**🚀 Updated Instructions for Running the Model**

This model allows users to ask questions based on a **PDF document** using **TF-IDF and cosine similarity** to retrieve the most relevant answer.

**1️ Install Required Dependencies**

Before running the model, install the required Python libraries:

pip install PyPDF2 numpy joblib scikit-learn

**2️ Extract Text from a PDF**

Run the following code to extract text from your uploaded **PDF file**:

from PyPDF2 import PdfReader

# Path to the uploaded PDF

pdf\_path = "your\_pdf\_file.pdf" # Replace with the actual file name

# Extract text from the PDF

reader = PdfReader(pdf\_path)

text = "\n".join([page.extract\_text() for page in reader.pages if page.extract\_text()])

# Save extracted text to a file

with open("book\_text.txt", "w", encoding="utf-8") as text\_file:

text\_file.write(text)

print("✅ PDF text extracted and saved to 'book\_text.txt'")

**3️ Preprocess and Chunk the Text**

Once the text is extracted, clean it and split it into smaller **chunks** for better searchability.

import re

import numpy as np

# Function to clean extracted text

def clean\_text(text):

text = re.sub(r'\n+', '\n', text) # Normalize multiple new lines

text = re.sub(r'[^\\x00-\\x7F]+', ' ', text) # Remove non-ASCII characters

text = re.sub(r'\s+', ' ', text) # Remove extra spaces

return text.strip()

# Load extracted text

with open("book\_text.txt", "r", encoding="utf-8") as file:

book\_text = file.read()

# Clean the text

cleaned\_text = clean\_text(book\_text)

# Split text into chunks (500 words per chunk)

chunk\_size = 500

words = cleaned\_text.split()

chunks = [" ".join(words[i:i + chunk\_size]) for i in range(0, len(words), chunk\_size)]

# Save chunks for retrieval

np.save("chunks.npy", np.array(chunks))

print("✅ Text has been split into chunks and saved as 'chunks.npy'")

**4️ Create and Train the TF-IDF Model**

Now, we transform the text chunks into **TF-IDF vectors** to allow efficient question-answering.

from sklearn.feature\_extraction.text import TfidfVectorizer

import joblib

# Load the saved text chunks

chunks = np.load("chunks.npy", allow\_pickle=True)

# Create a TF-IDF Vectorizer

vectorizer = TfidfVectorizer(stop\_words='english', max\_features=5000)

# Fit and transform the text chunks into TF-IDF vectors

tfidf\_matrix = vectorizer.fit\_transform(chunks)

# Save the vectorized model and text chunks for retrieval

joblib.dump(vectorizer, "vectorizer.pkl")

np.save("tfidf\_matrix.npy", tfidf\_matrix.toarray())

print("✅ TF-IDF model is now ready for question answering!")

**5️ Run the Question Answering Model**

Now, you can ask questions based on the **uploaded PDF**.

import numpy as np

import joblib

from sklearn.metrics.pairwise import cosine\_similarity

# Load the saved TF-IDF model and text chunks

vectorizer = joblib.load("vectorizer.pkl")

tfidf\_matrix = np.load("tfidf\_matrix.npy")

chunks = np.load("chunks.npy", allow\_pickle=True)

print("💬 Type your questions below. Type 'exit' to stop.\n")

while True:

# Ask a question

question = input("Ask a question: ")

# Stop if the user types "exit"

if question.lower() == "exit":

print("🚪 Exiting the Q&A system. Have a great day! 😊")

break

# Convert the question into a TF-IDF vector

question\_vector = vectorizer.transform([question])

# Compute cosine similarity between the question and book chunks

similarities = cosine\_similarity(question\_vector, tfidf\_matrix)

# Find the most relevant chunk

best\_match\_idx = np.argmax(similarities)

# Retrieve the most relevant text from the book

best\_answer = chunks[best\_match\_idx]

print("\n📖 Answer from the book:\n")

print(best\_answer)

print("\n" + "-"\*80 + "\n")

**6️ Sample Input & Expected Output**

Once the model is running, here’s an example interaction:

**Example Input**

Ask a question: What is a linked list?

**Expected Output**

📖 Answer from the book:

Linked lists are data structures that consist of nodes, where each node contains data and a reference to the next node.

**📌 How to Upload to GitHub**

**1️ Initialize a Git Repository**

Run the following command inside your project folder:

git init

**2️ Add the Files**

git add .

**3️ Commit the Changes**

git commit -m "Added TF-IDF Question Answering Model"

**4️ Create a GitHub Repository**

* Go to [GitHub](https://github.com/)
* Click on **"New Repository"**
* Name your repository (e.g., QA\_Model\_TFIDF)
* Copy the repository URL (e.g., https://github.com/yourusername/QA\_Model\_TFIDF.git)