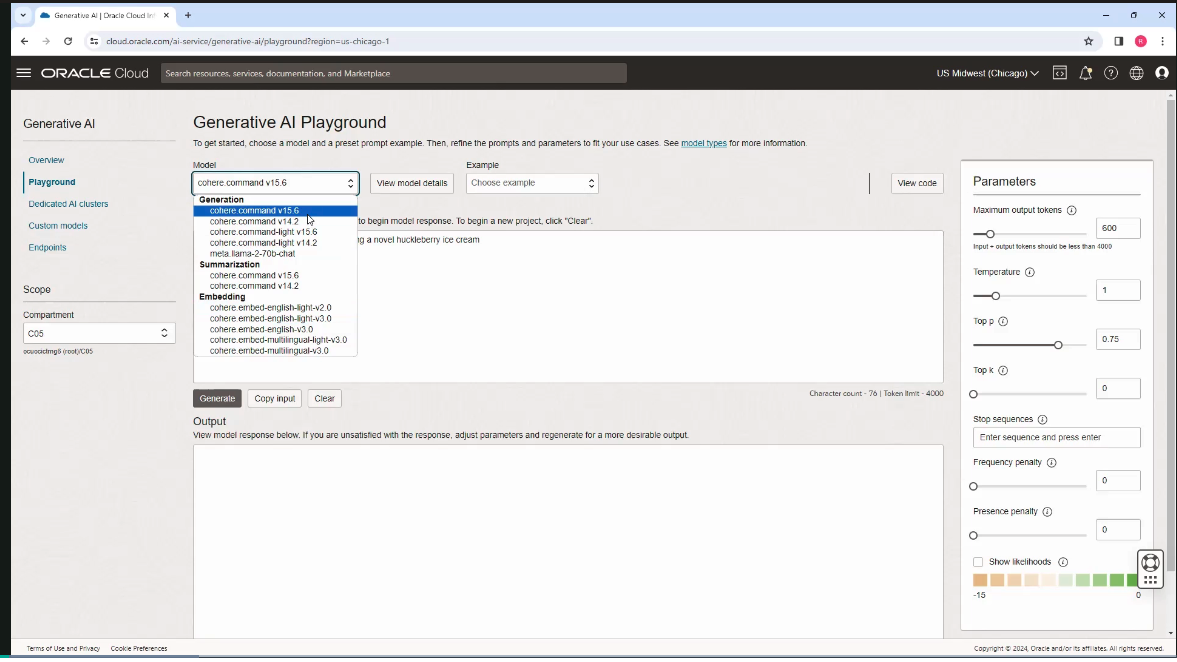
Demo : Generation Models - There are 3 different senarios

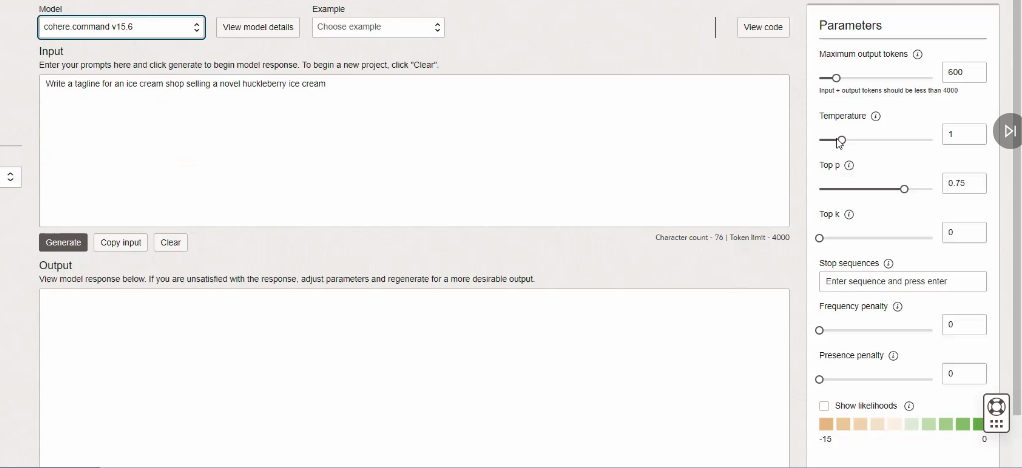
1. Text generation

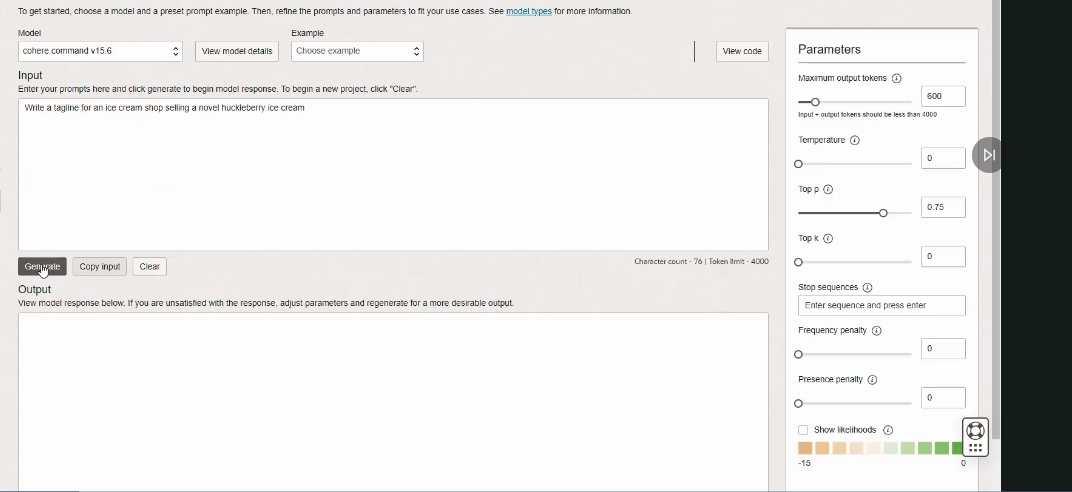
2. Data extraction

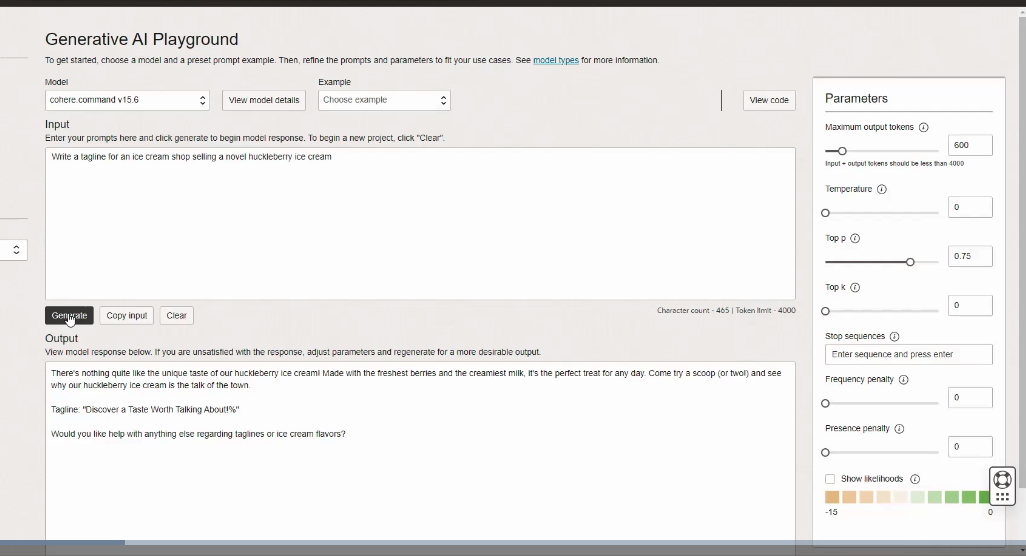
3. Text classification

1. First go to the palyground.







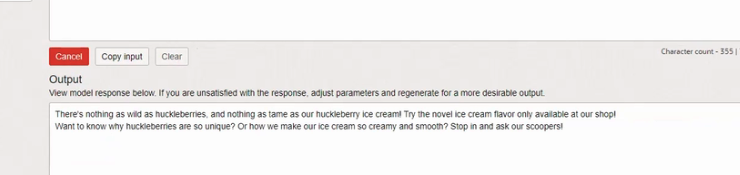


- Temperature is a parameter that controls the creativity or randomness of the text generated. A higher temperature results in more diverse and creative output, while a lower temperature makes it more deterministic.

- Temperature is affecting the probability distribution over the possible tokens at each step of the generation process.

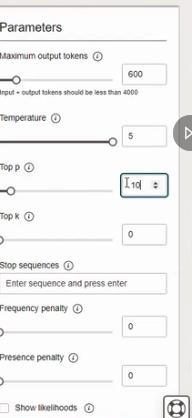
If temperature changed to 5. The output will be different as compared to zero temperature.



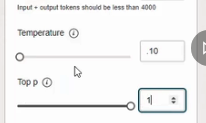


Another Important parameter is Top p which is an alternative to temperature.

For example, if we set top p as 0.10, then the model considers only the token that make up the top 10% of the probability of the next token.



In new tab change the temperature and top p values and keep the same prompt.

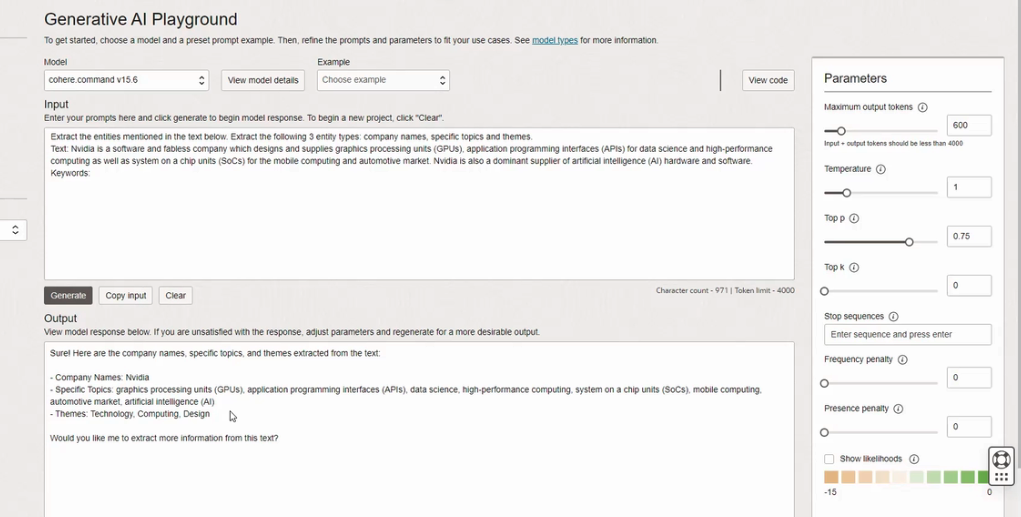


It will generate different responses for both.

**Both temperature and top p are powerful tools for controlling the behavior of large language models. You can use them independently or together when making these API calls.**

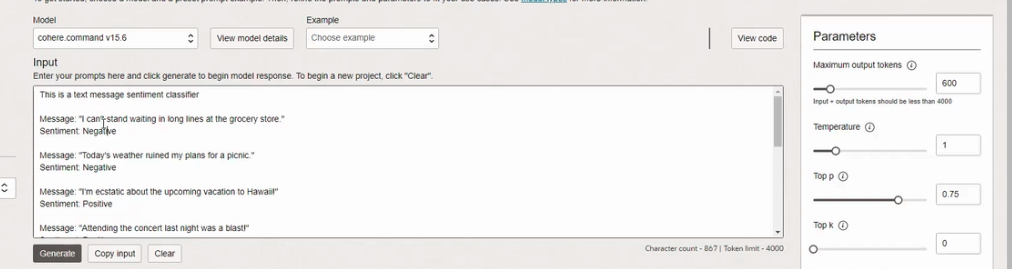
2. Data extraction: action. So in this case, asking the model-- providing the model a brief text. And asking it to extract the entities mentioned in the text below.

And specifically, asking for three entity types, company name, specific topics, and themes.

****

**You can use these generation models not only to generate text, but also extract data from text.**

3. Text Classification: Here, we are creating text classifier with the generation models. And we are providing a description of the task and providing a few examples.



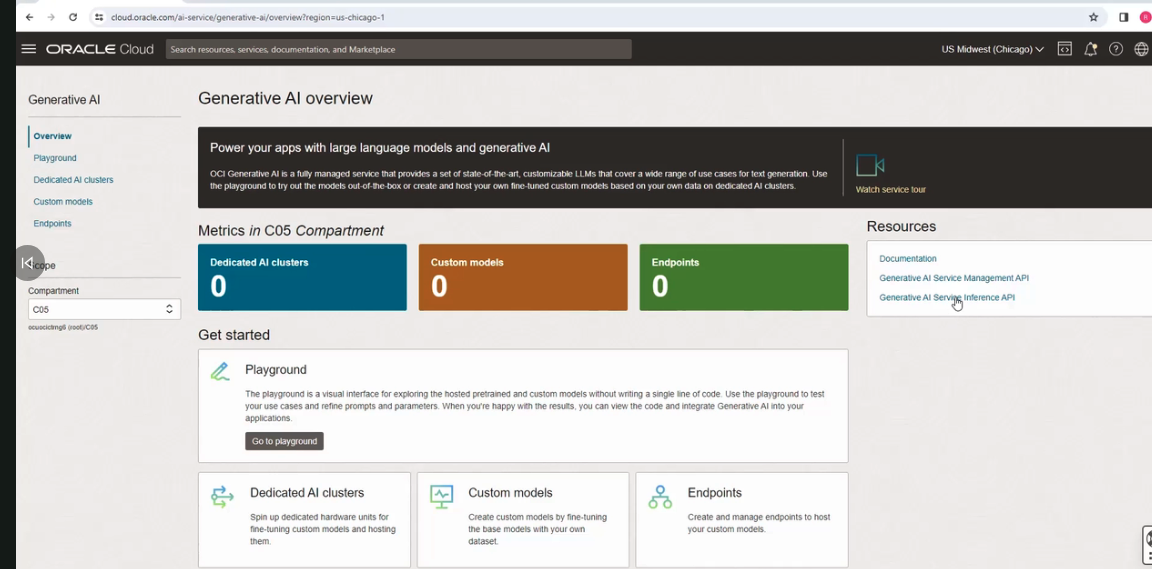
- Here showing the API how to classify the sentiment of text messages. The sentiment expresses the overall feeling or expression in the text.

- For the 10th one, we just have the text message. And we are asking the model to classify this text and provide us with the sentiment, whether it's positive or negative or neutral.

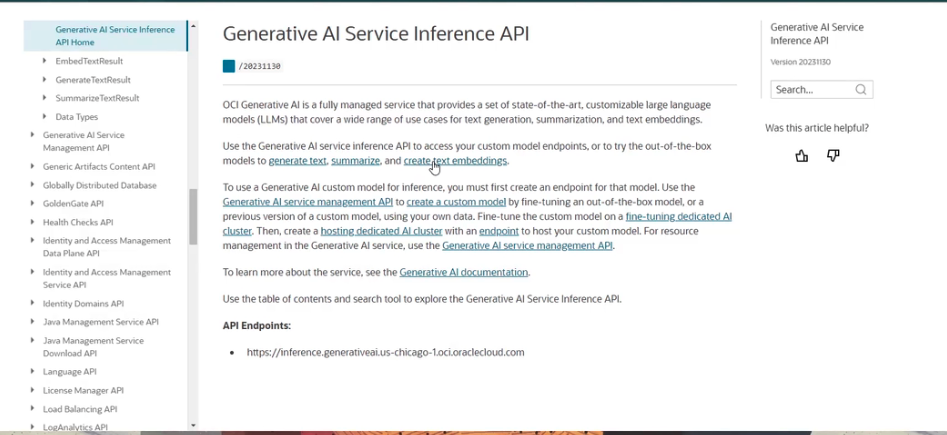




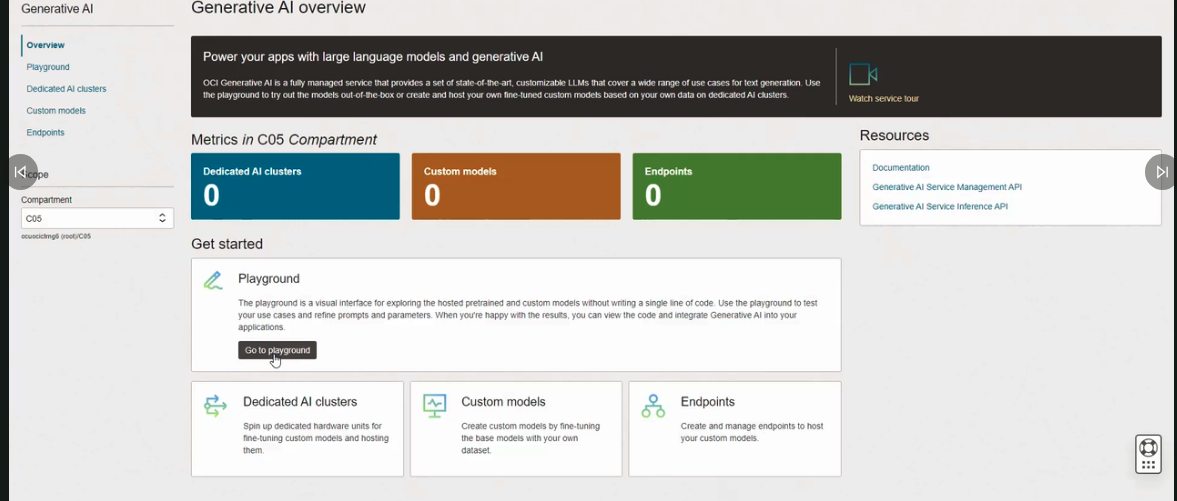




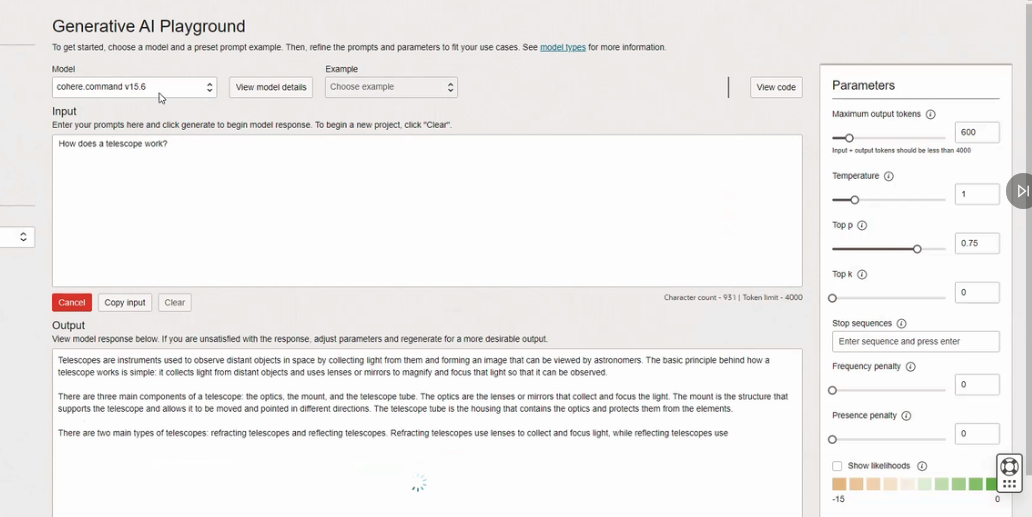
click it, you will get the documentation:



Go to playgroung.

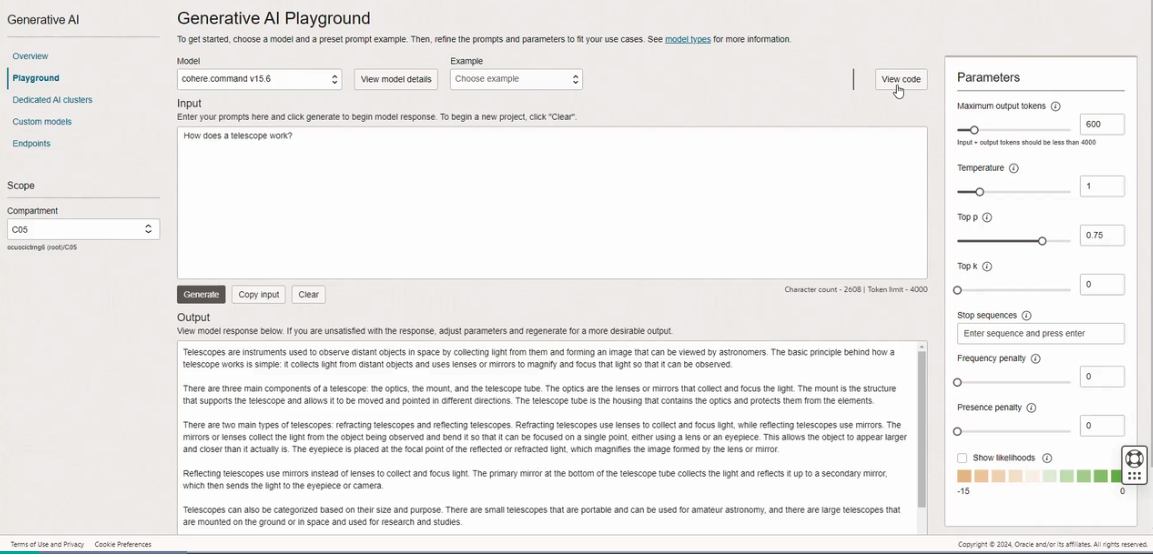


give simple prompt

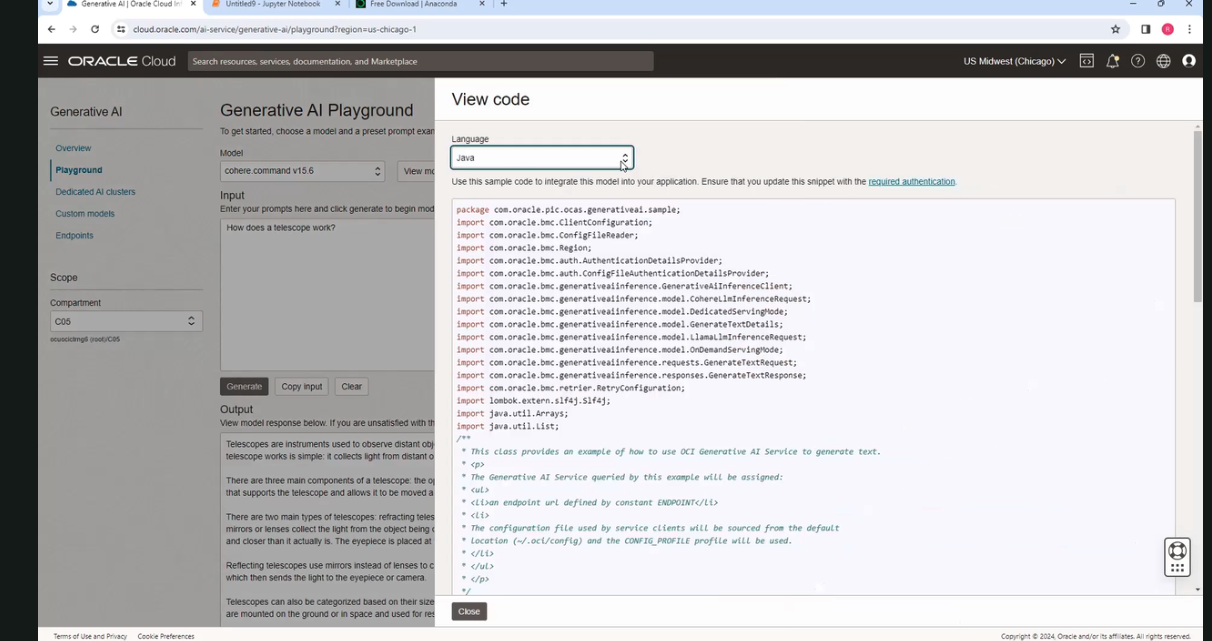


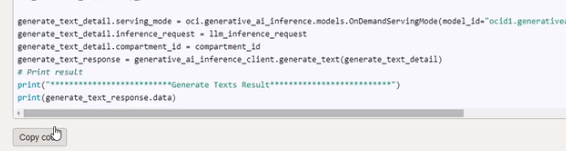
How does the API work? How can I write code to do the same activity?

Click on this button called View Code.



I can see the code for making this input output work in Java, as well as Python.





Copy and paste it in Jupyter Notebook - It lets you run the code in the browser, it sends it to a Notebook server that is running in the background. And it interfaces with whatever kernel you are running on your machine. So in this case, I'm running Python 3, which is the default. Now, the kernel will evaluate the code, and send the results back, which get displayed in the browser.)

