In [ ]:

```
#Libaries
from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline
import torch
from sklearn.model_selection import train_test_split
import pandas as pb

#Initiateing Model and Tokenizer
device = "cuda:0" if torch.cuda.is_available() else "cpu"
tokenizer = AutoTokenizer.from_pretrained("GPT-sw3-126m-BaseLargeTunedHyp_3Epoch_myown")
model = AutoModelForCausalLM.from_pretrained("GPT-sw3-126m-BaseLargeTunedHyp_3Epoch_myown
")
model.eval()
model.to(device)
```

In [ ]:

```
#Takes away all columns that have less than one procent of its boxes filled.
def taBortEnProcent(FromFile, PlaceToSave):
    data = pb.read_excel(FromFile)
    missing_percentage = (data.isnull().sum() / len(data)) * 100
    cols_to_drop = missing_percentage[missing_percentage >= 99].index
    data_filtered = data.drop(columns=cols_to_drop)

    print("Shape after filtering:", data_filtered.shape)

    data_filtered.to_excel(PlaceToSave, index=False)

    return data_filtered
```

In [ ]:

```
#Formats the input text in the same way as it was trained

def formatering(File):

    data = File

    print(data.shape)

    indata = data.iloc[:, 1:].values.tolist()

    train_texts, val_and_test_texts = train_test_split(indata, test_size=0.2)
    val_texts, test_texts_input = train_test_split(val_and_test_texts, test_size=0.5)


    input_texts = [f"<|endoftext|><s>User: Skriv en patientjornal efter en ultraljudsunde
rsökning utifrån dessa värden: {test_texts_input}<s>Bot:"
                   for test_texts_input in zip(test_texts_input)]


    return {
        "test": input_texts
    }
```

In [ ]:

```
#Calculate the leng of the input and takes away tokens if its to long. Then generates a t
ext.

def generating(text):
    prompt = text.strip()

    token_count = len(tokenizer.encode_plus(prompt)["input_ids"])
    max_token_count = 2048 - 140

    if token_count > max_token_count:
```

```
        end_index = len(prompt) - 7
        end_seq = prompt[end_index:]
        tokens = tokenizer.encode_plus(prompt[:end_index])["input_ids"]
        tokens = tokens[:max_token_count]
        prompt = (tokenizer.decode(tokens) + (end_seq)).strip()


    generator = pipeline('text-generation', tokenizer=tokenizer, model=model, device=dev
ice)
    generated = generator(prompt, max_new_tokens=140, do_sample=True, temperature=0.47,
top_p=1, top_k = 23, repetition_penalty = 1.05)[0]["generated_text"]
    return generated
```

In [ ]:

```
#process to generate text
'''
As you can see in the below functione a new file is saved after 1 % have been taken away.
So if you alredy have doen this function one time, its more time efficent
to just loade the new file directly
'''
file = taBortEnProcent("From_this_file", "To_this_file")

test_texter = formatering(file)["test"]
idx = 3
generated_text = generating(test_texter[idx])
print(generated_text)
```