```
!pip install transformers datasets accelerate sentencepiece
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.11/dist-packages (4.52.4)
Requirement already satisfied: datasets in /usr/local/lib/python3.11/dist-packages (2.14.4)
Requirement already satisfied: accelerate in /usr/local/lib/python3.11/dist-packages (1.7.0)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.11/dist-packages (0.2.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.32.
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.11/dist-packages (from transformers) (2024.11.6)
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) (0.21.1)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: pyarrow>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.8,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.3.7)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.11/dist-packages (from datasets) (0.70.15)
Requirement already satisfied: fsspec>=2021.11.1 in /usr/local/lib/python3.11/dist-packages (from fsspec[http]>=2021.11.1->datase
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datasets) (3.11.15)
Requirement already satisfied: psutil in /usr/local/lib/python3.11/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: torch>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from accelerate) (2.6.0+cu124)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (2.6.1
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.3.2)
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) (1.6.0)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (6.4.4)
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.11/dist-packages (from aiohttp->datasets) (1.20.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0,>=
Requirement already satisfied: hf-xet<2.0.0,>=1.1.2 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0,>=0.30.0
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->transformers)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (2.4.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->transformers) (2025.
Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate) (3.5)
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>=2.0.0->ac
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->ac
Requirement already satisfied: nvidia-cudnn-cu12==9.1.0.70 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->acceler
Requirement already satisfied: nvidia-cublas-cu12==12.4.5.8 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accele
Requirement already satisfied: nvidia-cufft-cu12==11.2.1.3 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->acceler
Requirement already satisfied: nvidia-curand-cu12==10.3.5.147 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->acce
Requirement already satisfied: nvidia-cusolver-cu12==11.6.1.9 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->acce
Requirement already satisfied: nvidia-cusparse-cu12==12.3.1.170 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->ac
Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accel
Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate
Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelera
Requirement already satisfied: nvidia-nvjitlink-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->acc
Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate) (3.2.0)
Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-packages (from torch>=2.0.0->accelerate) (1.13.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch>=2.0.0->a
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2.9.0.p
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2025.2)
```

```
import pandas as pd

# Load manually labeled CSV
df = pd.read_csv("https://raw.githubusercontent.com/psabhay2003/NLP-driven-Invoice-Management-System/refs/heads/main/labeled%20data%20sa

# Create prompt-response style training format
def make_prompt(row):
    return f"Extract invoice fields: {row['extracted_text']}"

def make_output(row):
    return f"Invoice No: {row['Invoice number']}, Date: {row['Date']}, Total Amount: {row['Total Amount']}, Vendor: {row['Vendor']}"

df['input_text'] = df.apply(make_prompt, axis=1)
df['target_text'] = df.apply(make_output, axis=1)
```

```
from datasets import Dataset
from transformers import T5Tokenizer

tokenizer = T5Tokenizer.from_pretrained("t5-base")

train_dataset = Dataset.from_pandas(df[['input_text', 'target_text']])

def preprocess(example):
    inputs = tokenizer(example['input_text'], max_length=512, truncation=True, padding="max_length")
    targets = tokenizer(example['target_text'], max_length=128, truncation=True, padding="max_length")
    inputs['labels'] = targets['input_ids']
    return inputs

tokenized_dataset = train_dataset.map(preprocess, batched=False)
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
spiece.model: 100%                                   792k/792k [00:00<00:00, 6.41MB/s]
tokenizer.json: 100%                                 1.39M/1.39M [00:00<00:00, 9.06MB/s]
config.json: 100%                                    1.21k/1.21k [00:00<00:00, 59.4kB/s]
You are using the default legacy behaviour of the <class 'transformers.models.t5.tokenization_t5.T5Tokenizer'>. This is expected, ar
Map: 100%                                            100/100 [00:00<00:00, 237.47 examples/s]
```

```
from transformers import T5ForConditionalGeneration, TrainingArguments, Trainer

model = T5ForConditionalGeneration.from_pretrained("t5-base")

training_args = TrainingArguments(
    output_dir="./t5_invoice_model",
    per_device_train_batch_size=4,
    num_train_epochs=5,
    logging_steps=10,
    save_steps=50,
    save_total_limit=1,
    fp16=True,
    report_to="none"
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset,
    tokenizer=tokenizer
)

trainer.train()
```

```
model.safetensors: 100%                              892M/892M [00:09<00:00, 118MB/s]
generation_config.json: 100%                         147/147 [00:00<00:00, 15.3kB/s]
<ipython-input-9-1824018829>:16: FutureWarning: `tokenizer` is deprecated and will be removed in version 5.0.0 for `Trainer.__init_
  trainer = Trainer(
Passing a tuple of `past_key_values` is deprecated and will be removed in Transformers v4.48.0. You should pass an instance of `Enco
[125/125 01:52, Epoch 5/5]
```

Step	Training Loss
10	10.776300
20	2.748100
30	0.631800
40	0.304100
50	0.171800
60	0.102700
70	0.062400
80	0.049500
90	0.043300
100	0.039200
110	0.032300
120	0.042700

```
TrainOutput(global_step=125, training_loss=1.2016682304143906, metrics={'train_runtime': 114.5922, 'train_samples_per_second':
4.363, 'train_steps_per_second': 1.091, 'total_flos': 304478945280000.0, 'train_loss': 1.2016682304143906, 'epoch': 5.0})
```

```
# Load OCR CSV
ocr_df = pd.read_csv("https://raw.githubusercontent.com/psabhay2003/NLP-driven-Invoice-Management-System/refs/heads/main/invoice_texts.c

# Prepare input prompts
ocr_df['prompt'] = ocr_df['extracted_text'].apply(lambda x: f"Extract invoice fields: {x}")

# Generate predictions
def generate_prediction(text):
    inputs = tokenizer(text, return_tensors="pt", truncation=True, max_length=512).to(model.device)
    output = model.generate(**inputs, max_length=128)
    return tokenizer.decode(output[0], skip_special_tokens=True)

ocr_df['extracted_fields'] = ocr_df['prompt'].apply(generate_prediction)



ocr_df[['filename', 'extracted_fields']].to_csv("T5_output.csv", index=False)
from google.colab import files
files.download("T5_output.csv")
```

```
#push this T5 output to github and then convert into the final csv which will be used in SQL database
t5_df = pd.read_csv("https://raw.githubusercontent.com/psabhay2003/NLP-driven-Invoice-Management-System/refs/heads/main/T5_output.csv")
import re
def parse_fields(txt):
    #regex patterns to cover different label styles
    patterns = [
        # Standard “Invoice No: …, Date: …, Total Amount: …, Vendor: …”
        r'Invoice\s*No[:\-]?\s*(?P<inv>[^,;\n]+)[,;\n]\s*Date[:\-]?\s*(?P<date>[^,;\n]+)[,;\n]\s*Total\s*Amount[:\-]?\s*(?P<amt>[^,;\n]+
        # Using “#” instead of “No”
        r'Invoice\s*#[:\-]?\s*(?P<inv>[^,;\n]+)[,;\n]\s*Date[:\-]?\s*(?P<date>[^,;\n]+)[,;\n]\s*Total[:\-]?\s*(?P<amt>[^,;\n]+)[,;\n]\s
        # All four as key:value pairs separated by semicolons
        r'Invoice\s*No[:\-]?\s*(?P<inv>[^;]+);\s*Date[:\-]?\s*(?P<date>[^;]+);\s*Total\s*Amount[:\-]?\s*(?P<amt>[^;]+);\s*Vendor[:\-]?\s
        # CSV-style “1234,2023-01-01,1500,Acme Corp”
        r'^(?P<inv>[A-Z0-9\-]+)\s*,\s*(?P<date>\d{1,2}[\/\-\.][A-Za-z0-9\/\-\.]+)\s*,\s*(?P<amt>[₹\$]?\s*\d+[.,]?\d*)\s*,\s*(?P<vend>.+)
    ]
    for pat in patterns:
        m = re.search(pat, txt.strip(), flags=re.IGNORECASE)
        if m:
            return m.group('inv').strip(), m.group('date').strip(), m.group('amt').strip(), m.group('vend').strip()
    # Fallback: split on commas/newlines, then pick by prefix
    parts = re.split(r',|\n|;', txt)
    inv = date = amt = vend = None
    for p in parts:
        p = p.strip()
        low = p.lower()
        if inv is None and 'invoice' in low:
            inv = re.sub(r'[^A-Z0-9\-]', '', p)
        elif date is None and re.search(r'\d{1,2}[\/\-\.\s][A-Za-z]{3,}\s*\d{2,4}', p):
            date = p
        elif amt is None and re.search(r'\d+[.,]?\d*', p):
            amt = p
        elif vend is None and len(p) > 3:
            vend = p
    return inv, date, amt, vend

# Apply parsing
parsed = t5_df['extracted_fields'].apply(lambda x: pd.Series(parse_fields(str(x)),
                                                             index=['Invoice No','Date','Total Amount','Vendor']))
final = pd.concat([t5_df['filename'], parsed], axis=1)

# Save and download
final.to_csv("final_structured_output.csv", index=False)
from google.colab import files
files.download("final_structured_output.csv")
```