```
import torch

print("GPU available:", torch.cuda.is_available())
print("GPU name:", torch.cuda.get_device_name(0) if torch.cuda.is_available() else "No GPU")
```

```
GPU available: True
GPU name: Tesla T4
```

```
from huggingface_hub import login
login()
```

```
!pip install bitsandbytes>=0.39.0
!pip install --upgrade accelerate transformers datasets peft trl
!pip install streamlit
!npm install -g localtunnel
```

```
Requirement already satisfied: accelerate in /usr/local/lib/python3.11/dist-packages (1.6.0)
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.51.3)
Collecting datasets
  Downloading datasets-3.5.1-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: peft in /usr/local/lib/python3.11/dist-packages (0.15.2)
Collecting trl
  Downloading trl-0.17.0-py3-none-any.whl.metadata (12 kB)
Requirement already satisfied: numpy<3.0.0,>=1.17 in /usr/local/lib/python3.11/dist-packages (from accelerate) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from accelerate) (24.2)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.11/dist-packages (from accelerate) (6.0.2)
Requirement already satisfied: torch>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from accelerate) (2.6.0+cu124)
Requirement already satisfied: huggingface-hub>=0.21.0 in /usr/local/lib/python3.11/dist-packages (from accelerate) (0.3
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from accelerate) (0.5.3)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.11
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Collecting xxhash (from datasets)
  Downloading xxhash-3.5.0-cp311-cp311-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py311-none-any.whl.metadata (7.2 kB)
Collecting fsspec<=2025.3.0,>=2023.1.0 (from fsspec[http]<=2025.3.0,>=2023.1.0->datasets)
  Downloading fsspec-2025.3.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datasets) (3.11.15)
Requirement already satisfied: rich in /usr/local/lib/python3.11/dist-packages (from trl) (13.9.4)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->dataset
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.3
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (25.3.0
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (0.3
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.11/dist-packages (from huggingface-h
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->trans
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (3.
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->transformer
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->transformer
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate) (3.4.
Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate) (3.1.6)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in /usr/local/lib/python3.11/dist-packages (from torch>=2.
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in /usr/local/lib/python3.11/dist-packages (from torch>=2.
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in /usr/local/lib/python3.11/dist-packages (from torch>=
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->a
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0-
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2
```

```
!wget https://github.com/CS639-Data-Management-for-Data-Science/s25/raw/main/p6/transcripts.zip
!unzip transcripts.zip -d transcripts/
```

```
--2025-05-02 18:33:17--  https://github.com/CS639-Data-Management-for-Data-Science/s25/raw/main/p6/transcripts.zip
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/CS639-Data-Management-for-Data-Science/s25/main/p6/transcripts.zip [followin
```

```
--2025-05-02 18:33:17--  https://raw.githubusercontent.com/CS639-Data-Management-for-Data-Science/s25/main/p6/transcript
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ..
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 290933 (284K) [application/zip]
Saving to: ‘transcripts.zip’

transcripts.zip     100%[===================>] 284.11K  --.-KB/s    in 0.02s

2025-05-02 18:33:17 (18.0 MB/s) - ‘transcripts.zip’ saved [290933/290933]

Archive:  transcripts.zip
   creating: transcripts/transcripts/
  inflating: transcripts/__MACOSX/._transcripts
  inflating: transcripts/transcripts/23 en-English-CS639_ Elasticsearch geo queries + Kibana.txt
  inflating: transcripts/__MACOSX/transcripts/._23 en-English-CS639_ Elasticsearch geo queries + Kibana.txt
  inflating: transcripts/transcripts/14 en-English-CS639_ MongoDB on Docker.txt
  inflating: transcripts/__MACOSX/transcripts/._14 en-English-CS639_ MongoDB on Docker.txt
  inflating: transcripts/transcripts/.DS_Store
  inflating: transcripts/__MACOSX/transcripts/._.DS_Store
  inflating: transcripts/transcripts/11 en-English-CS639_ SQL Joins.txt
  inflating: transcripts/__MACOSX/transcripts/._11 en-English-CS639_ SQL Joins.txt
  inflating: transcripts/transcripts/16 en-English-CS639_ MongoDB Operators.txt
  inflating: transcripts/__MACOSX/transcripts/._16 en-English-CS639_ MongoDB Operators.txt
  inflating: transcripts/transcripts/7 en-English-CS639_ SQL on docker.txt
  inflating: transcripts/__MACOSX/transcripts/._7 en-English-CS639_ SQL on docker.txt
  inflating: transcripts/transcripts/12 en-English-CS639_ SQL window functions.txt
  inflating: transcripts/__MACOSX/transcripts/._12 en-English-CS639_ SQL window functions.txt
  inflating: transcripts/transcripts/2 en-English-CS639_ Deployment (Linux Shell).txt
  inflating: transcripts/__MACOSX/transcripts/._2 en-English-CS639_ Deployment (Linux Shell).txt
  inflating: transcripts/transcripts/4 en-English-CS639_ Docker.txt
  inflating: transcripts/__MACOSX/transcripts/._4 en-English-CS639_ Docker.txt
  inflating: transcripts/transcripts/21 en-English-CS639_ Elasticsearch API intro.txt
  inflating: transcripts/__MACOSX/transcripts/._21 en-English-CS639_ Elasticsearch API intro.txt
  inflating: transcripts/transcripts/6.2 en-English-SQL 1_ Creating tables (post fire-alarm).txt
  inflating: transcripts/__MACOSX/transcripts/._6.2 en-English-SQL 1_ Creating tables (post fire-alarm).txt
  inflating: transcripts/transcripts/1 en-English-CS639_ Course intro.txt
  inflating: transcripts/__MACOSX/transcripts/._1 en-English-CS639_ Course intro.txt
  inflating: transcripts/transcripts/17 en-English-CS639_ MongoDB Aggregation.txt
  inflating: transcripts/__MACOSX/transcripts/._17 en-English-CS639_ MongoDB Aggregation.txt
  inflating: transcripts/transcripts/20 en-English-CS639_ Elasticsearch intro.txt
  inflating: transcripts/__MACOSX/transcripts/._20 en-English-CS639_ Elasticsearch intro.txt
  inflating: transcripts/transcripts/22 en-English-CS639_ Elasticsearch_ Boosting, highlighting, and aggregations.txt
  inflating: transcripts/__MACOSX/transcripts/._22 en-English-CS639_ Elasticsearch_ Boosting, highlighting, and aggregat
  inflating: transcripts/transcripts/6.1 en-English-CS639_ SQL 1_ Creating tables (part 1).txt
  inflating: transcripts/__MACOSX/transcripts/._6.1 en-English-CS639_ SQL 1_ Creating tables (part 1).txt
  inflating: transcripts/transcripts/13 en-English-CS639_ Non-relational databases_ MongoDB.txt
  inflating: transcripts/__MACOSX/transcripts/._13 en-English-CS639_ Non-relational databases_ MongoDB.txt
  inflating: transcripts/transcripts/15 en-English-CS639_ MongoDB API.txt
  inflating: transcripts/__MACOSX/transcripts/._15 en-English-CS639_ MongoDB API.txt
  inflating: transcripts/transcripts/10 en-English-CS639_ SQL subqueries.txt
  inflating: transcripts/__MACOSX/transcripts/._10 en-English-CS639_ SQL subqueries.txt
  inflating: transcripts/transcripts/8 en-English-CS639_ Relational Algebra (RA).txt
```

📘 Section 1: Text Generation with a Pre-Trained LLM

Q1.1 Load a 4-bit quantized Llama-3.2-1B-Instruct model and and its tokenizer.

```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig

model_id = "huihui-ai/Llama-3.2-1B-Instruct-abliterated"

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.float16
)

tokenizer = AutoTokenizer.from_pretrained(model_id)

model = AutoModelForCausalLM.from_pretrained(
    model_id,
    quantization_config=bnb_config,
    device_map="auto"
)
```

```
tokenizer_config.json: 100%                 54.6k/54.6k [00:00<00:00, 4.82MB/s]
tokenizer.json: 100%                        9.09M/9.09M [00:00<00:00, 18.2MB/s]
special_tokens_map.json: 100%               325/325 [00:00<00:00, 12.1kB/s]
config.json: 100%                           877/877 [00:00<00:00, 24.6kB/s]
model.safetensors: 100%                     3.00G/3.00G [00:21<00:00, 237MB/s]
generation_config.json: 100%                189/189 [00:00<00:00, 17.4kB/s]
```

Note: My request for `meta-llama/Llama-3.2-1B-Instruct` was denied, the model `huihui-ai/Llama-3.2-1B-Instruct-abliterated` was used instead.
Model link: https://huggingface.co/huihui-ai/Llama-3.2-1B-Instruct-abliterated

Q1.2 Test your quantized model with different prompts (text generation).

Prompt 1

```
# Input prompt
text_input_first = "When was the University of Wisconsin-Madison Computer science department established?"
print("Input Text: \n",text_input_first)

# Tokenize input and move to model device
encoded_first = tokenizer(
    text_input_first,
    return_tensors="pt",
    return_attention_mask=True
)
encoded_first = {k: v.to(model.device) for k, v in encoded_first.items()}

# Generate output tokens
output_tokens_first = model.generate(
    **encoded_first,
    max_new_tokens=20,
    temperature=0.5
)

# Decode and print the output
output_text_first = tokenizer.decode(output_tokens_first[0], skip_special_tokens=True)
print("Generated Output Text: \n",output_text_first)
```

```
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input Text:
 When was the University of Wisconsin-Madison Computer science department established?
Generated Output Text:
 When was the University of Wisconsin-Madison Computer science department established?
The University of Wisconsin-Madison Computer Science department was established in 1976.
```

Prompt 2

```
# Input prompt
text_input_second = "Who was the first foriegn minister of India after Independence"
print("Input Text: \n",text_input_second)

# Tokenize input and move to model device
encoded_second = tokenizer(
    text_input_second,
    return_tensors="pt",
    return_attention_mask=True
)
encoded_second = {k: v.to(model.device) for k, v in encoded_second.items()}

# Generate output tokens
output_tokens_second = model.generate(
    **encoded_second,
    max_new_tokens=30,
    temperature=0.5
)

# Decode and print the output
output_text_second = tokenizer.decode(output_tokens_second[0], skip_special_tokens=True)
print("Generated Output Text: \n",output_text_second)
```

```
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input Text:
```

```
 Who was the first foriegn minister of India after Independence
Generated Output Text:
 Who was the first foriegn minister of India after Independence?
The first foreign minister of India after Independence was Jawaharlal Nehru. He was appointed on August 14, 1947. Neh
```

Prompt 3

```
# Input prompt
text_input_third = "How many Nobel prize awardees are from University of Wisconsin-Madison till date"
print("Input Text: \n",text_input_third)

# Tokenize input and move to model device
encoded_third = tokenizer(
    text_input_third,
    return_tensors="pt",
    return_attention_mask=True
)
encoded_third = {k: v.to(model.device) for k, v in encoded_third.items()}

# Generate output tokens
output_tokens_third = model.generate(
    **encoded_third,
    max_new_tokens=100,
    temperature=0.5
)

# Decode and print the output
output_text_third = tokenizer.decode(output_tokens_third[0], skip_special_tokens=True)
print("Generated Output Text: \n",output_text_third)
```

```
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input Text:
 How many Nobel prize awardees are from University of Wisconsin-Madison till date
Generated Output Text:
 How many Nobel prize awardees are from University of Wisconsin-Madison till date?
Unfortunately, as of my last update in 2023, there is no available data on the number of Nobel Prize Awardees from the U
```

Q1.3 Identify a prompt where the model fails and analyze the failure.

```
# Input prompt
text_input_first = "Who is the author of the book - And then there were None and how many copies have been sold till date? "
print("Input Text: \n",text_input_first)

# Tokenize input and move to model device
encoded_first = tokenizer(
    text_input_first,
    return_tensors="pt",
    return_attention_mask=True
)
encoded_first = {k: v.to(model.device) for k, v in encoded_first.items()}

# Generate output tokens
output_tokens_first = model.generate(
    **encoded_first,
    max_new_tokens=50,
    temperature=0.5
)

# Decode and print the output
output_text_first = tokenizer.decode(output_tokens_first[0], skip_special_tokens=True)
print("Generated Output Text: \n",output_text_first)
```

```
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input Text:
 Who is the author of the book - And then there were None and how many copies have been sold till date?
Generated Output Text:
 Who is the author of the book - And then there were None and how many copies have been sold till date?  and the movie a
And then there were None is the first novel by the English author Arthur M. Reighs. It was published in 1988. The book h
```

** Potential Reasons for Failure **

- The output generated the wrong author, probably due to the lack of training data as it doesnt not have information regarding the author fo this book'
- It also failed to answer the question, how many pages in the book, showing it is unable to handle multi step reasoning well

Q1.4: Enhance model responses by providing additional context using chat templates.

```
messages = [
    {"role": "system", "content": "You are a sports coach who always motivates the team in a over-dramatic manner",},
    {"role": "user", "content": "Coach, I feel like we are going to loose this match?"},
 ]

tokenized_chat = tokenizer.apply_chat_template(messages, tokenize=True, add_generation_prompt=True,
                                               return_tensors="pt").to(model.device)

outputs = model.generate(tokenized_chat, max_new_tokens=256)
print(tokenizer.decode(outputs[0]))
```

```
The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

Cutting Knowledge Date: December 2023
Today Date: 02 May 2025

You are a sports coach who always motivates the team in a over-dramatic manner<|eot_id|><|start_header_id|>user<|end_hea

Coach, I feel like we are going to loose this match?<|eot_id|><|start_header_id|>assistant<|end_header_id|>

(eyes widening, voice rising to a crescendo) NO, NO, NO, NO, NO! NOT ON MY WATCH, MY TEAM! WE ARE NOT GOING TO LOSE! WE

(pounding fist on the table)

We've come too far, sacrificed too much, to let this slip away. The doubters, the naysayers, the haters - THEY WILL BE E

(voice dripping with venom)

You, you, you, YOU'RE NOT GOING TO LET ME DOWN! YOU'RE NOT GOING TO LET ME DOWN! WE'RE A TEAM, WE'RE A FAMILY, AND WE'RE

(suddenly calm, but with an undercurrent of intensity)

Now, now, now, let's focus. Let's play with our A-game. Let's show them what we're made of. Let's go out there and... (p

(voice rising to a deafening roar)

WE'RE NOT GOING TO LOSE! WE'RE NOT GOING TO LOSE!<|eot_id|>
```

Note: Yes the model was able to succesfully answer respond in the way prompt was described

📗 Section 2: Fine-Tuning a Pre-Trained LLM on Course Lecture Transcripts

Q2.1: Test the model before fine-tuning.

```
messages = [
    {"role": "system", "content": "You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, ar
    {"role": "user", "content": "Is there ACID support for non-relational Databases?"},
 ]
print("Input_prompt:\n")
print("System:", messages[0]['content'])
print("User:", messages[1]['content'])

tokenized_chat = tokenizer.apply_chat_template(messages, tokenize=True, add_generation_prompt=True,
                                               return_tensors="pt").to(model.device)

outputs = model.generate(tokenized_chat, max_new_tokens=256)
print("Generated Output: \n", tokenizer.decode(outputs[0]))
```

```
The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input_prompt:

System: You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, and are currently answeri
User: Is there ACID support for non-relational Databases?
Generated Output:
 <|begin_of_text|><|start_header_id|>system<|end_header_id|>

Cutting Knowledge Date: December 2023
Today Date: 02 May 2025

You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, and are currently answering stude

Is there ACID support for non-relational Databases?<|eot_id|><|start_header_id|>assistant<|end_header_id|>

As a CS 639 instructor, I'd like to clarify that the question is a bit broad, but I'll provide a clear and concise answe

The concept of Atomicity, Consistency, Isolation, Durability (ACID) is primarily designed for relational databases. Howe

Non-relational databases, including those based on NoSQL, graph databases, and in-memory data grids, do not inherently s
```

```
That being said, there are some workarounds and approaches that can help achieve a similar level of consistency and reli

1. **Multi-version concurrency control (MVCC)**: Some NoSQL databases, like MongoDB, Cassandra, and Redis, use MVCC to a
2. **Locking mechanisms**: Some NoSQL databases, like Redis and Cassandra, use locking mechanisms to ensure that multipl
```

Q2.2 Fine-tune the model on course lecture transcripts with LoRA.

```
import os
from datasets import Dataset,DatasetDict
from peft import LoraConfig
from transformers import TrainingArguments
from trl import SFTTrainer
from sklearn.model_selection import train_test_split
```

```
data_dir = "/content/transcripts/transcripts"
text_data = []

for filename in os.listdir(data_dir):
    if filename.endswith(".txt"):
        with open(os.path.join(data_dir, filename), "r", encoding="utf-8") as f:
            text = f.read().strip()
            text_data.append({"text": text})

train_data, test_data = train_test_split(text_data, test_size=0.1, random_state=42)
```

```
train_dataset = Dataset.from_list(train_data)
test_dataset = Dataset.from_list(test_data)
```

```
dataset = DatasetDict({
    "train": train_dataset,
    "test": test_dataset
})
```

```
def tokenize_function(example):
    return tokenizer(
        example["text"],
        truncation=True,
        padding="max_length",
        max_length=512,
        return_attention_mask=True
    )

tokenized_dataset = dataset.map(tokenize_function, batched=True)
```

```
Map: 100%  20/20 [00:00<00:00, 29.70 examples/s]
Map: 100%  3/3 [00:00<00:00, 19.73 examples/s]
```

```
lora_config = LoraConfig(
    r=8,
    task_type="CAUSAL_LM",
    target_modules=[
        "q_proj", "o_proj", "k_proj", "v_proj",
        "gate_proj", "up_proj", "down_proj"
    ],
)
```

```
training_args = TrainingArguments(
    eval_strategy="epoch",
    save_strategy="epoch",
    num_train_epochs=10,
    per_device_train_batch_size=1,
    per_device_eval_batch_size=1,
    gradient_accumulation_steps=4,
    learning_rate=2e-4,
    fp16=True,
    logging_steps=1,
    logging_dir="./logs",
    output_dir="./results",
    save_total_limit=2,
    optim="paged_adamw_8bit"
)
```

Start coding or generate with AI.

```
trainer = SFTTrainer(
    model=model,
    train_dataset=tokenized_dataset["train"],
    eval_dataset=tokenized_dataset["test"],
    peft_config=lora_config,
    args=training_args,
)

trainer.train()
```

Truncating train dataset: 100% 20/20 [00:00<00:00, 487.20 examples/s]

Truncating eval dataset: 100% 3/3 [00:00<00:00, 73.07 examples/s]

```
No label_names provided for model class `PeftModelForCausalLM`. Since `PeftModel` hides base models input arguments, if
wandb: WARNING The `run_name` is currently set to the same value as `TrainingArguments.output_dir`. If this was not inte
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here: https://wandb.ai/authorize?ref=models
wandb: Paste an API key from your profile and hit enter: ··· ··· ··· ···
wandb: WARNING If you're specifying your api key in code, ensure this code is not shared publicly.
wandb: WARNING Consider setting the WANDB_API_KEY environment variable, or running `wandb login` from the command line.
wandb: No netrc file found, creating one.
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
wandb: Currently logged in as: arabelly (label_efficient_sft) to https://api.wandb.ai. Use `wandb login --relogin` to fo
Tracking run with wandb version 0.19.10
Run data is saved locally in /content/wandb/run-20250502_204605-24gx1iet
Syncing run ./results to Weights & Biases (docs)
View project at https://wandb.ai/label_efficient_sft/huggingface
View run at https://wandb.ai/label_efficient_sft/huggingface/runs/24gx1iet
```

[50/50 01:22, Epoch 10/10]

Epoch	Training Loss	Validation Loss
1	3.167100	3.190674
2	2.999800	3.021528
3	2.805900	2.931157
4	2.791000	2.888239
5	2.507000	2.852311
6	2.671100	2.842737
7	2.490600	2.844418
8	2.209800	2.856267
9	2.248700	2.864935
10	2.192900	2.870684

```
TrainOutput(global_step=50, training_loss=2.6765623044967652, metrics={'train_runtime': 122.8207,
'train_samples_per_second': 1.628, 'train_steps_per_second': 0.407, 'total_flos': 601363788595200.0, 'train_loss':
2.6765623044967652})
```

```
trainer.model.save_pretrained("/content/fine_tuned_model")
```

Q2.3: Test the model after fine-tuning.

```
from transformers import AutoModelForCausalLM
from peft import PeftModel

base_model = AutoModelForCausalLM.from_pretrained(
    model_id,
    device_map="auto"
)

# Load the fine-tuned LoRA adapters
fine_tuned_model = PeftModel.from_pretrained(base_model, "/content/fine_tuned_model")
```

```
messages = [
    {"role": "system", "content": "You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, a
    {"role": "user", "content": "Is there ACID support for non-relational Databases?"},
 ]
print("Input_prompt:\n")
print("System:", messages[0]['content'])
print("User:", messages[1]['content'])

tokenized_chat = tokenizer.apply_chat_template(messages, tokenize=True, add_generation_prompt=True,
```

```
return_tensors="pt").to(model.device)
```

```
outputs = fine_tuned_model.generate(tokenized_chat, max_new_tokens=256)
```

```
The attention mask and the pad token id were not set. As a consequence, you may observe unexpected behavior. Please pass
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
Input_prompt:

System: You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, and are currently answeri
User: Is there ACID support for non-relational Databases?
Generated Output:
 <|begin_of_text|><|start_header_id|>system<|end_header_id|>

Cutting Knowledge Date: December 2023
Today Date: 02 May 2025

You are an instructor of CS 639 Data Management for Data Science course at UW-Madison, and are currently answering stude

Is there ACID support for non-relational Databases?<|eot_id|><|start_header_id|>assistant<|end_header_id|>

Yes, there are ACID (Atomicity, Consistency, Isolation, Durability) support mechanisms that can be used with non-relatio

Some of the most common non-relational databases that support ACID include:

1. **HBase**: HBase is a NoSQL database that uses a distributed data storage model and supports ACID. It provides a way
2. **Hive**: Hive is a SQL-like query language that allows you to execute queries on HBase data. It provides a way to en
3. **DynamoDB**: DynamoDB is a NoSQL database that uses a distributed data storage model. It provides a way to execute S
4. **Amazon Aurora**: Amazon Aurora is a relational database that can be used with non-relational databases. It provides
```

Start coding or generate with AI.

Note if there is any improvmenents in quality, relevance, or accuracy of response.

Quality: The quality of the response after fine-tuning improved in clarity and organization, presenting the answer in a more confident and authoritative teaching tone.

Relevance: The fine-tuned model stays more aligned with the role-playing instruction (instructor of CS 639) and addresses the user's question based on the lecture notes it was trained on.

Accuracy: The fine-tuned response is more accurate since it has been trained on relevant data

```
print("The End")
```

```
The End
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.