Are you amazed by the power of chat GPT and Bart? Guys with little bit of Python programming, you can actually make your own chatcode for custom use cases like this, this or this. Do you want to know how? Well, that's exactly what we are going to discuss in today's video. And the best part is going to be absolutely free and you guys can follow me along. So whether you are a student looking for our resume pointer or a professional wanting to have fun, this video is for you. So there are essentially three things we are going to do in the next 10 minutes or so. First up, we'll build a simple yet powerful version of our chatbot in Python powered by the Google Power DPI. This first step is absolutely beginner friendly. Then from there, we are going to step up and build a more sophisticated chatbot for a custom use case which is tech summarization again using Bart API. Why are we using Google Bart API and no chat GPT you may ask? Well, the answer is Bart API is absolutely free to access, especially for learning purposes. So all you guys can follow me along and replicate it on your end. But hey, I'll still be discussing chat GPT API in the third part of this video and briefly cover how you can build a chatbot powered by a chat GPT for the same use case which is tech summarization. On this note, let's begin. To get started, you need two things. Number one, an IDE for Python coding. I'll be using Google Collab in this demonstration which I find very beginner friendly and easy to use. Second is the API keys for Google Bart and chat GPT. I'll be showing you the steps to retrieve these keys later in the video when we get there. At this point, let's jump onto the screen recording for our hands-on. So guys, this is our project folder on Google Drive. For tech summarization, I've kept these couple of PDF files over here. The first one is the famous book called The Art of War. And the second one is a research paper from Google titled Attention is All We Need. I'm sure you would have heard of this. The third file here is the code file. So let's fire it up to get going on building our chatbots. And of course, I'll share a link to this drive folder in the description part of this video so that you guys can access both of these books and the code file. So let's get started. All right, now I'm on Google Collab over here to the left. You see the sections of this code file. As I said, a while back will be first building a simpler version of our chatboard followed by a custom chatboard for the specific use case powered by Bart. Towards the end, we discuss a charge-upity API too. On this note, let's begin our code review. All right, let's begin by setting up our environment first. Over here, we are first installing the Bart API, Bython package. Bart API returns response of Google Bart through API. For your further reading, I'll put a link to the documentation in the comment section. So you may read that through. Next up, we are importing these essential libraries. All right, as third step here, we need to configure for Bart access. For this, we need to obtain this ID from your browser cookies and here are the steps to do it. First go to your Bart window, do a right click, click on inspect. Then within this window on the right click on application cookies. And within cookies, you will find bar.google.com, which is the URL. And within this table, you will find your ID, which is over here, over here. And then all you have to do is copy this. Remember, this is your personal ID. So handle it with care. Or at once you have obtained this ID, replace it over here in this code cell and run it. Now that we have everything in place, let's start with the simplest version of our chatbot. We'll use the Bart API to get answers to our questions. For instance, in this example, we are asking the question, why is the sky blue to our chatbot and it will provide us with the answer. Let's take a look. Addemi, we got the answer to our question from Google Bard. As we all know, that Bard is connected with internet, unlike chatgbd, which is trained on data till 2021. So you may even query on the recent events, like what is the current status of Azure Series 2023? Let's see what we get. Addemi, we are getting the correct response. Australia is leading the 2023 Azure Series right now to east to zero. All right, so this is the simple chatbot, part by Google Bard, allowing you to perform prompting from within your Jupyter notebook. Cool, right? Guys, now when we have successfully configured and tested our Bard Bard chatbot, skies are limitarily on the kind of customizations we may build on top of this using Python. So next up, let's jump on to building a custom chatbot. In this section, guys, we are going to build a text summarizer that takes a PDF document as input, which could be our research paper or a book, and summarize it into a handful of words. So let's do this. For building our text summarizer, the first thing we need to do is to install this Python library called Pi PDF. Pi PDF is a free and open source Python library that helps us retrieve text from PDF document, which is something essential for summarizing PDF documents. Okay, once it's installed, then we import it. Now, as I showed you a while back, I have gathered these couple of PDF documents for this demonstration. Link to this drive folder is in the description part below, guys, I sort of told you a while back. All right, coming back to Colab, to access our PDF documents from within the Google drive, we need to first mount Google Drive to Colab. So that's what we do over here. We have followed these steps. As next step, I'm declaring the directory to our drive folder and file names. Here, this attention is all you need is the paper that will try summarizing first. Guys, we all know Bard have a limit on the input prompt length, which is around 4,000 characters. So we can't send the entire PDF text into a single query for summarization. Rather, what we can do is to find out a way of sending this data into parts. A simple way of doing this is to divide the whole text of the PDF on the basis of pages, which we are following in our case. However, a better strategy is to split the text on the basis of paragraphs, but then it will increase the number of API calls that will have to make and thereby increasing the overall compute time. So we are going to go with the simpler approach over here. So here in this next code cell, we are calling the PDF reader module from Pi PDF to read the text from our PDF document one page at a time and store it in this empty list called text. Let's run it. This next step is sort of optional for you. Here we have this function called join elements that merges text from multiple pages stored in our list as single element. See, the idea here is this. Bard has prompt length limits, but it's still a sizeable 4,000 characters at this point. So again, for the sake of reducing the number of API calls, what we can do is to actually merge multiple pages of text into one. And over here in this code, I'm actually merging three pages of text into one and storing it into this new list, which I'm calling new text. And guys, if you have tried an alternate strategy for handling the text, do share it in the comment section. I would be great hearing from you. Next up, we define this get completion function to fetch response from Bard for our specific prompt. And now is the time when we start prompting for our specific use case, which is text summarization. This is a matter of experiment really guys to figure out the best prompt for you. For now, you can use this prompt, which has worked well for me. I highly encourage you to try out your own version of prompt and do share those in the comment section below, along with brief on the result you go from it. This cell might take a few minutes to run, so I'll sort of fast forward the video. So guys, this is the summary of our attention is all you need paper. Finally, in this last code cell, we get to save our summary that we have obtained in a text file. And once we do that, it gets stored in the Google Drive directly folder that we have over here. And with this guys, our text summarizer is now ready. To summarize a different book, all you have to do is to put it in your drive folder, like I have this out of work book over here, and declare the file name to that book over here in this code cell. Trust me guys, that's really not the end of route for you. There are these bunch of practice use cases I'm putting up for you as a challenge. Do try to solve them and share your code file as a link in the comment section of this video, along with a brief on the use case you prepared the chatboard for. Alright, in this last section of our code, we are doing everything we discussed so far in this video, but with chat GPT API now. I'm keeping this code here for your ready reference if you plan on building a chat GPT powered chatboard. For sure, you will get better results with this chat GPT API as compared to BARD, but for now, the chat GPT API is not freely available. So BARD is the next best thing really. But if you can somehow manage to get access to chat GPT API or you have a paid plan with OpenAI, go ahead and use this code for your use cases. If you get stuck, do ping us in the comment section and we'll get back to you. Guys, pat yourself on the bat for raging till here. In this tutorial, you learned how to build a chat powered by BARD and chat GPT starting from scratch. Remember, building a chatboard opens up endless possibilities for creating innovative applications. Whether it's answering questions or summarizing text, chatbots are here to assist us in a variety of ways. And if you found this video helpful, make sure you give it a thumbs up and subscribe to our channel for more exciting tutorials on data science and ready-made API. Also, do not forget to check out the code that we have put for you in the description part of this video. Thanks for watching and I'll see you in the next video. Happy coding to you. Bye!