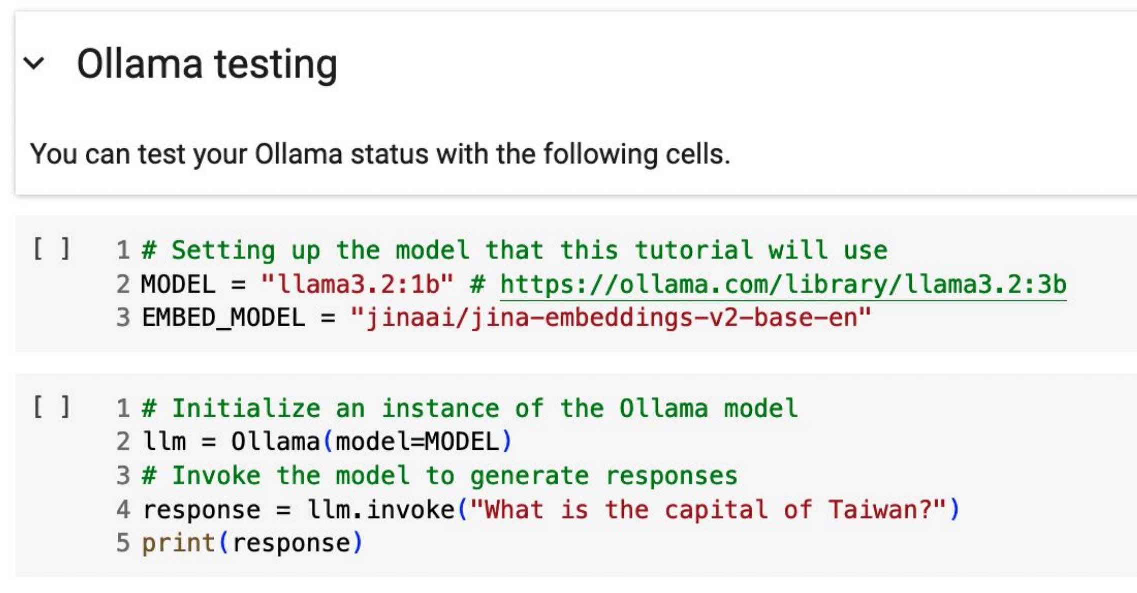
```
%xterm
```

```
# TODO1-3: Pull Llama3.2:1b via Ollama and start the Ollama service in the xterm
# Write your commands in the xterm
```

Use “%xterm” and the followint command to the command line:

- curl -fsSL https://ollama.com/install.sh | sh
- Download Llama3.2-1b using pull command in Ollama
- ollama serve

A simple test of Ollama



Ollama testing

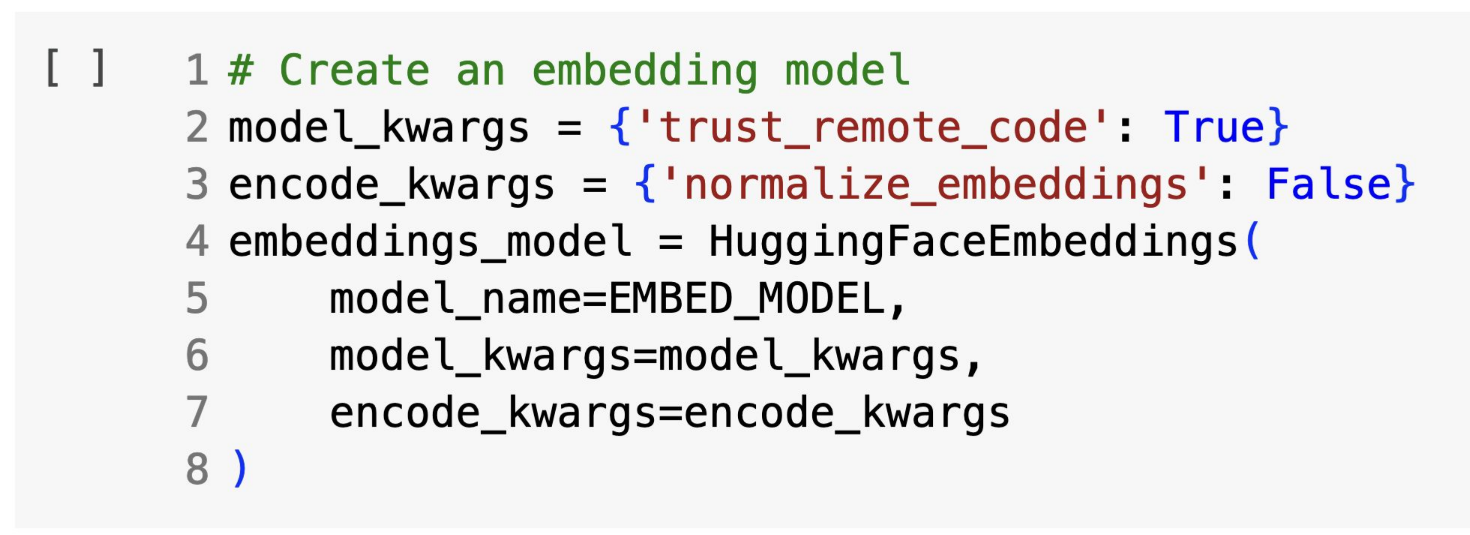
You can test your Ollama status with the following cells.

```python
# Setting up the model that this tutorial will use
MODEL = "llama3.2:1b" # https://ollama.com/library/llama3.2:3b
EMBED_MODEL = "jinaai/jina-embeddings-v2-base-en"
```

```python
# Initialize an instance of the Ollama model
llm = Ollama(model=MODEL)
# Invoke the model to generate responses
response = llm.invoke("What is the capital of Taiwan?")
print(response)
```

- Here we define the retrieval model and document reader model.
- We can perform a simple test for checking Ollama’s status.
- The .invoke API is to input sequence to the model and obtain the outputs.

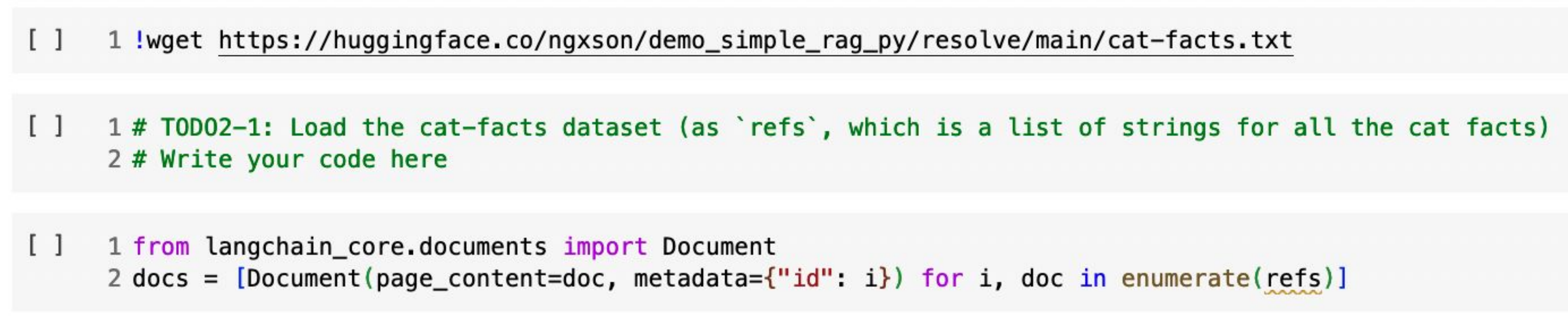
Define the retriever model



```python
# Create an embedding model
model_kwargs = {'trust_remote_code': True}
encode_kwargs = {'normalize_embeddings': False}
embeddings_model = HuggingFaceEmbeddings(
    model_name=EMBED_MODEL,
    model_kwargs=model_kwargs,
    encode_kwargs=encode_kwargs
)
```

Build a simple document retriever using LangChain (HuggingFaceEmbeddings).

TODO2: Load the cat-facts dataset and prepare the retrieval database



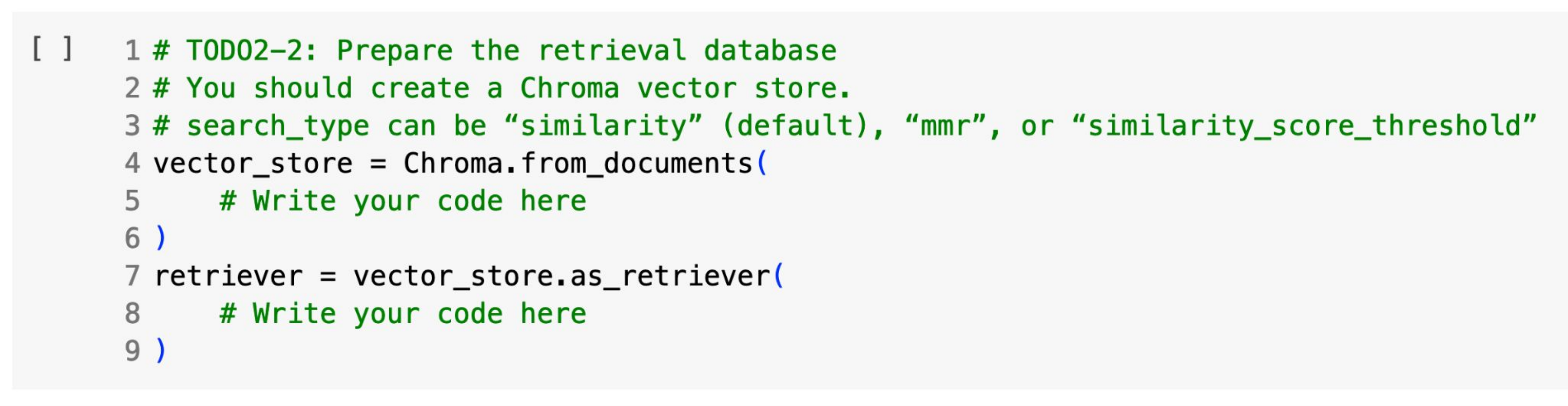
```
!wget https://huggingface.co/ngxson/demo_simple_rag_py/resolve/main/cat-facts.txt
```

```
# TODO2-1: Load the cat-facts dataset (as `refs`, which is a list of strings for all the cat facts)
# Write your code here
```

```
from langchain_core.documents import Document
docs = [Document(page_content=doc, metadata={"id": i}) for i, doc in enumerate(refs)]
```

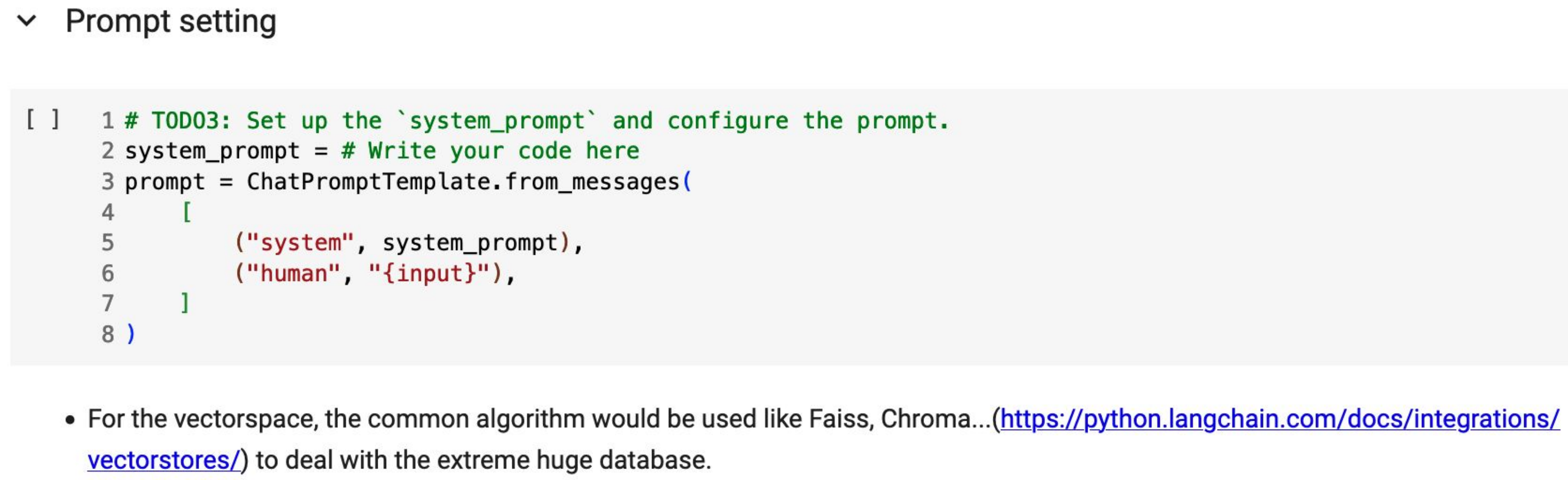
Load the data to be retrieved and use the langchain.docstore.document.Document API to construct a database.

TODO2: Load the cat-facts dataset and prepare the retrieval database



```
# TODO2-2: Prepare the retrieval database
# You should create a Chroma vector store.
# search_type can be “similarity” (default), “mmr”, or “similarity_score_threshold”
vector_store = Chroma.from_documents(
    # Write your code here
)
retriever = vector_store.as_retriever(
    # Write your code here
)
```

TODO3: Set up the `system_prompt` and configure the prompt



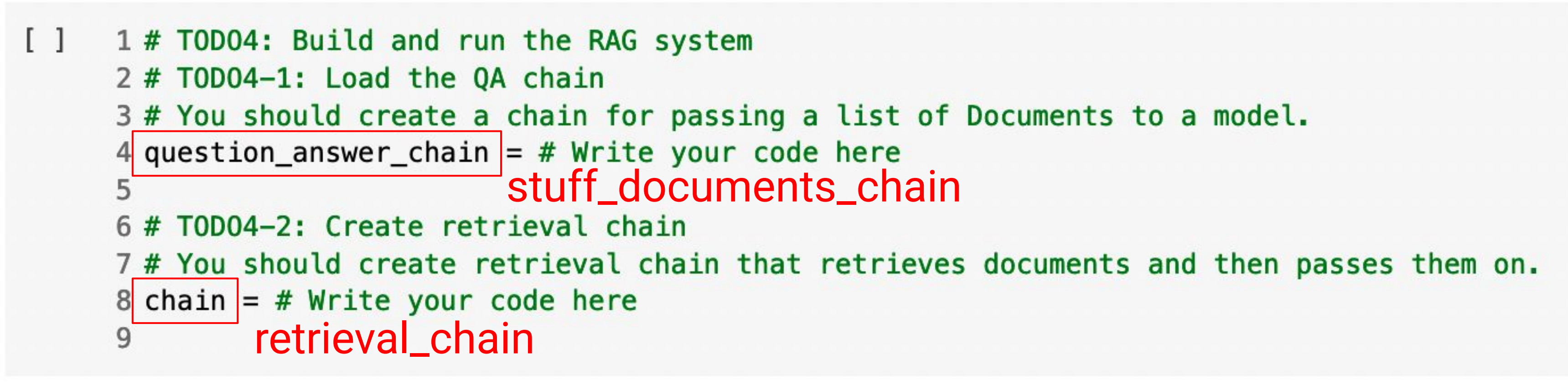
Prompt setting

```
# TODO3: Set up the `system_prompt` and configure the prompt.
system_prompt = # Write your code here
prompt = ChatPromptTemplate.from_messages(
    [
        ("system", system_prompt),
        ("human", "{input}"),
    ]
)
```

- For the vectorspace, the common algorithm would be used like Faiss, Chroma...(https://python.langchain.com/docs/integrations/vectorstores/) to deal with the extreme huge database.

Write a system prompt to guide the document reader model in performing the desired task.

TODO4: Build and run the RAG system

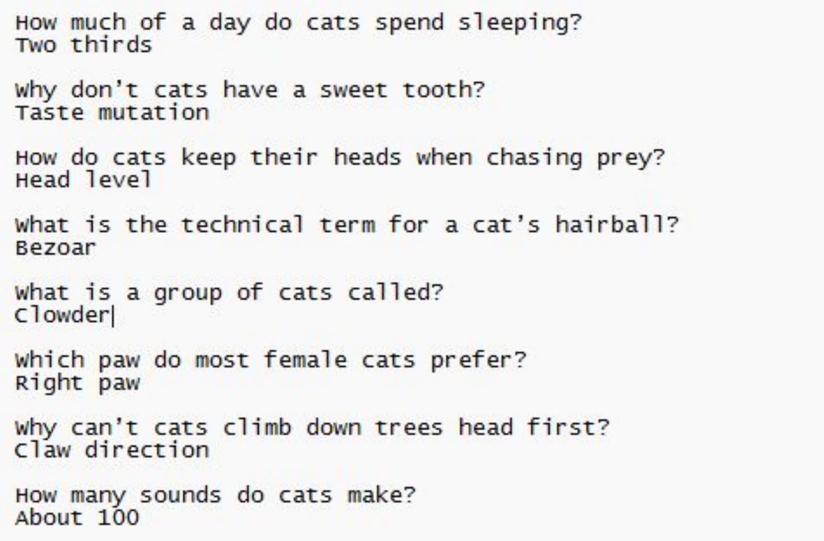


```
# TODO4: Build and run the RAG system
# TODO4-1: Load the QA chain
# You should create a chain for passing a list of Documents to a model.
question_answer_chain = # Write your code here

# TODO4-2: Create retrieval chain
# You should create retrieval chain that retrieves documents and then passes them on.
chain = # Write your code here

```

stuff_documents_chain

retrieval_chain

The Question and Answer Pairs

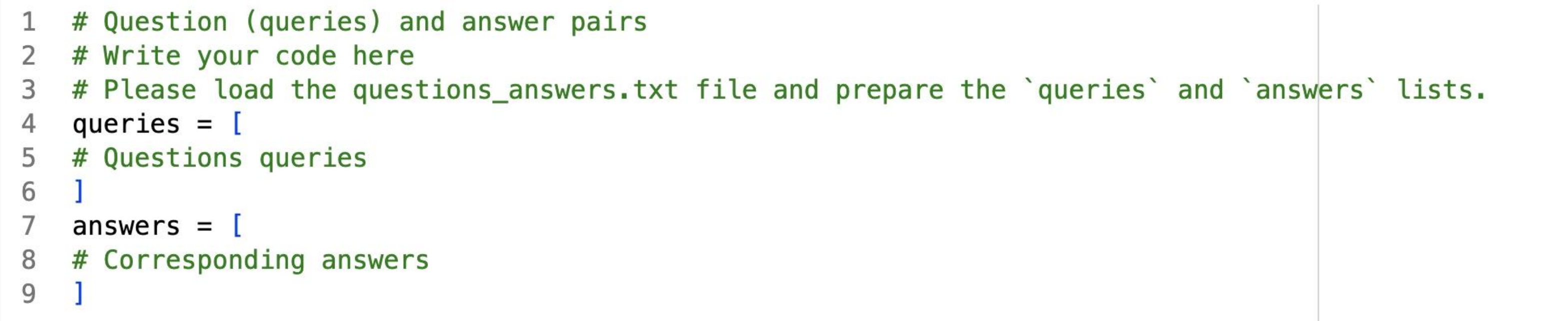


```
How much of a day do cats spend sleeping?
Two thirds

Why don't cats have a sweet tooth?
Taste mutation

How do cats keep their heads when chasing prey?
Head level

What is the technical term for a cat's hairball?
Bezoar

What is a group of cats called?
Clowder|

Which paw do most female cats prefer?
Right paw

Why can't cats climb down trees head first?
Claw direction

How many sounds do cats make?
About 100
```

The queries (the ten questions) and the correct answers are provided in the notebook file.

There are 150 QA pairs, each corresponding to an entry in the Cat-Fact dataset, and they appear in the same order as in the original dataset.

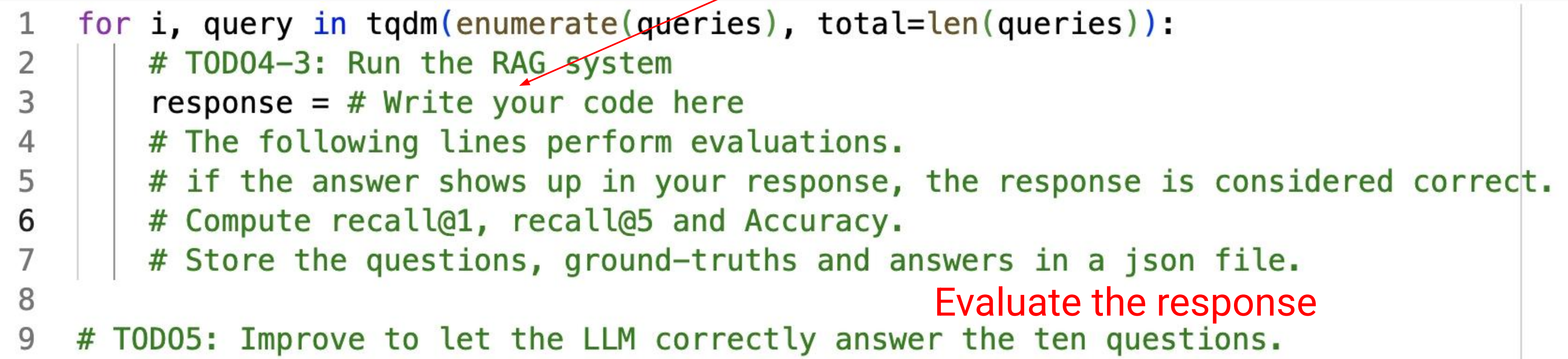
Load



```python
# Question (queries) and answer pairs
# Write your code here
# Please load the questions_answers.txt file and prepare the `queries` and `answers` lists.
queries = [
# Questions queries
]
answers = [
# Corresponding answers
]
```

TODO4-3 and TODO5: Improve to let the LLM correctly answer the ten questions.

Input the query to the RAG system and get the response



```python
for i, query in tqdm(enumerate(queries), total=len(queries)):
    # TODO4-3: Run the RAG system
    response = # Write your code here
    # The following lines perform evaluations.
    # if the answer shows up in your response, the response is considered correct.
    # Compute recall@1, recall@5 and Accuracy.
    # Store the questions, ground-truths and answers in a json file.

# TODO5: Improve to let the LLM correctly answer the ten questions.
```

Evaluate the response

Print the scores

Submission

Scoring for TODOs

TODO	Score
TODO1: Set up the environment of Ollama	5%
TODO2: Load the cat-facts dataset and prepare the retrieval database	10%
TODO3: Set up the `system_prompt` and configure the prompt.	10%
TODO4: Build and run the RAG system (Your submitted verion should use Llama3.2-1b model, if not, your score for TODO4 will be reduced by 50%)	10%
TODO5: Improve to let the LLM correctly answer the 150 questions. (Use Llama3.2-1b)	10%

You must upload a JSON file that includes all **test questions**, their **ground-truth answers**, and your corresponding **predictions**. The JSON file is like:

[{“Query”: … , “Ground_Truth”: …, “Prediction”: …}, {“Query”: … , “Ground_Truth”: …, “Prediction”: …}, …]

Scoring

Coding work : 45%

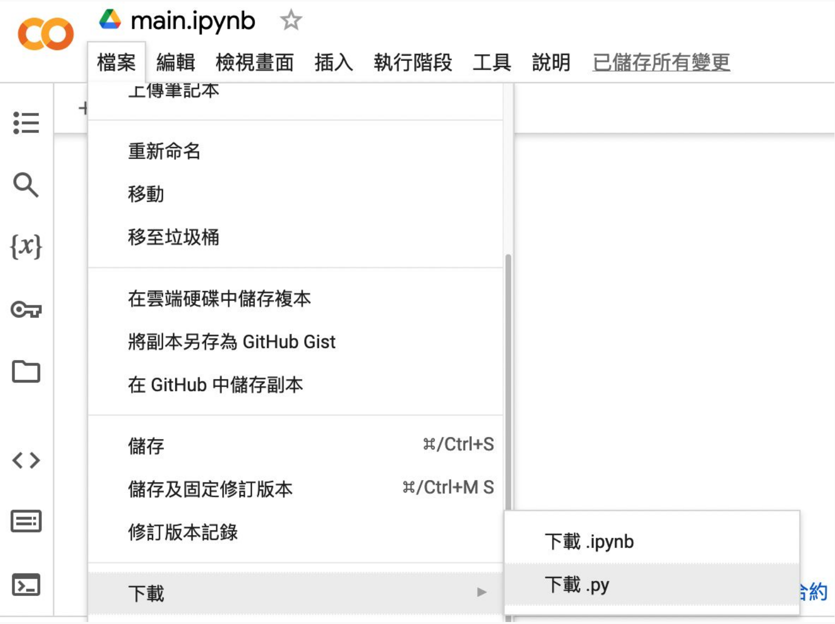
- You must upload a JSON file that includes all **test questions**, their **ground-truth answers**, and your corresponding **predictions**. Additionally, your report must include a screenshot showing your testing logs and the resulting accuracy.
- For retrieval please report recall@1 and recall@5 scores, for generation, please report the exact match (EM) score.
- If either of these required items is missing, your score for TODO5 will be reduced by 50%.

Report: 55%

- (5%) Please describe the **details of your implementation for the RAG system** (please tell us 1. What's in your RAG system? 2. What's your prompt? 3. What's new in your code in comparison with the code from our lab course?) in this assignment.
- (10%) Please provide analysis for the RAG performance using different prompts in **generator** model.
- (10%) Please analyze the RAG performance under different input data formats provided to the **retriever**.
- (10%) Please provide analysis for the RAG performance using different **input order** of the retrieved documents to the generator model.
- (10%) Please analyze the generation performance of the model when **counterfactual information** is introduced into the input sequence of the generator.
- (10%) Anything that can strengthen your report.

Delivery policies: File formats

- Coding work: Python file (.py or .ipynb)
    - Download your script via Colab.
- Predictions: Json file (.json)
- Package list: requirements.txt
    - E.g., numpy==1.26.3
- Report: Microsoft Word (.docx or .pdf)
- No other formats are allowed.
- Zip the files above before uploading you assignment.



Delivery policies: Filenames

- Do not forget the correct formats!

	Filename rule	Filename example
Coding work	NLP_HW4_school_student_ID.py	NLP_HW4_NTHU_12345678.py
Predictions	NLP_HW4_school_student_ID.json	NLP_HW4_NTHU_12345678.json
Report	NLP_HW4_school_student_ID.docx	NLP_HW4_NTHU_12345678.docx
Package list	requirements.txt	
Zipped file	NLP_HW4_school_student_ID.zip	NLP_HW4_NTHU_12345678.zip

Delivery policies: Things You should include

- In your report:

	Example	
Environment types	**If Colab or Kaggle**	**If local**
Running environment	Colab	System: Ubuntu 22.04, CPU: Ryzen 7-7800X3D
Python version	Colab	Python 3.10.1

Delivery policies: Rules of coding

- If you use ChatGPT or Generative AI, please specify your usage **both** in:
  - **Code comments**
  - **Reports**
- **No plagiarism**. You should not copy and paste from your classmates. **Submit duplicate code or report will get 0 point !**
- Please provide links if you take the code from the Internet as reference.
- The following behaviors will cause loss in the score of the assignment: **(1) Usage with Generative AI without specifications (2) Internet sources without specifications (3) Plagiarism.**

Uploading the zipped file

- Please upload your file to NTU COOL.
- Typically, you will have at most three weeks to finish this assignment.
- If you have any question, please e-mail to **nthuikmlab@gmail.com**
- For sending an email, please include **[NLP AS4]** in your email title.
  - Example email title: [NLP AS4] Question about the scoring

Punishments

Rule	Name your code: NLP_HW3_school_student_ID.py (.ipynb is also acceptable)	Name your report: NLP_HW3_school_student_ID.docx (.pdf is also acceptable)	Name your file: NLP_HW3_school_student_ID.zip	Include requirements.txt
Punishment	-5	-5	-5	-5
Rule	Include python version in your report	Do not modify the report template	Your code or report should not shows a high degree of similarity to another student’s submission.	
Punishment	-5	-5	-100 for both	

If you are using Colab, go to **File** → **Download** → **Download .py** to obtain the Python file.

Uploading the zipped file

- Please upload your file to NTU COOL.
- You will have three weeks to finish this assignment.
- If you have any question, please e-mail to **nthuikmlab@gmail.com**