Perfect 👌 You’ve chosen **Flan-T5** — a great option for **Q&A in a specific domain**.  
I’ll walk you through **how to use it, fine-tune it, and test it** step by step.

**🔹 Step 1: Install Required Libraries**

pip install transformers datasets accelerate peft

**🔹 Step 2: Load a Pretrained Flan-T5 Model**

from transformers import AutoTokenizer, AutoModelForSeq2SeqLM

# Load Flan-T5 small/base/large

model\_name = "google/flan-t5-base"

tokenizer = AutoTokenizer.from\_pretrained(model\_name)

model = AutoModelForSeq2SeqLM.from\_pretrained(model\_name)

**🔹 Step 3: Prepare Your Dataset**

👉 For Q&A, you need **(question, context, answer)** format.  
Example domain dataset (can be JSON, CSV, etc.):

from datasets import Dataset

# Example domain Q&A dataset

data = {

"question": [

"What is the policy number?",

"Who is the insurance provider?"

],

"context": [

"The insurance policy number is 12345 issued by ICICI Lombard.",

"The provider of this insurance is HDFC ERGO."

],

"answer": [

"12345",

"HDFC ERGO"

]

}

dataset = Dataset.from\_dict(data)

# Split into train and test

dataset = dataset.train\_test\_split(test\_size=0.2)

**🔹 Step 4: Tokenize Data**

def preprocess(example):

inputs = ["question: " + q + " context: " + c for q, c in zip(example["question"], example["context"])]

model\_inputs = tokenizer(inputs, padding="max\_length", truncation=True, max\_length=256)

labels = tokenizer(example["answer"], padding="max\_length", truncation=True, max\_length=64)

model\_inputs["labels"] = labels["input\_ids"]

return model\_inputs

tokenized = dataset.map(preprocess, batched=True)

**🔹 Step 5: Fine-tune with Hugging Face Trainer**

from transformers import Seq2SeqTrainer, Seq2SeqTrainingArguments, DataCollatorForSeq2Seq

training\_args = Seq2SeqTrainingArguments(

output\_dir="./flan\_t5\_finetuned",

evaluation\_strategy="epoch",

learning\_rate=5e-5,

per\_device\_train\_batch\_size=4,

per\_device\_eval\_batch\_size=4,

num\_train\_epochs=3,

weight\_decay=0.01,

save\_total\_limit=2,

predict\_with\_generate=True,

fp16=True

)

data\_collator = DataCollatorForSeq2Seq(tokenizer, model=model)

trainer = Seq2SeqTrainer(

model=model,

args=training\_args,

train\_dataset=tokenized["train"],

eval\_dataset=tokenized["test"],

tokenizer=tokenizer,

data\_collator=data\_collator

)

trainer.train()

**🔹 Step 6: Test the Model**

def answer\_question(question, context):

input\_text = f"question: {question} context: {context}"

inputs = tokenizer(input\_text, return\_tensors="pt", truncation=True)

outputs = model.generate(\*\*inputs, max\_length=64)

return tokenizer.decode(outputs[0], skip\_special\_tokens=True)

# Example test

print(answer\_question("What is the policy number?", "The insurance policy number is 12345 issued by ICICI Lombard."))

print(answer\_question("Who is the insurance provider?", "The provider of this insurance is HDFC ERGO."))

✅ Expected output:

12345

HDFC ERGO

**🔹 Step 7: Save & Reload Model**

model.save\_pretrained("./flan\_t5\_finetuned")

tokenizer.save\_pretrained("./flan\_t5\_finetuned")

# Reload later

model = AutoModelForSeq2SeqLM.from\_pretrained("./flan\_t5\_finetuned")

tokenizer = AutoTokenizer.from\_pretrained("./flan\_t5\_finetuned")

**✅ Summary**

1. Use **Flan-T5** (flan-t5-base) for Q&A.
2. Prepare domain dataset as (question, context, answer).
3. Fine-tune with Hugging Face **Trainer**.
4. Test by asking questions against domain context.

Would you like me to also show you how to do the **same fine-tuning with PEFT (LoRA)**?  
👉 That way, you can fine-tune Flan-T5 with **much less GPU memory** (good if you don’t have a big GPU).

Top of Form

Bottom of Form