

Δημιουργία RAG αρχιτεκτονικής για ερωτήσεις φυσικής γλώσσας σε ειδησεογραφικά άρθρα

ΜΥΕ003-ΑΝΑΚΤΗΣΗ ΠΛΗΡΟΦΟΡΙΑΣ

ΑΘΑΝΑΣΙΟΣ ΤΟΝΙΟΣ ΑΜ:4933, ΠΑΝΑΓΙΩΤΗΣ ΤΡΥΠΟΣ ΑΜ:5131

Περιεχόμενα

Εισαγωγή.....	2
Υλοποίηση.....	3
rag_data_preparation	3
query_engine.py.....	5
gui_app.py	7
Πειραματικές δοκιμές.....	9

Github Repository:

<https://github.com/ThePhantom2307/MYE003-Articles-SearchEngine>

Εισαγωγή

Στην παρούσα εργασία θα υλοποιήσουμε την κατασκευή μιας RAG αρχιτεκτονικής που θα μας επιτρέψει να απαντάμε ερωτήσεις σχετικά ειδησεογραφικά άρθρα. Τα άρθρα που θα χρησιμοποιηθούν για το augmented κομμάτι της υλοποίησης, προέρχονται από το CNN, είναι δημοσιευμένα κατά την περίοδο 2011 έως 2022 και καλύπτουν ένα ευρύ φάσμα δημοσιογραφικών θεμάτων.

Η RAG αρχιτεκτονική έχει στόχο την βελτίωση των απαντήσεων που παράγει ένα LLM. Αυτό γίνεται αρχικά από την ανάκτηση των πληροφοριών που κάνει πχ στην δικιά μας περίπτωση από το .csv αρχείο που περιέχει τα άρθρα. Στη συνέχεια δίνονται ως ενίσχυση στο LLM όπου και τελικά θα παράγει την απάντηση του χρησιμοποιώντας τόσο την ερώτηση του χρήστη όσο και τις ανακτημένες πληροφορίες.

Στη έκταση της παρούσας αναφοράς θα περιγράψουμε την υλοποίηση μας για την RAG αρχιτεκτονική καθώς και θα εμβαθύνουμε σχετικά με τις βιβλιοθήκες και τις συναρτήσεις αυτών που χρησιμοποιήσαμε. Τέλος θα παραθέσουμε τα μερικά πειραματικά queries μαζί με τα αποτελέσματα τους και θα τα συγκρίνουμε με και χωρίς context.

Υλοποίηση

Για την υλοποίηση της εργασίας έχουμε χωρίσει το source code σε τρία μέρη (modular σχεδίαση) καθώς θεωρήσαμε ότι θα είναι πιο εύκολο στην κατανόηση ενός τρίτου αλλά και για λόγους επεκτασιμότητας ή συντήρησης στο μέλλον. Τα μέρη που το αποτελούν είναι το *rag_data_preparation.py* το οποίο είναι υπεύθυνο για το preprocessing των αρχείων, το *query_engine.py* το οποίο παίρνει τα chunks που έχουν παραχθεί, κατασκευάζει το query καθώς και το response του μοντέλου το οποίο και καλεί. Τέλος, το τρίτο μέρος αποτελεί την υλοποίηση ενός απλού gui περιβάλλοντος για την ευκολότερη χρήση της μηχανής.

rag_data_preparation

Σε αυτό το μέρος θα αναλύσουμε το preprocessing που θα κάνουμε στο αρχείο *CNN_Articles_clean* για να μπορέσουμε κάνουμε extract τα άρθρα από αυτό και να τα χρησιμοποιήσουμε αργότερα ως context στο LLM.

Δημιουργούμε το αρχείο *rag_data_preparation.py* και κάνουμε import όλες τις βιβλιοθήκες που θα χρησιμοποιήσουμε στην συνέχεια.

```
import os
import torch
import pandas as pd
import numpy as np
import faiss
from sentence_transformers import SentenceTransformer
from langchain.text_splitter import RecursiveCharacterTextSplitter
```

Έπειτα θέτουμε ως σταθερές τα ονόματα των αρχείων που θα δημιουργηθούν, δηλαδή το index τύπου .faiss, το .csv που θα περιέχει τα chunks των άρθρων και το .csv από το οποίο θα εξάγουμε τα chunks.

```
INDEX_FILE = "articles_index.faiss"
CHUNKS_FILE = "chunk_metadata.csv"
CSV_FILE = "CNN_Articles_clean.csv"
```

Όλο το preprocessing γίνεται με την συνάρτηση *generate_faiss_and_chunks()*. Αρχικά, ελέγχουμε αν έχει ήδη παραχθεί το index file και το chunk file. Αυτός ο έλεγχος γίνεται ώστε ο χρήστης να μη χρειάζεται να δημιουργεί κάθε φορά τα αρχεία αυτά, καθώς ανάλογα με το hardware του (όπως θα δούμε και παρακάτω) μπορεί να πάρει αρκετή ώρα για την δημιουργία τους. Στη συνέχεια διαβάζουμε το αρχείο με τα άρθρα και το κρατάμε σε ένα dataframe με χρήση της βιβλιοθήκης pandas. Έπειτα, από το dataframe κρατάμε μόνο το πεδίο που περιέχει τα κείμενα των άρθρων (*df["Article_text"]*) στην μεταβλητή *articles* και τα μετατρέπουμε σε λίστα.

```
def generate_faiss_and_chunks():
    if os.path.exists(INDEX_FILE) and os.path.exists(CHUNKS_FILE): #Tsekaroume an yparxoun hdh
        return

    df = pd.read_csv(CSV_FILE) #Diavazoume to CSV
    articles = df["Article_text"].dropna().tolist() #Kratame to pedio Article text gia na paroume ta arthra

    text_splitter = RecursiveCharacterTextSplitter(chunk_size=500, chunk_overlap=100)
```

Συνεχίζοντας, ορίζουμε chunker/text splitter. Στη υλοποίηση μας αποφασίσαμε να χρησιμοποιήσουμε τον RecursiveCharacterTextSplitter από την βιβλιοθήκη LangChain. Ο λόγος που χρησιμοποιήσαμε αυτόν και όχι τον SemanticChunker είναι επειδή με τον RecursiveCharacterTextSplitter ορίσουμε εμείς το chunk size καθώς και το chunk overlap. Συγκεκριμένα, το `chunk_size=500` δηλώνει ότι κάθε chunk θα είναι 500 χαρακτήρων και το `chunk_overlap=100` που σημαίνει ότι κάθε chunk επικαλύπτεται με το προηγούμενο κατά 100 χαρακτήρες. Αυτό βοηθάει στο να μη χάνεται το νόημα όταν σπάμε στο κείμενο σε chunks.

Έπειτα, δημιουργούμε μια λίστα όπου σε αυτή θα κρατάμε τα chunks και διατρέχουμε όλα τα άρθρα. Για κάθε άρθρο χρησιμοποιούμε τη συνάρτηση `split_text` η οποία καλείται από τον splitter που έχουμε ορίσει, δηλαδή τον RecursiveCharacterTextSplitter και επιστρέφει μια λίστα με τα chunks που προέκυψαν από το άρθρο, την οποία και προσθέτουμε/ενώνουμε με τη λίστα που δημιουργήσαμε αρχικά.

```
chunks = [] #Πίνακας για τα chunks
for article in articles:
    chunks.extend(text_splitter.split_text(article)) #Προσθέτουμε τα chunks που μας δίνει ο text_splitter
```

Στη συνέχεια, κρατάμε σε μια μεταβλητή `device` το οποίο παίρνει την τιμή “cuda” ή `cpu` ανάλογα με το αν υπάρχει διαθέσιμη `gpu` που υποστηρίζει `cuda` ή όχι. Με αυτό τον τρόπο δίνουμε στο χρήστη τη δυνατότητα στη συνέχεια όπου και θα χρειαστεί να δημιουργηθούν τα `embeddings` για το κάθε chunk, να χρησιμοποιήσει την κάρτα γραφικών του για να κερδίσει χρόνο (περισσότερα στη συνέχεια).

```
device = "cuda" if torch.cuda.is_available() else "cpu"
print("Using device:", device) #DEBUGGING
```

Σε αυτό το σημείο θέτουμε το μοντέλο που θα δημιουργήσει τα επιμέρους διανύσματα (`embeddings`). Συγκεκριμένα, από την βιβλιοθήκη `sentence_transformers` χρησιμοποιούμε τη μέθοδο `SentenceTransformer` και της περνάμε ως όρισμα το μοντέλο `all-MiniLM-L6-v2` και το `device` με το οποίο θα το χρησιμοποιήσουμε. Το μοντέλο αυτό, μετατρέπει τα chunks σε διανύσματα σταθερού μεγέθους των 384 διαστάσεων. Επίσης, το συγκεκριμένο μοντέλο μπορεί εκτός από `cpu` να χρησιμοποιηθεί και με τη χρήση `gpu` μέσω του `pytorch`. Έτσι, στη συνέχεια καλούμε τη συνάρτηση `encode` του `model` για τα chunks που έχουμε δημιουργήσει πιο πριν. Επειδή λοιπόν αυτό που μας επιστρέφεται είναι ένας `NumPy` πίνακας όπου το μηδενικό στοιχείο είναι ο αριθμός των `embeddings` που έχουμε και το πρώτο στοιχείο είναι ο αριθμός των διαστάσεων αυτών. Έτσι, κρατάμε στην μεταβλητή `dimension` τη διάσταση των `embeddings` καθώς θα μας χρειαστεί στη συνέχεια.

```
model = SentenceTransformer('all-MiniLM-L6-v2', device=device)
embeddings = model.encode(chunks, show_progress_bar=True)
dimension = embeddings.shape[1]
```

Κάτι που πιστεύουμε αξίζει να σημειωθεί είναι η διαφορά στον χρόνο δημιουργίας των `embeddings` με `gpu` και με `cpu`. Συγκεκριμένα, τρέξαμε την συνάρτηση σε `pc` που δεν υποστηρίζει `cuda` και ο χρόνος που χρειάστηκε για την δημιουργία των

διανυσμάτων ήταν περίπου μισή ώρα. Αντίθετα, σε pc που υποστήριζε cuda και συγκεκριμένα με κάρτα γραφικών GTX1650Ti και cuda 11.8 ο χρόνος που χρειάστηκε ήταν μόλις 2 λεπτά. Για αυτό στο github repository περιλαμβάνουμε εξ'αρχής τα αρχεία `chunk_metadata.csv` και `articles_index.faiss` ώστε σε περίπτωση που ο υπολογιστής του χρήστη δεν υποστηρίζει cuda να μπορεί να τα περάσει απευθείας.

Προχωρώντας, φτάνουμε σε ένα σημαντικό σημείο όπου, ορίζουμε το λεξικό που θα αποθηκευτούν τα embeddings. Θα χρησιμοποιήσουμε ένα FAISS λεξικό και συγκεκριμένα αυτό που χρησιμοποιεί την ευκλείδεια απόσταση (FlatL2) για να συγκρίνει την απόσταση μεταξύ των διανυσμάτων, δηλαδή το similarity τους. Αυτό θα μας χρειαστεί στη συνέχεια όταν θα πρέπει να συγκρίνουμε το similarity του διανύσματος του query με τα διανύσματα που θα έχουμε αποθηκευμένα στο index, ώστε να παίρνουμε τα πιο σχετικά αποτελέσματα. Για να χρησιμοποιήσουμε τη συνάρτηση `faiss.IndexFlatL2` θα χρειαστούμε την βιβλιοθήκη faiss. Περνάμε ως όρισμα στην συνάρτηση τη διάσταση των διανυσμάτων ώστε το index να είναι τέτοιο μεγέθους και το κρατάμε στην ομώνυμη μεταβλητή. Προσθέτουμε σε αυτό τα embeddings και το γράφουμε σε αρχείο.

```
index = faiss.IndexFlatL2(dimension)
index.add(np.array(embeddings))
faiss.write_index(index, INDEX_FILE)
```

Τέλος, δημιουργούμε ένα dataframe που θα περιέχει μια στήλη στην οποία θα υπάρχουν όλα τα chunks. Τέλος μετατρέπουμε το dataframe σε csv αρχείο.

```
chunk_metadata = pd.DataFrame({"chunk": chunks})
chunk_metadata.to_csv(CHUNKS_FILE, index=False)
```

query_engine.py

Στο παρόν σημείο της εργασίας θα αναλύσουμε το αρχείο `query_engine.py` στο οποίο υλοποιούμε τις συναρτήσεις για την δημιουργία του prompt καθώς και για την δημιουργία της απάντησης που θα πάρουμε από το LLM.

Όπως και πριν, ξεκινάμε με το να κάνουμε import τις απαραίτητες βιβλιοθήκες που θα μας χρειαστούν στη συνέχεια.

```
import faiss
import pandas as pd
import numpy as np
from sentence_transformers import SentenceTransformer
from ollama import Client
```

Έπειτα, ορίζουμε και πάλι κάποιες σταθερές όπου κρατάμε το όνομα του index, του csv αρχείου που περιέχει τα chunks και το μοντέλο που θα χρησιμοποιήσουμε για τον υπολογισμό των embeddings (όπως και πριν χρησιμοποιούμε το all-MiniLM-L6-v2). Επιπλέον, δημιουργούμε την σταθερά client όπου είναι αυτή που θα κρατάει το local host του ollama. Η συνάρτηση `Client` προέρχεται από το ollama package, το οποίο επιτρέπει την επικοινωνία με τον ollama server που τρέχει locally στο pc μας. Το

ollama κατά την εκτέλεση του, ξεκινάει έναν local server στο port 11434 και έτσι καλώντας την Client δημιουργούμε ένα αντικείμενο που μπορεί να στείλει requests με prompts και να πάρει responses από το μοντέλο.

```
INDEX_FILE = "articles_index.faiss"
CHUNKS_FILE = "chunk_metadata.csv"
embedding_model = SentenceTransformer('all-MiniLM-L6-v2')
client = Client(host='http://localhost:11434')
```

Η πρώτη συνάρτηση που θα υλοποιήσουμε στο αρχείο είναι η *build_prompt*. Η συνάρτηση είναι πολύ απλή και ουσιαστικά ο σκοπός της είναι η δημιουργία ενός δομημένου prompt με βάση το query του χρήστη και των chunks που θα έχουν βρεθεί προηγουμένως και τα οποία θα χρησιμοποιηθούν ως context. Κρατάμε ως context την ένωση όλων των chunks που έχουμε ανακτήσει και απλά επιστρέφουμε το prompt που μας προτάθηκε από την εκφώνηση της άσκησης.

```
def build_prompt(query, retrieved_chunks):
    context = "\n\n".join(retrieved_chunks)
    return (
        f"You are an AI assistant. Answer the following question:\n"
        f"{query}\n\n"
        f"based on the following context:\n\n"
        f"{context}"
    )
```

Στη συνέχεια, υλοποιούμε την συνάρτηση *answer_query*. Η συνάρτηση αυτή, είναι στην ουσία μια μηχανή ερωτήσεων-απαντήσεων η οποία παίρνει το query του χρήστη και χρησιμοποιώντας το LLM που έχουμε επιλέξει (στη περίπτωση μας το llama3.2:3b) δίνει ένα response με βάση τα 5 πιο relevant chunks σε σχέση με το query του χρήστη. Αρχικά, διαβάζουμε το faiss index και το csv αρχείο με τα chunks. Δημιουργούμε το embedding του query του χρήστη, με τον ίδιο τρόπο που δημιουργούσαμε τα embeddings των chunks που προσθέσαμε στο faiss index. Το διάνυσμα του query που θα πάρουμε έχει την ίδια διάσταση και με τα διανύσματα των chunks που βρίσκονται στο faiss index έτσι ώστε να μπορούν να συγκριθούν με αυτά. Η σύγκριση γίνεται στην επόμενη εντολή όπου και γίνεται το search στο index. Συγκεκριμένα, πάνω στο index καλούμε την συνάρτηση search της βιβλιοθήκης faiss, η οποία δέχεται ως όρισμα έναν NumPy πίνακα (εξού και η μετατροπή του query_embedding σε np.array) και τον αριθμός που καθορίζει πόσους κοντινούς γείτονες να επιστρέψει για κάθε διάνυσμα αναζήτησης (στην περίπτωση μας επιλέξαμε τα 5 κοντινότερα).

```
def answer_query(user_query, model_name="llama3.2:3b", k=5):
    index = faiss.read_index(INDEX_FILE)
    chunks_df = pd.read_csv(CHUNKS_FILE)

    query_embedding = embedding_model.encode([user_query])
    distances, indices = index.search(np.array(query_embedding), k)
    top_chunks = [chunks_df.iloc[idx]["chunk"] for idx in indices[0]]
    prompt = build_prompt(user_query, top_chunks)

    response = client.generate(model=model_name, prompt=prompt)
    return response['response'].strip(), top_chunks
```

Αυτά που επιστρέφει η search είναι έναν πίνακα με τις αποστάσεις το k κοντινότερων διανυσμάτων από το query και έναν πίνακα με τα indices αυτών των διανυσμάτων στο faiss index. Στην υλοποίηση μας αυτό που θα μας χρειαστεί είναι το δεύτερο. Έπειτα, κάνοντας ένα list comprehension διατρέχουμε το dataframe των chunks (`chunks_df.iloc[idx]["chunk"] for idx`) και κρατάμε τα chunks που βρίσκονται στις θέσεις που μας δείχνει ο πίνακας indices (`in indices[0]`). Ουσιαστικά, έχοντας κρατήσει τις θέσεις των k-relevant chunks στον πίνακα indices (σε μορφή διανυσμάτων πριν στο faiss index) απλά πηγαίνουμε στις αντίστοιχες γραμμές του data frame με μοναδική στήλη “chunk” και κρατάμε στην λίστα `top_chunks` τα πιο σχετικά με την ερώτηση chunks. Κρατάμε ακόμα το prompt στην ομώνυμη μεταβλητή, καλώντας απλά την `built_prompt` και δίνοντας της ως ορίσματα το query και τα `top_chunks` που ανακτήσαμε πριν. Χρησιμοποιώντας το instance client που ορίσαμε στην αρχή, καλούμε το LLM (llama3.2:3b) που τρέχει τοπικά μέσω του ollama server και του δίνουμε το prompt που έχουμε φτιάξει. Αυτό που θα μας επιστραφεί είναι ένα λεξικό όπου ο πρώτος της όρος είναι το “`response`”: `response string` “. Για αυτό και στην τελευταία γραμμή επιστρέφουμε το περιεχόμενο του πεδίου response (`response['response'].strip()`) καθώς και τα `top_chunks`. Όσον αφορά του LLM, όπως αναφέρθηκε και προηγουμένως, επιλέξαμε το llama3.2:3b αντί του llama3.2:1b καθώς μετά από μια σύντομη έρευνα καταλήξαμε ότι είναι πιο ικανό για μεγαλύτερα project όπως πχ η υλοποίηση μιας RAG αρχιτεκτονικής. Συγκεκριμένα, η κύρια διαφορά μεταξύ τους βρίσκεται στις παραμέτρους όπου μπορεί να δεχτεί το εκάστοτε μοντέλο, όπου το llama3.2:3b δέχεται 3 billion parameters ενώ το llama3.2:1b δέχεται 1 billion. Όσον αφορά τον τεχνικό τομέα, το μοντέλο που επιλέξαμε έχει μεγαλύτερη κατανάλωση πόρων, ωστόσο θεωρήσαμε ότι είναι ένα καλό tradeoff για την αυξημένη απόδοση που θα παίρναμε σε σχέση με το άλλο μοντέλο.

gui_app.py

Το `gui_app.py` είναι το αρχείο του προγράμματος που περιέχει το main πρόγραμμα. Είναι υπεύθυνο για την κλήση των κατάλληλων συναρτήσεων κατά την εκτέλεσή του και προσφέρει το γραφικό περιβάλλον στον χρήστη.

Το main πρόγραμμα ελέγχει αν εκτελέστηκε απευθείας ώστε να καλέσει την launch_gui() συνάρτηση η οποία δημιουργεί το γραφικό περιβάλλον και θέτει ποιες λειτουργίες θα εκτελεστούν αν πατηθεί το “submit” πλήκτρο ή το “enter” πλήκτρο από το πληκτρολόγιο.

```
def launch_gui():
    global root, entry, text_output, submit_button

    root = Tk()
    root.title("RAG AI Assistant")
    root.geometry("700x500")

    Label(root, text="Enter your prompt:", font=("Arial", 12)).pack(pady=5)

    entry = Entry(root, width=80)
    entry.pack(pady=5)
    entry.bind("<Return>", process_query)

    submit_button = Button(root, text="Submit", command=process_query)
    submit_button.pack(pady=5)

    scrollbar = Scrollbar(root, orient=VERTICAL)
    scrollbar.pack(side=RIGHT, fill=Y)

    text_output = Text(root, wrap="word", yscrollcommand=scrollbar.set)
    text_output.pack(expand=True, fill="both")
    scrollbar.config(command=text_output.yview)
```

Επίσης στην ίδια συνάρτηση γίνεται το initialization των αρχείων τα οποία θα χρησιμοποιηθούν αργότερα για να μπορεί να βγάλει αποτέλεσμα το μοντέλο. Αν υπάρχουν τα αρχεία αυτά τότε δεν τα δημιουργεί ξανά.

```
if not os.path.exists(INDEX_FILE) or not os.path.exists(CHUNKS_FILE):
    initialize_data()
else:
    text_output.insert(END, "Data already initialized. Enter your prompt.\n")
```

Η συνάρτηση initialize_data απενεργοποιεί το πλήκτρο “submit” και καλεί με την συνάρτηση Thread() την run_init() ώστε να μην μπλοκάρει το πρόγραμμα κατά την διάρκεια αυτής της διαδικασίας. Η run_init() καλεί την generate_faiss_and_chunks() την οποία αναφέραμε νωρίτερα.

```
def run_init():
    try:
        generate_faiss_and_chunks()
        def finish():
            text_output.insert(END, "Initialization completed. Enter your prompt.\n")
            submit_button.config(state=NORMAL)
        root.after(0, finish)
    except Exception as e:
        root.after(0, lambda: text_output.insert(END, f"\nInitialization failed:\n{str(e)}"))

def initialize_data():
    text_output.insert(END, "Initializing FAISS index and chunks...\n")
    submit_button.config(state=DISABLED)
    threading.Thread(target=run_init, daemon=True).start()
```

Τέλος η το πλήκτρο “submit” όπως και το “enter” καλούν την συνάρτηση process_query η οποία καλεί την run_query() με την χρήση Thread() συνάρτησης

ώστε να μην μπλοκάρει το πρόγραμμα. Η `run_query()` καλεί με την σειρά της την `answer_query()` ώστε να πάρει το αποτέλεσμα από το ερώτημα του χρήστη και το εμφανίζει.

```
def run_query(query):
    try:
        response, used_chunks = answer_query(query)

        def update_ui():
            text_output.delete("1.0", END)
            text_output.insert(END, "► Response:\n")
            text_output.insert(END, response)

            text_output.insert(END, "\n\n► Relevant chunks:\n")
            for i, chunk in enumerate(used_chunks, start=1):
                text_output.insert(END, f"\n[{i}] {chunk}\n")

            submit_button.config(state=NORMAL)
            entry.config(state=NORMAL)
            entry.delete(0, END)
            entry.focus()

        root.after(0, update_ui)

    except Exception as e:
        def show_error():
            text_output.insert(END, f"\nError:\n{str(e)}")
            submit_button.config(state=NORMAL)
            root.after(0, show_error)

def process_query(event=None):
    query = entry.get().strip()
    if not query:
        return

    text_output.delete("1.0", END)
    text_output.insert(END, "Searching...\n")
    submit_button.config(state=DISABLED)
    entry.config(state=DISABLED)

    threading.Thread(target=run_query, args=(query,), daemon=True).start()
```

Πειραματικές δοκιμές

Στο τελευταίο σημείο της παρούσας εργασίας, θα κάνουμε κάποιες δοκιμές όπως μας ζητήθηκε από την εκφώνηση της άσκησης. Συγκεκριμένα, θα εκτελέσουμε 5 queries και στη συνέχεια θα παραθέσουμε τα αποτελέσματα που θα μας δώσει το μοντέλο με και χωρίς το context.

“What is Covid?”

Enter your prompt:

What is covid?

Submit

> Response:

Based on the context provided, Covid appears to refer to the COVID-19 pandemic caused by the SARS-CoV-2 virus. This pandemic has had significant effects on global health, economies, and society as a whole, leading to widespread illness, hospitalizations, and deaths. The text mentions long-term effects of COVID-19 infection, known as long Covid, which can cause symptoms such as fatigue, brain fog, shortness of breath, and cardiac issues in some individuals.

> Relevant chunks:

[1] to deal with Covid as part of our lives."This story has been updated to include more from the full interview and additional context.

[2] in the future. In this case, it means acknowledging an entirely new disease: long Covid. Many Americans are enduring the lingering effects of a past infection, battling health conditions like fatigue, brain fog, shortness of breath, cardiac issues. The list of long Covid symptoms is lengthy and varied: there are no answers as to who and why, nor are there easy, one-size-fits-all treatments. We are in the early days of this disease, but I was particularly struck by the recent paper indicating a

[3] analysis. My producers and I spent a lot of time looking at whiteboards where we kept tabs on the growing numbers and locations of Covid-19 cases. One day, I remember thinking, "If this isn't the very definition of a pandemic, I don't know what is. So why is no one else calling it that?" And so we did.JUST WATCHEDDr. Gupta: Here's why CNN is calling this a pandemic (2020)ReplayMore Videos...MUST WATCHDr. Gupta: Here's why CNN is calling this a pandemic (2020) 04:40 Although the line was clear

[4] the Covid-19 pandemic as "an example of gloomy scientists being proved right." "We still cling with parts of our minds to the infantile belief that the world was made for our gratification and pleasure," he said. "And we combine this narcissism with an assumption of our own immortality." "We believe that someone else will clear up the mess, because that is what someone else has always done," he added. "We trash our habitats, again and again, with the inductive reasoning that we've gotten away

[5] Covid-19 crisis and pleading with everyone to get vaccinated. Kenneth Coenye, chief medical officer at the Sint-Jan Clinic in Brussels, told CNN that during the last weekend of August, only four out of 70 patients being treated in Covid ICUs in the city were vaccinated. "It's very painful, of course, because they don't only get sick, they die. And that is so completely preventable," he said.Speaking Saturday, the leader of Wallonia -- Belgium's largest region -- urged citizens who have not yet

Me context

Enter your prompt:

What is covid?

Submit

> Response:

COVID-19, also known as coronavirus disease 2019, is a respiratory illness caused by the SARS-CoV-2 virus. It was first identified in Wuhan, China in December 2019 and has since become a global pandemic.

The COVID-19 virus is a member of the coronavirus family, which includes other viruses that can cause respiratory illness in humans. The SARS-CoV-2 virus is believed to have originated from an animal source, possibly bats, and was transmitted to humans through an intermediate host, possibly at a market in Wuhan where animals were sold.

The COVID-19 virus attacks the lungs and airways, causing inflammation and damage to these tissues. This can lead to symptoms such as:

- * Fever
- * Cough
- * Shortness of breath
- * Fatigue
- * Headache
- * Sore throat

In severe cases, COVID-19 can cause pneumonia, acute respiratory distress syndrome (ARDS), and even death.

The spread of COVID-19 is often facilitated by close contact with an infected person, such as through:

- * Respiratory droplets that are released when an infected person talks, coughs, or sneezes
- * Close contact with an infected person's skin or eyes
- * Touching contaminated surfaces and then touching one's face

To prevent the spread of COVID-19, public health officials recommend practices such as:

- * Frequent handwashing with soap and water
- * Wearing a mask in public places
- * Staying at least 6 feet away from others
- * Avoiding close contact with people who are sick
- * Getting vaccinated against COVID-19

Vaccines have been developed to prevent severe illness and hospitalization due to COVID-19. These vaccines have been shown to be highly effective in preventing the spread of the virus.

Overall, COVID-19 is a serious public health threat that requires continued vigilance and cooperation to prevent its spread.

Χωρίς context

“Tell me some facts about Wimbledon”

Enter your prompt:

Tell me some facts about Wimbledon

Submit

> Response:

Here are some facts about Wimbledon:

1. It's one of four Grand Slam tennis tournaments, along with the Australian Open, French Open, and US Open.
2. The tournament is held at the All England Lawn Tennis Club in London.
3. Wimbledon is the only Grand Slam event still played on grass.
4. Roger Federer holds the record for most singles wins (male) with eight victories.
5. Martina Navratilova holds the record for most singles wins (female) with nine victories.
6. The oldest winner of the Men's Singles title is Roger Federer, who won in 2017 at the age of 35.
7. Wimbledon was suspended during World War I and World War II.
8. A bomb hit Centre Court in 1940, resulting in the loss of 1,200 seats.
9. Female winners received equal cash awards for the first time in 2007.
10. The retractable roof over Centre Court was unveiled in 2009.
11. Wimbledon is no longer held during the summer due to the COVID-19 pandemic, but it returned in 2022.

Additionally, I found some interesting facts about this year's tournament:

- * It was the first time a Romanian player won the tournament.
- * Australian top seed Dylan Alcott completed a golden slam by winning both singles and wheelchair events.

> Relevant chunks:

[1] also the first Romanian to win the tournament.Other FactsWimbledon is one of four Grand Slam tennis tournaments. The others are the Australian Open, the French Open and the US Open.Wimbledon takes place at the All England Lawn Tennis Club in London. Wimbledon is the only Grand Slam event that is still played on grass.RecordsMost singles wins (male) - Roger Federer with eight.Most singles wins (female) - Martina Navratilova with nine. Oldest winner (male) - Roger Federer won the Men's Singles

[2] the year before his first championship there."I noticed the mishits, awkward looking points from Roger and obviously the last set of course, 6-0," three-time Wimbledon champion Boris Becker told the BBC.This was Federer's first straight sets loss at Wimbledon since 2002 when he was beaten by Mario Ancic in the first round.

[3] is suspended during World War I.1922 - The Championships move to Church Road, Wimbledon's current location.1940-1945 - Wimbledon is suspended during World War II.October 1940 - During WWII, a bomb hits Centre Court, which results in the loss of 1,200 seats in the stadium.2007 - Female winners receive the same cash awards as the male winners for the first time.May 17, 2009 - The retractable roof over Centre Court is unveiled during an exhibition match.October 2018 - The The All England Lawn

[4] this month."For a long time it wasn't possible because Wimbledon didn't have a singles event. For a few years now they've had it... it's just great to see the tour and the professionalism going up each year," De Groot told reporters."To be the first one to actually get that title is just so special. I don't think I will ever forget that."Not long after on Sunday, Australian top seed Dylan Alcott also completed a golden slam in his respective event, taking down Dutchman Niels Vink 7-5 6-2 in the

[5] infectious around the time of the event and the rest likely to have acquired an infection then.The Wimbledon tennis championships, with around 300,000 people attending over the two weeks, recorded 881 cases."We've shown that we can reintroduce mass sports and cultural events safely but it is important that people remain cautious when mixing in very crowded settings," said Culture Minister Oliver Dowden."So that we can keep the football season, theaters and gigs safe with full crowds this

Me context

Enter your prompt:

Tell me some facts about Wimbledon

Submit

Response:

Wimbledon! Here are some fascinating facts about one of the most prestigious tennis tournaments in the world:

- "Oldest Tennis Tournament":** Wimbledon is the oldest tennis tournament in the world, with a rich history dating back to 1877. It was founded by Walter Clopton Wingfield, an Englishman who patented the first netted tennis court.
- "Grass Courts":** Wimbledon is still known for its iconic grass courts, which are maintained to perfection every year. The All England Lawn Tennis and Croquet Club (AELTC) spends over £100,000 per week on maintaining the grounds.
- "The White Suit Tradition":** Men's tennis players have traditionally worn white suits at Wimbledon since 1884. This tradition is still observed today, making it one of the most iconic fashion statements in sports.
- "Royal Connection":** The tournament has a long history of royal connections. Queen Victoria was the first monarch to attend Wimbledon, and her successors continued to support the event.
- "Women's Tennis Pioneer":** In 1884, Maud Watson became the first woman to win a singles title at Wimbledon, defeating Lottie Dod in the final. She paved the way for future generations of women's tennis players.
- "The Championships Name":** The official name of the tournament is "The Championships, Wimbledon," which reflects its origins as a competition between English and Scottish clubs.
- "Record Attendance":** In 2019, Wimbledon set an all-time record attendance of over 437,000 spectators, with some matches selling out in advance.
- "Unique Ball Boys/Girls":** Wimbledon is one of the few tennis tournaments that still employs ball boys and girls to retrieve balls during matches. This tradition dates back to the early days of tennis.
- "The Straw Hat Tradition":** In the 1970s, British players began wearing straw hats on court as a symbol of their national heritage. Today, it's a quirky tradition that adds to the tournament's charm.
- "A True Test of Endurance":** Wimbledon is known for its physically demanding nature, with players often suffering from fatigue and injury during the five-day competition.

These facts showcase the rich history, traditions, and unique aspects of Wimbledon, making it one of the most beloved tennis tournaments in the world.

Χωρίς context

“Tell me about Tom Felton”

Enter your prompt:

Tell me about Tom Felton

Submit

Response:

Based on the context, here's what I know about Tom Felton:

Tom Felton is a British actor, musician, and golfer. He is best known for playing the role of Draco Malfoy in the Harry Potter film franchise. He was born on September 22, 1987.

Recently, he experienced a medical incident while participating in a celebrity golf match at Whistling Straits, Wisconsin, ahead of the Ryder Cup. He collapsed on the course and was carted off for treatment, but was conscious after being transported to a local hospital. The cause of his medical incident is not specified.

Additionally, Tom Felton's personal life was mentioned in a separate context, where Tom Brady Sr., the father of NFL quarterback Tom Brady, discussed how he struggled with alcoholism during Tom DeBlass' childhood.

Relevant chunks:

- (CNN) Harry Potter star Tom Felton suffered a "medical incident" while playing in a celebrity golf match at Whistling Straits, Wisconsin, ahead of the Ryder Cup. Felton, best known for portraying Draco Malfoy in the Harry Potter films, was conscious after collapsing on the course on Thursday. He was then carted off for treatment. "In today's Ryder Cup Celebrity Match, actor and musician Tom Felton experienced a medical incident on the course while participating for Europe," said a statement from
- [2] experienced a medical incident on the course while participating for Europe," said a statement from PGA of America. "He was transported to a local hospital for treatment. No further details are available." CNN has reached out to Felton's representatives to inquire how he's doing. **Read More:** Felton is carted off the course. **HEAD:** The weird and wonderful outfits of fans at the Ryder Cup Felton, who turned 34 this week, was representing Team Europe in the celebrity Ryder Cup match alongside former
- [3] cited the 44-year-old quarterback as "a special person and the greatest quarterback of all-time" upon his departure. Speaking ahead of the week four fixture on Tom Curran's Patriots Talk podcast, Brady's father, Tom Brady Sr., discussed his son's exit from the team he spent two decades with. "My feeling is Tommy is extremely appreciative of everything that happened during his New England career and he's more than happy that he's moved on because it was pretty obvious that the Patriot regime felt
- [4] kind. His voice would raise, and he wasn't affectionate and then I realized there was something ... not smooth. **Read More:** "I really didn't comprehend at that time what alcoholism was, but I understood something wasn't normal." Tom DeBlass with his father, Tom DeBlass Sr. His father's alcohol and drug addiction was a recurring theme throughout DeBlass' childhood. On the day he was born, DeBlass says the police had to be called to his house to "rip me out of my father's hands" because he was so
- [5] he exceptionally honored, according to a Downing Street press release at the time. "Colonel Tom's fantastic fundraising broke records, inspired the whole country and provided us all with a beacon of light through the fog of coronavirus," Johnson said in the release.

Με context

Enter your prompt:

Tell me about Tom Felton

Submit

Response:

Tom Felton is a British actor, musician, and social media personality. He was born on September 22, 1987, in Kensington, London, England.

Felton is best known for his iconic role as Draco Malfoy in the Harry Potter film franchise, based on the book series by J.K. Rowling. He played the character from 2000 to 2011 and became a household name among fans of the series.

Apart from acting, Felton has pursued other interests, such as music. In 2016, he released his debut single "Wonderful World," followed by an album called "A Different Song" in 2022.

Felton has been open about his struggles with anxiety and depression, using his platform to raise awareness and reduce stigma around mental health issues. He is also a vocal advocate for LGBTQ+ rights and has supported various charitable organizations over the years.

In addition to acting and music, Felton has become increasingly popular on social media platforms such as Instagram and YouTube, where he shares his thoughts on life, creativity, and personal growth. His down-to-earth personality and witty humor have endeared him to fans worldwide.

Felton has continued to work in film and television, including roles in "The Flash," a CW series based on the DC Comics character, and "Origin," a 2020 sci-fi thriller film.

Throughout his career, Tom Felton has demonstrated his range as an actor and musician, while also building a loyal fan base across various platforms.

Χωρίς context

“Tell me about London's statues”

Enter your prompt:

Tell me about London's statues

Submit

► Responses:

Based on the provided context, here are some interesting facts about London's statues:

1. "Low representation of women": Only 4% of public sculptures in London depict women, while a staggering 8% represent animals and just 1% represent people of color.

2. "Highest percentage dedicated to women among UK cities": Despite this, London has the highest percentage of sculptures dedicated to women compared to other major UK cities.

3. "Queen Victoria is the most represented woman": Queen Victoria is the most frequently depicted woman in London's statues, reflecting her long and influential reign from 1819-1901.

4. "Inadequate representation of underrepresented groups": The study found that people of color account for just 1% of the city's sculptures, with women of color making up an even smaller 0.2%.

5. "Dominance of male subjects": Statues and sculptures dedicated to men account for over 20% of London's monuments, while those dedicated to "named people" (presumably men) make up over 79% of all statues.

6. "Royalty, military figures, politicians, writers, artists, designers, and actors are common male subjects": These groups are among the most frequently depicted in London's statues.

Overall, the study highlights a lack of diversity and representation in London's public sculptures, with women and underrepresented groups being significantly underrepresented.

► Relevant chunks:

[1] data on London's sculptures since 2017 as part of a major research project, which is funded in part by City Hall. Among the UK's largest cities, London has the highest percentage of sculptures dedicated to women. Why the toppled statue of slave trader Edward Colston is going on public display in Englandationwide, Queen Victoria, who reigned from 1819 -- 1901, is the most represented woman. Many of Britain's monuments have faced a reckoning since global protests against systemic racism and

[2] London (CNN)Just seeing Princes William and Harry in the same space is rare enough these days, let alone at an event they organized together. But that's where we found ourselves Thursday -- at the unveiling of the statue of Diana, Princess of Wales. The two were intimately involved in the design of the long-awaited artwork and the redesign of the Sunken Garden around it. According to Kensington Palace, this was one of Diana's "favourite locations" and it's where the young princes played growing

[3] (CNN)London has more public sculptures of animals than it does of women or people of color, a new study has revealed. Across London, 8% of public sculptures depict animals, while only 4% depict women, according to the study from the British charity Art UK, which was published on Thursday. People of color represent just 1% of the city's sculptures, with women of color accounting for 0.2%, it found. The figures sit in stark contrast to that of statues and sculptures dedicated to men, which

[4] found. The figures sit in stark contrast to that of statues and sculptures dedicated to men, which account for over 20% of the city's 1,500 monuments, and over 79% of all statues dedicated to "named people," the report said.Royalty, military figures, politicians, writers, artists, designers and actors are among the most commonly depicted male subjects, Art UK said. Read MoreThe group has been collecting data on London's sculptures since 2017 as part of a major research project, which is funded

[5] unveiling next month of a memorial to their mother -- Diana, Princess of Wales -- that we see them reunite once more.The brothers co-commissioned the statue from British sculptor Ian Rank-Broadley -- whose portrait of the Queen appears on all British coins -- to "allow all those who visit Kensington Palace to remember and celebrate her life and legacy." It's been given prime position in the palace's Sunken Garden, which Diana enjoyed when she lived there. That quote was from 2017 -- in the days

Me context

Enter your prompt:

Tell me about London's statues

Submit

► Responses:

London, the capital city of England, is renowned for its rich history and cultural heritage, which is reflected in its numerous iconic statues. Here are some of the most notable ones:

1. "The Duke of Wellington": Located in St. James's Park, this statue commemorates the victory of British forces over Napoleon at the Battle of Waterloo in 1815. The statue features General Arthur Wellesley (later known as the Duke of Wellington) riding a horse.

2. "Abraham Lincoln": Situated in Trafalgar Square, this statue was a gift from the American people to honor the 16th President of the United States. Designed by sculptor Daniel Chester French, it features Lincoln sitting in contemplation, holding his famous top hat and wearing his iconic suit.

3. "Queen Elizabeth II": Located in Whitehall Gardens, this statue celebrates the reign of Queen Elizabeth II (1926-2022). The sculpture was designed by sculptor Thomas Woolner and depicts the Queen standing majestically with her hands on a globe.

4. "Ivan the Terrible": Situated in Trafalgar Square, this imposing bronze statue is one of the most famous works by artist Ivan Kulbin. It depicts Ivan III (also known as Ivan the Great) holding an axe and wearing traditional Russian armor.

5. "Winston Churchill": Located in Parliament Square, this statue was designed by sculptor Ivor Roberts-Watts and features Churchill standing tall with his cigar and arm outstretched.

6. "Lord Nelson": Situated on Trafalgar Square's iconic column, this statue honors the British naval hero Horatio Nelson, who won a decisive victory over Napoleon at the Battle of Trafalgar in 1805.

7. "Charles Dickens": Located in Covent Garden, this statue was designed by sculptor Sir Francis Galton and features the famous author sitting on a bench, surrounded by books.

8. "William Shakespeare": Situated in Stratford-upon-Avon (not London), but often associated with the city, there is no official London statue dedicated to the Bard. However, there are several tributes and monuments throughout the city.

These statues showcase London's rich history, cultural heritage, and iconic figures who have shaped the city's identity.

Would you like me to tell you more about any specific statue or location?

Χωρίς context

“Tell me about Germany”

Enter your prompt:

Tell me about Germany

Submit

It seems I've been pulled into an unexpected conversation about Germany.

Okay, let me try to stay on topic and provide some general information about Germany:

Germany is a federal parliamentary republic located in Central Europe. It is the largest economy in Europe, the fourth-largest in the world, and has a population of around 83 million people (although the article mentions 60.4 million eligible to vote, which is likely due to EU voting regulations).

Germany is known for its rich history, culture, and contributions to science, technology, engineering, arts, and mathematics (STEAM). The country has produced famous philosophers like Immanuel Kant, Johann Wolfgang von Goethe, and Friedrich Nietzsche, as well as influential scientists like Albert Einstein.

German cuisine is famous for its sausages, beer, bread, and pastries. Some popular dishes include schnitzel, sauerbraten, and spätzle. The country is also known for its vibrant music scene, with genres like techno, house, and classical music originating from Germany.

Germany has a strong economy, with major industries like automotive (Volkswagen, Mercedes-Benz), aerospace (Airbus), and renewable energy (solar, wind power). Berlin is the capital city, and Munich is the largest city in Bavaria. The country has a well-developed public transportation system and is known for its efficient autobahn network.

Germany's history is complex and tumultuous, having experienced two devastating World Wars, as well as the Holocaust during WWII. However, the country has made significant efforts to come to terms with its past and promote reconciliation.

That's a brief overview of Germany!

► Relevant chunks:

[1] majority see it for what it really is: two footballing powerhouses, squads packed with talent and a history unlike any other in international football."For us as players, we don't tend to think too much about the history. It is about the here and now. What Germany are good at and the areas we can exploit. What we can do to hurt them," England player Jordan Henderson told reporters ahead of Tuesday."Things we can do in the game, that's where all our energy goes. We don't get too caught up on the

[2] Poland over their adherence to the EU's rule of law."We can manage this!There were also times when Merkel didn't go looking for a compromise, most notably in 2015 when she welcomed refugees fleeing conflicts in Syria and elsewhere. "Politically persecuted people have the right to asylum. We can be proud of the humanity of our Basic Law ... we also grant protection to all those who flee to us from wars," she told the German people that summer. "I put it simply: Germany is a strong country. The

[3] Germany's most populous state, since 2017. Voters cast their ballots in the federal parliamentary elections in Berlin, Germany. Long lines form in front of a polling station at a school in Friedrichshain.Backbook caused a brief sensation in German politics when she surged in the polls early in the campaign, prompting voters to wonder whether she could become the country's first Green chancellor.Some 60.4 million people age 18 and above are eligible to vote in this election, according to figures

[4] discussion. This needs a common approach but it is a discussion that I think has to be led," she added.Germany, much like Austria, has one of the lowest vaccination rates in western Europe, with 68.4% and 65.6% of their populations fully vaccinated respectively, according to the ECDC.Surging cases Germany continues to see a record-breaking number of cases, particularly in its eastern states. On Wednesday, the country recorded 446 Covid-19 related deaths -- its highest number of daily deaths in

[5] big thing. It's a real shame that there won't be as many German supporters in the ground because of the Covid restrictions. Wembley is [the very best] for German football fans," he adds. There are still those, mainly from the English tabloid press, who have looked to find unhealthy links between the football rivalry and the two countries' political past, including the two World Wars. But the majority see it for what it really is: two footballing powerhouses, squads packed with talent and a

Me context

Enter your prompt:

Tell me about Germany

Submit

► Response:

Germany is a federal parliamentary republic located in Central Europe, bordered by Denmark, Poland, Czech Republic, Austria, Switzerland, France, Belgium, and the Netherlands. With a population of approximately 83 million people, it is the most populous country in the European Union.

Geography and Climate:

Germany's terrain is varied, with mountains, forests, and plains covering about 60% of its land area. The Alps cover the southeastern part of the country, while the North Sea and Baltic Sea border Germany to the north. The climate ranges from temperate maritime in the northwest to continental in the southeast.

History:

Germany has a rich and complex history, with many empires rising and falling over the centuries. The Holy Roman Empire (10th-19th century) was a loose confederation of states that eventually fragmented into modern-day Germany. The unification of Germany under Otto von Bismarck's leadership in 1871 marked the beginning of a new era. However, this period also saw two devastating world wars, including World War I and World War II.

Economy:

Germany is one of the world's leading economies, with a strong focus on innovation, quality, and sustainability. The country has a highly developed manufacturing sector, particularly in the automotive and engineering industries. Germany is also home to many world-renowned companies like Mercedes-Benz, BMW, and Siemens.

Culture:

German culture is known for its rich heritage, vibrant arts scene, and unique festivals. Oktoberfest, a famous beer festival held annually in Munich, attracts millions of visitors worldwide. Germany has produced influential artists like Bach, Beethoven, and Wagner, as well as iconic writers like Goethe and Mann.

Places to Visit:

Some popular destinations in Germany include:

1. Berlin: The capital city is known for its vibrant arts scene, historic landmarks like the Brandenburg Gate, and museums like the Pergamon Museum.
2. Munich: Famous for Oktoberfest and home to the iconic Neuschwanstein Castle, a fairytale-like palace built by King Ludwig II.
3. Hamburg: A major port city with a rich maritime history, colorful buildings, and vibrant nightlife.
4. Cologne: Known for its stunning cathedral, Roman-Germanic Museum, and lively music scene.
5. The Black Forest: A picturesque region famous for its beautiful forests, traditional villages, and delicious local cuisine.

Cuisine:

Traditional German dishes include sausages (e.g., Bratwurst), schnitzel, sauerbraten, potato dumplings (Kartoffelpüree), and apple strudel. Germany is also famous for its beer culture, with many breweries offering tours and tastings.

Language:

The official language of Germany is German (Deutsch), but many people also speak English, especially in tourist areas.

Χωρίς context

Το συμπέρασμα στο οποίο καταλήγουμε, είναι ότι σε περιπτώσεις όπου η αναζήτηση αναφέρεται σε ένα πολύ συγκεκριμένο θέμα ή πρόσωπο, πχ “Tell me some facts about Wimbledon” ή “Tell me some facts about Tom Felton” η RAG αρχιτεκτονική βοηθάει πολύ στο να έχουμε πιο στοχευμένες απαντήσεις επάνω στο θέμα. Σε ερωτήσεις γενικότερης κλίμακας οι απαντήσεις είναι και πάλι πολύ εύστοχες και σίγουρα βοηθάει και πάλι η αρχιτεκτονική RAG, οι διαφορές όμως με context και χωρίς είναι λιγότερο αισθητές.