

★ Get unlimited access to the best of Medium for less than \$1/week. [Become a member](#)



How to Chat with Your PDF using Python & Llama2



Woyera · [Follow](#)

5 min read · Jul 31, 2023



Listen



Share



More

With the recent release of Meta's Large Language Model(LLM) **Llama-2**, the possibilities seem endless.

What if you could chat with a document, extracting answers and insights in real-time?

Well with Llama2, you can have your own chatbot that engages in conversations, understands your queries/questions, and responds with accurate information.



Credit: VentureBeat made with Midjourney

In this article, we'll reveal how to create your very own chatbot using Python and Meta's Llama2 model.

[Open in app](#) ↗



Search

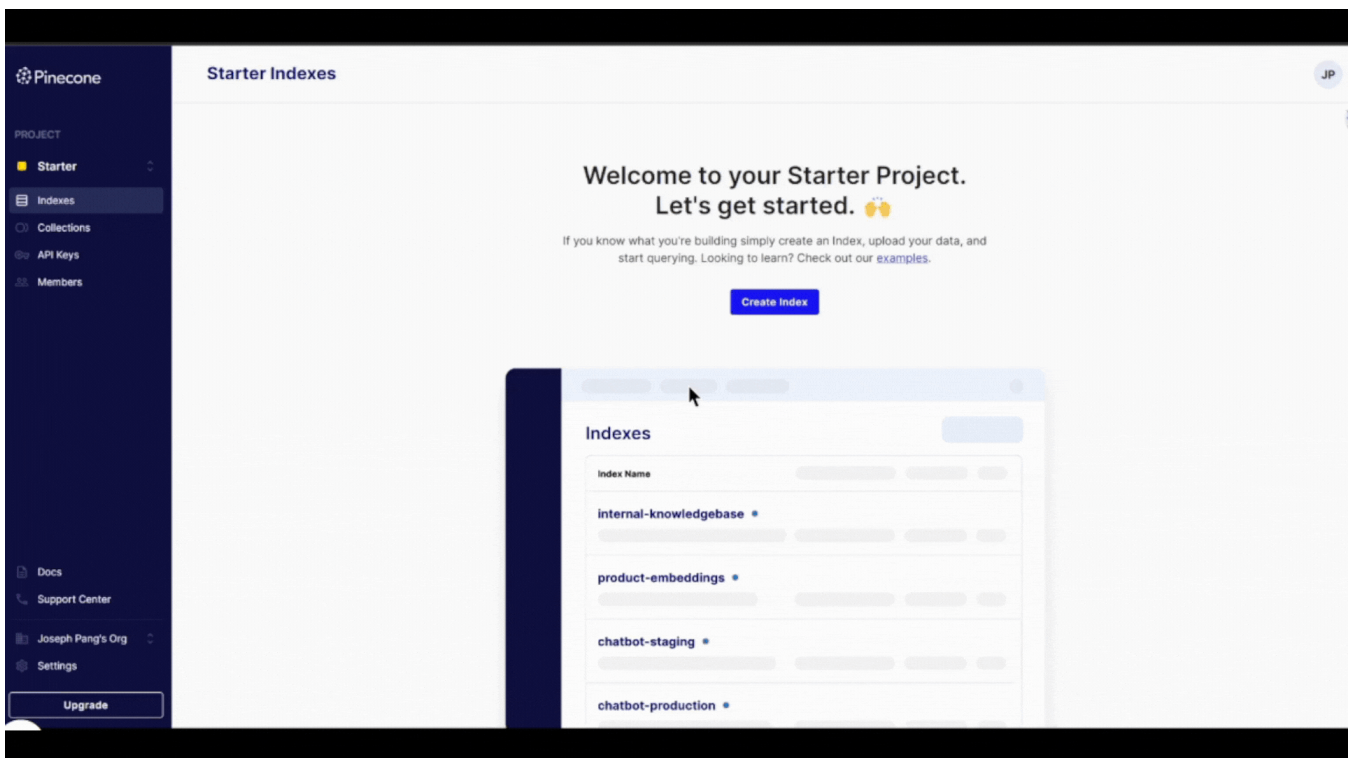


What You Will Need:

1. **Pinecone API Key:** The Pinecone vector database can store vector embeddings of documents or conversation history, allowing the chatbot to retrieve relevant responses based on the user's input.
2. **Streamlit Replicate API Key:** This is how we will apply the Llama2 model for our chatbot.
3. **Python:** The programming language we will be using to build our chatbot. (*Make sure to download Python versions 3.8 or higher*)

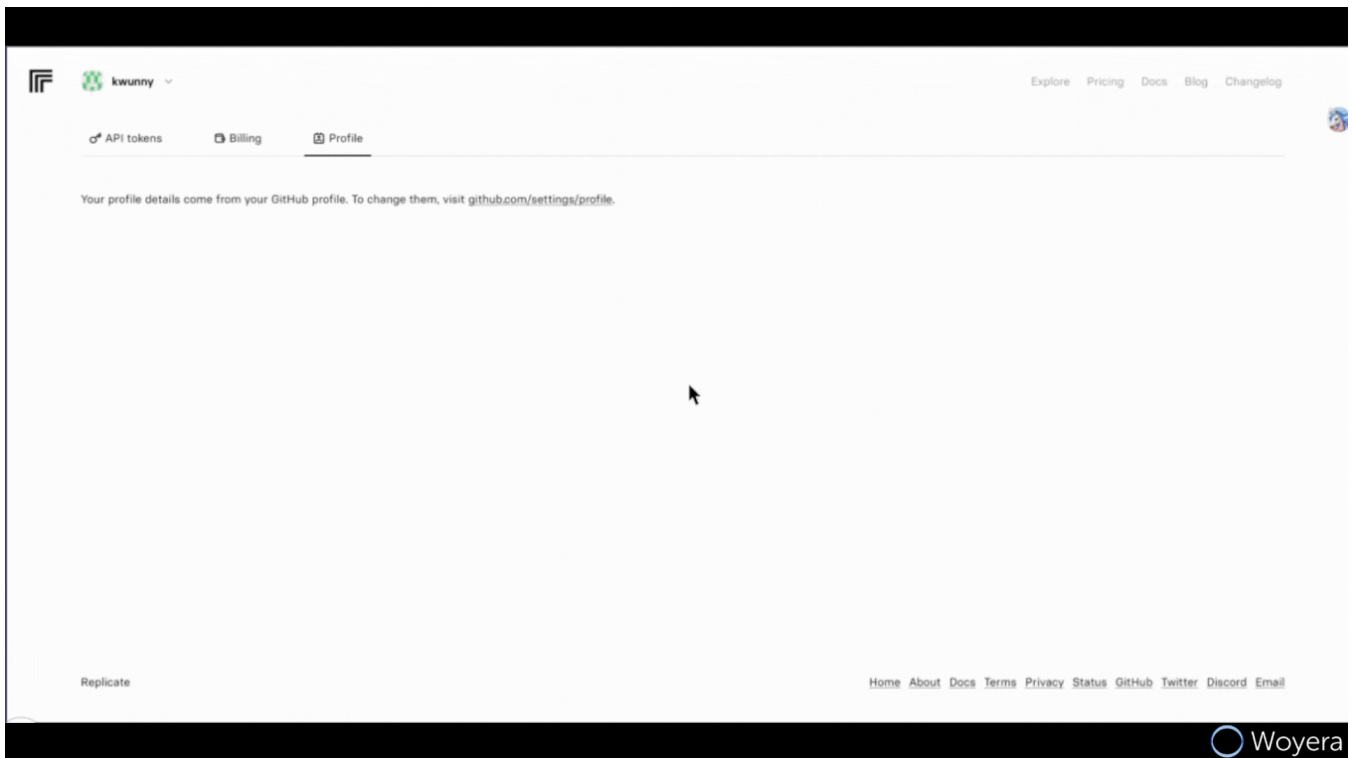
Steps for Pinecone:

1. Sign up for an account on the [Pinecone](#) website.
2. Once you are signed up and logged in, on the left side navigation menu click “API Keys”.
3. Copy the API key displayed on the screen (we will use this key later).
4. Now, go back to the “Indexes” tab and create a new index.
5. Name it whatever you want and make the dimensions 768.
6. Create the index and copy the environment of the index, we’ll need it for later.



Steps for how to get Replicate API Key:

1. Go to the [Replicate](#) website and sign up.
2. Once you are signed up and logged in, navigate to this link to see your API Key:
<https://replicate.com/account/api-tokens>
3. You should see your own key on the page. Copy the key and save it for later.



Lets start Building our Chatbot!

Before diving into the code, ensure that you have the required libraries installed. You can install them using the following commands into your terminal:

```
pip install pinecone-client langchain
```

Step 1: Initializing the Environment

1. Make a python file ex. *app.py* and open it with your code editing application of choice.
2. Next, we need to set up the environment with the necessary libraries and tokens. The code snippet below imports the libraries and initializes the Replicate API and Pinecone, enabling us to access their functionalities. Paste your previously copied API keys in the designated spots.

```
import os
import sys
import pinecone
from langchain.llms import Replicate
from langchain.vectorstores import Pinecone
from langchain.text_splitter import CharacterTextSplitter
from langchain.document_loaders import PyPDFLoader
```

```
from langchain.embeddings import HuggingFaceEmbeddings
from langchain.chains import ConversationalRetrievalChain

# Replicate API token
os.environ['REPLICATE_API_TOKEN'] = "YOUR REPLICATE API HERE"

# Initialize Pinecone
pinecone.init(api_key='YOUR PINECONE API HERE', environment='YOUR ENVIRONMENT H
```

Step 2: Preparing the Data

Next, we need data to build our chatbot.

In this example, we load a PDF document in the same directory as the python application and prepare it for processing by splitting it into smaller chunks using the `CharacterTextSplitter`.

```
# Load and preprocess the PDF document
loader = PyPDFLoader('./NAME_OF_YOUR_PDF_HERE.pdf')
documents = loader.load()

# Split the documents into smaller chunks for processing
text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=0)
texts = text_splitter.split_documents(documents)
```

Step 3: Leveraging HuggingFace Embeddings

HuggingFace's embeddings play a crucial role in transforming the text data into dense numerical vectors, a fundamental step in building our intelligent chatbot.

```
# Use HuggingFace embeddings for transforming text into numerical vectors
embeddings = HuggingFaceEmbeddings()
```

Step 4: Creating the Pinecone Vector Database

With our text data converted into embeddings, we can proceed to create the Pinecone vector database.

This database allows for efficient similarity search and retrieval of vectors, which is essential for our chatbot's conversational capabilities.

```
# Set up the Pinecone vector database
index_name = "NAME OF YOUR DATABASE(PINECONE INDEX THAT YOU CREATED)"
index = pinecone.Index(index_name)
vectordb = Pinecone.from_documents(texts, embeddings, index_name=index_name)
```

Step 5: Integrating Replicate LLama2 Model

Llama2, with its powerful language generation model, is the heart of our chatbot.

We integrate this model to enable our chatbot to understand user queries and generate intelligent responses.

```
# Initialize Replicate Llama2 Model
llm = Replicate(
    model="a16z-infra/llama13b-v2-chat:df7690f1994d94e96ad9d568eac121aecf50684a
    input={"temperature": 0.75, "max_length": 3000}
)
```

Step 6: Building the Conversational Retrieval Chain

To create a dynamic and interactive chatbot, we construct the ConversationalRetrievalChain by combining Llama2 LLM and the Pinecone vector database.

This chain enables the chatbot to retrieve relevant responses based on user queries and the chat history.

```
# Set up the Conversational Retrieval Chain
qa_chain = ConversationalRetrievalChain.from_llm(
    llm,
    vectordb.as_retriever(search_kwargs={'k': 2}),
    return_source_documents=True
)
```

Step 7: Chatting with the Chatbot

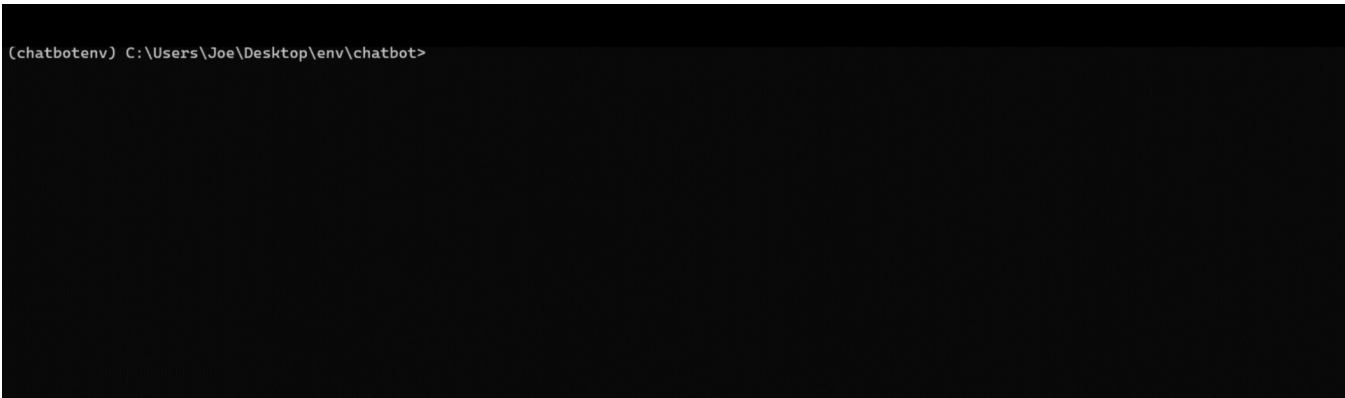
The code snippet below initiates an infinite loop, where the user inputs queries and the chatbot responds with intelligent answers based on the chat history.

```
# Start chatting with the chatbot
chat_history = []
while True:
    query = input('Prompt: ')
    if query.lower() in ["exit", "quit", "q"]:
        print('Exiting')
        sys.exit()
    result = qa_chain({'question': query, 'chat_history': chat_history})
    print('Answer: ' + result['answer'] + '\n')
    chat_history.append((query, result['answer']))
```

Final Step: Test the Chatbot!

By this point, all of your code should be put together and you should now be able to chat with your PDF document.

1. Now simply run the command to execute the python application into your terminal ex. *python app.py*
2. Test the chatbot!



Congratulations! You've now built an intelligent chatbot for your documents using the Llama2 model. Have fun!

Now if you find yourself in need of help on any of these steps, feel free to [book a call with us at www.woyera.com](https://www.woyera.com)

[Follow](#)

Written by Woyera

329 Followers

We build custom, secure, robust chat bots for security & privacy minded enterprises

More from Woyera



Woyera

How to Use Llama 2 with an API on GCP (Vertex AI) to Power Your AI Apps

Ever wanted to use Google Cloud Platform (GCP) to deploy your Llama 2 LLM?

9 min read · Sep 25, 2023



224



4





Woyera

Pinecone vs. Chroma: The Pros and Cons

In recent years, vector databases have gained significant attention for their ability to efficiently store and retrieve high-dimensional...

4 min read · Jul 21, 2023

92

...



Woyera

How to Chat with CSV Files Using Llama 2

Have a bunch of CSV files that require manual review?

4 min read · Aug 16, 2023



74



1



Woyera

How to switch from the OpenAI API to Llama 2

Are you considering migrating from OpenAI's API, but unsure of the benefits and drawbacks?

5 min read · Sep 21, 2023



277

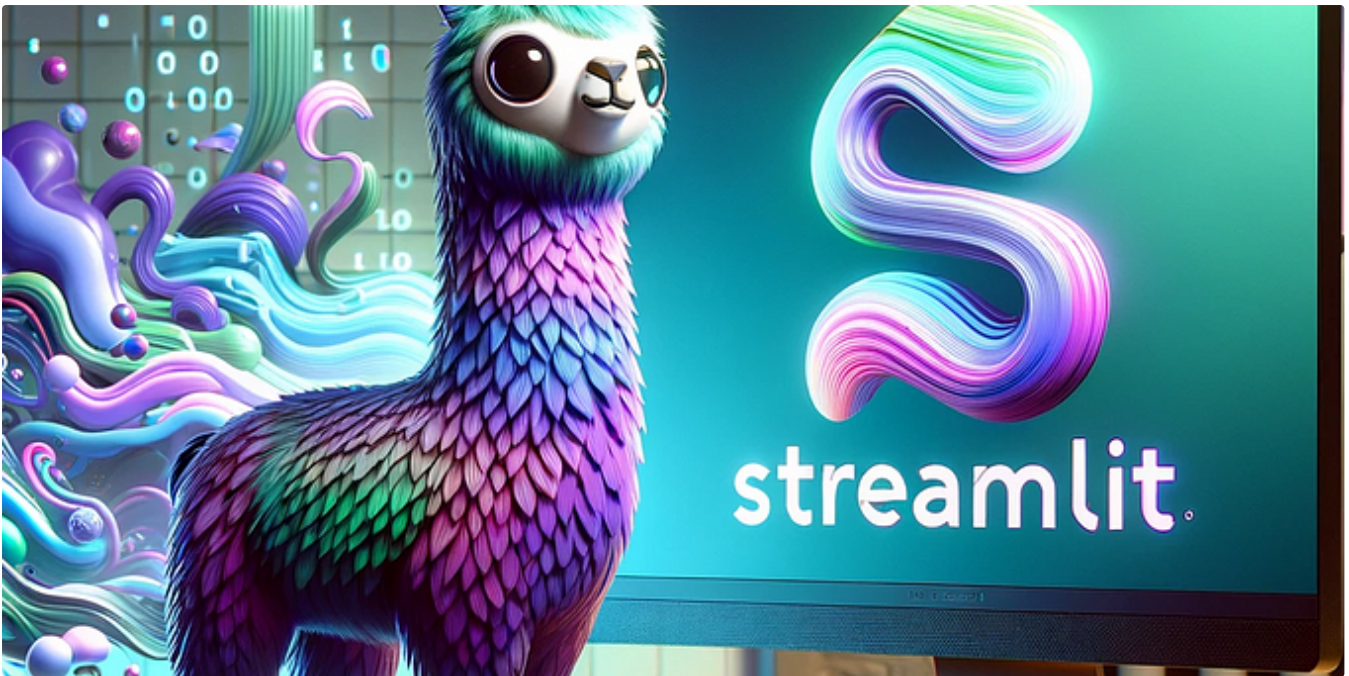


2



See all from Woyera

Recommended from Medium



Gaurav Jakhar

Building an AI-Powered 'Chat with PDF' App with Streamlit, LangChain, FAISS and LLaMA2

Introduction:

5 min read · Nov 4, 2023



59



localhost:8501

Query PDF Source

Enter Query

what is anesthesia consent ?

Generate

Anesthesia consent is a form that a patient signs to acknowledge that they have been informed about the risks and benefits of anesthesia before a medical procedure. It states that the patient understands the potential complications and agrees to receive anesthesia services in order to undergo the operation or procedure. The form also lists specific types of anesthesia that may be used, such as general anesthesia or spinal/epidural anesthesia, and outlines the expected effects and risks associated with each type.



Diptiman Raichaudhuri

Rapid Q&A on multiple PDFs using langchain and chromadb as local disk vector store

Disclosure: All opinions expressed in this article are my own, and represent no one but myself and not those of my current or any previous...

11 min read · Sep 25, 2023



90



2

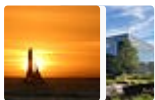


Lists



Staff Picks

570 stories · 705 saves



Stories to Help You Level-Up at Work

19 stories · 452 saves



Self-Improvement 101

20 stories · 1289 saves



Productivity 101

20 stories · 1179 saves



Richard Song in Epsilla

Build A ChatBot That Runs Purely On Your Local Machine (Using Llama 2 + Epsilla + LangChain +...

Since late 2022, ChatGPT has gained significant attention, capturing the interest of people across the globe. The technology behind it...

7 min read · Sep 28, 2023



26



1



Madhav Thaker

(Part 1) Build your own RAG with Mistral-7B and LangChain

LLMs have taken the world by storm and rightfully so. They help you with every day tasks like building a coding project, creating a recipe...

14 min read · Nov 14, 2023

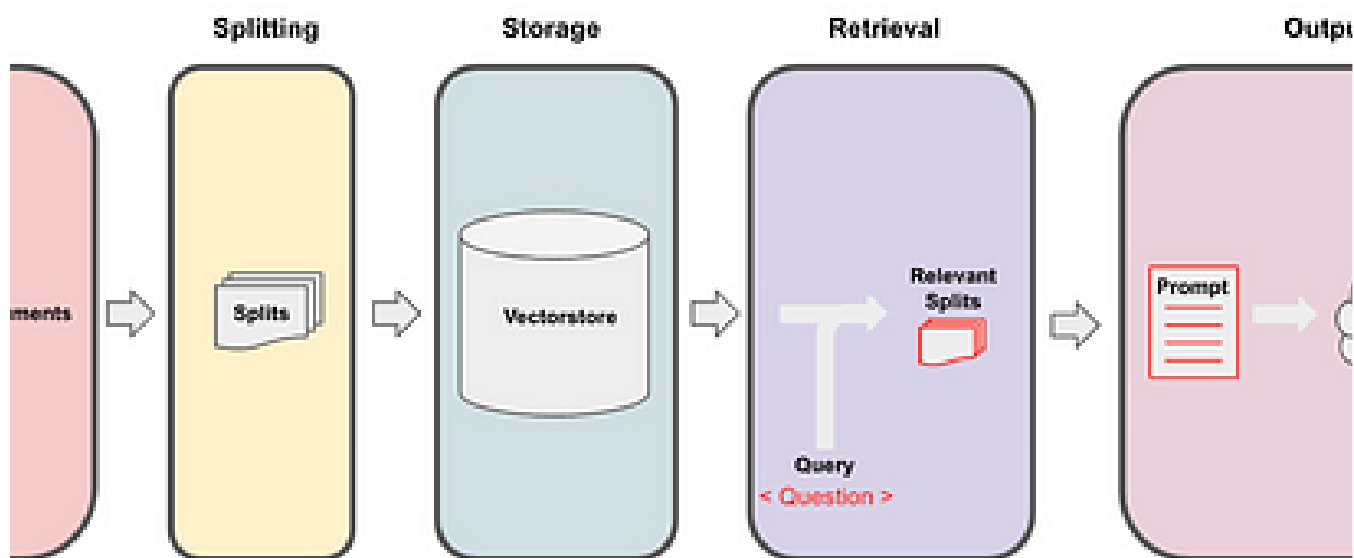


1.8K



11





 Onkar Mishra

Using langchain for Question Answering on own data

Step-by-step guide to using langchain to chat with own data

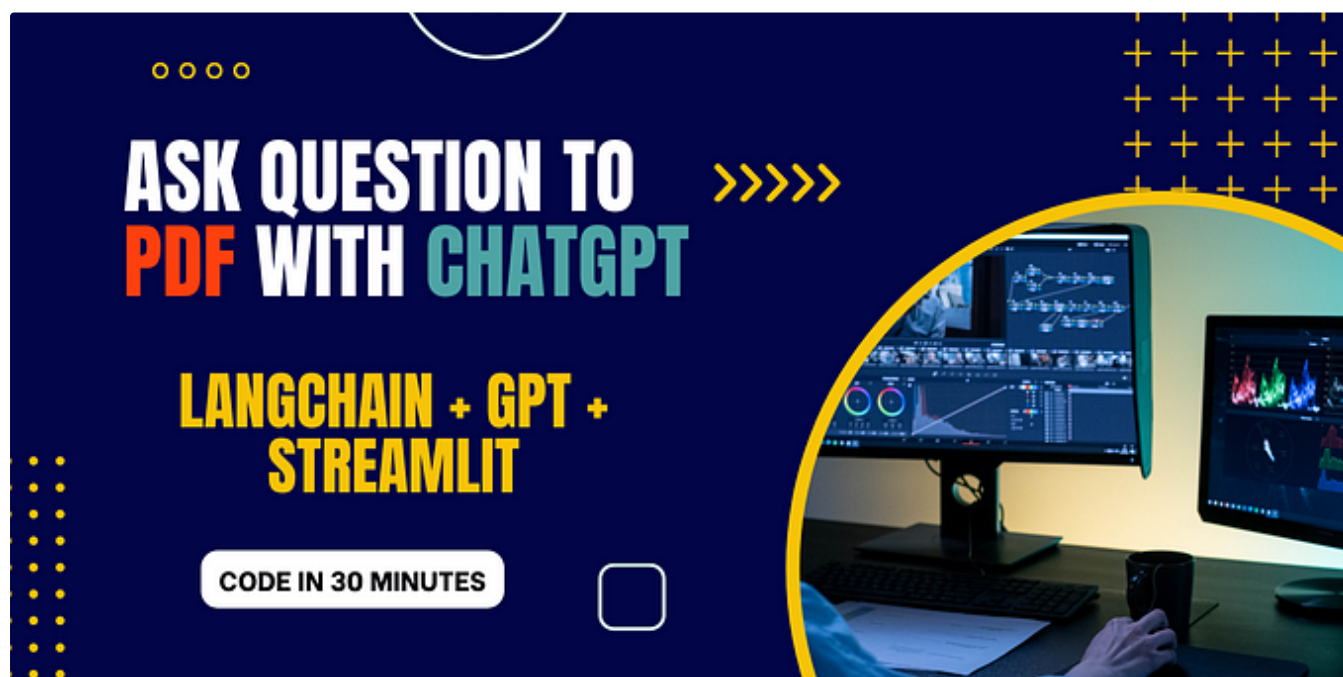
23 min read · Aug 7, 2023



1.3K



16



 Livia Ellen in Python in Plain English

Create Your Own PDF Question Answering System with OpenAI GPT, LangChain, and Streamlit

Are you interested in building a chatbot that can read your PDF documents and answer questions about their content? In this tutorial, I'll...

★ · 4 min read · Aug 20, 2023



20



See more recommendations