Assignment: Fine-Tuning a Language Model (LLM) for a Specific Topic

Introduction:

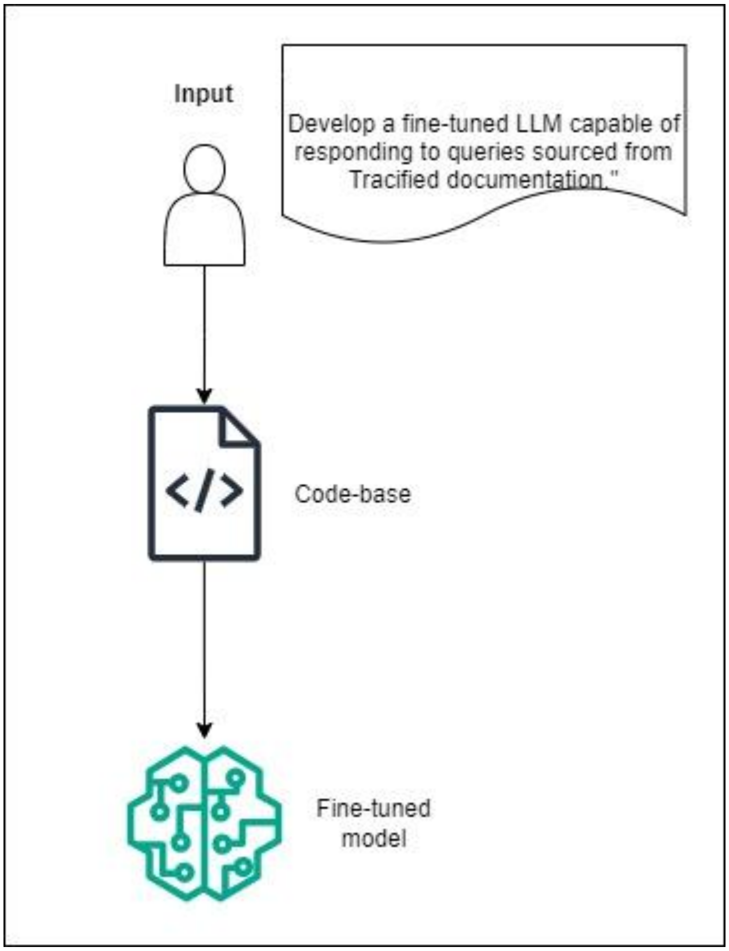
This document serves as technical documentation detailing the process of automatically constructing a fine-tuning model based on user instructions.

You can find the complete codebase of this implementation here.
https://github.com/shashikalaranathunga/fine_tuned_LLM.git

Usecase:

Our selected use case is to fine-tune a Large Language Model (LLM) capable of answering any questions from the Tracified documentation.

High-Level overview of the System


Input
Develop a fine-tuned LLM capable of responding to queries sourced from Tracified documentation."
Code-base
Fine-tuned model

Methodology

The process entails the following steps:

1.Dataset Generation: Automatically generating the dataset.
2.System Message Generation: The proposed system will create an effective system prompt for your model.
3.Fine-Tuning: Upon dataset generation, the system will automatically fine-tune a model, and prepare it for inference.

1.Dataset Generation

In this step, we automatically generate the dataset to fine-tune the model. We initialize the following parameters:

1. Temperature: We opt for a lower value to ensure more precise question answers from the documentation.

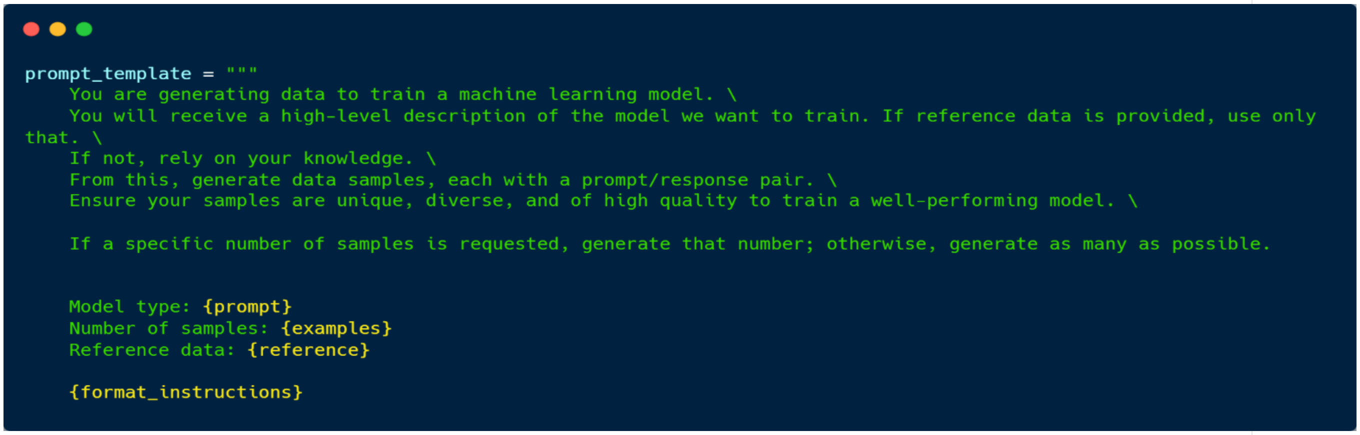
2. Number of examples: We leave this parameter empty as the number of examples generated depends on the given content.

3. Reference document: For our selected use case, we provide a reference document to the model for generating data samples. If you don't need to provide reference data for your use case, which means the model itself needs to generate examples, you can leave the reference document field empty. In our case, we automatically scrape the content from the corresponding web documentation and input the extracted content into the model to generate data samples. You can find the web-scraped data [here](link_here).

Next, we define the data generation chain, which comprises the following components:

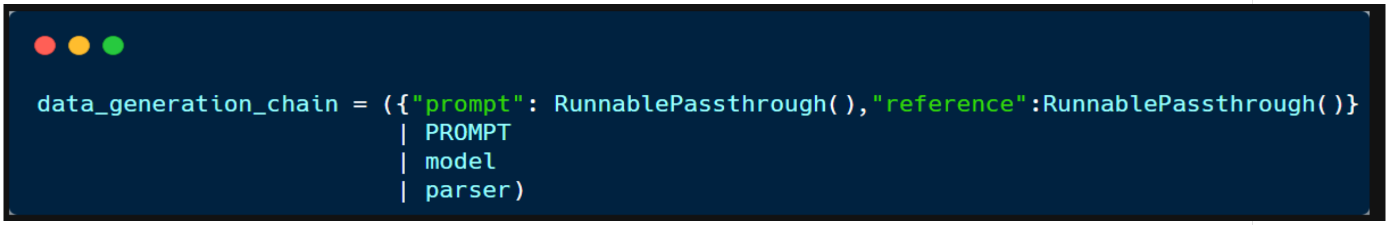
1. Prompt: We establish a clear prompt with input variables.
2. Model: We utilize the latest 'gpt-4-1106-preview' model.
3. Parser: We specify a parser to ensure that the output is structured and consistent at all times.

Dataset_generation_prompt:



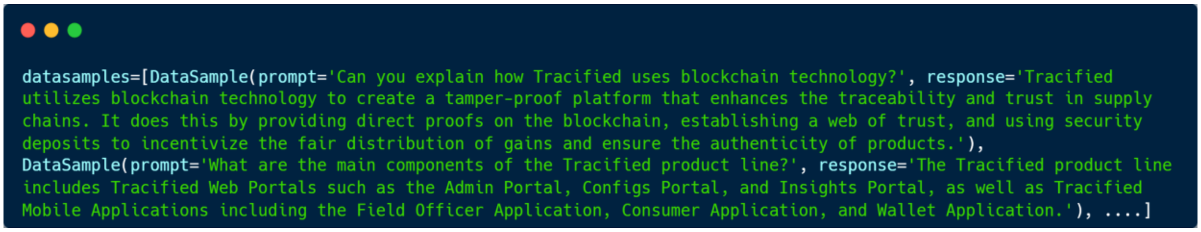
```
prompt_template = """
    You are generating data to train a machine learning model. \
    You will receive a high-level description of the model we want to train. If reference data is provided, use only
that. \
    If not, rely on your knowledge. \
    From this, generate data samples, each with a prompt/response pair. \
    Ensure your samples are unique, diverse, and of high quality to train a well-performing model. \

    If a specific number of samples is requested, generate that number; otherwise, generate as many as possible.


    Model type: {prompt}
    Number of samples: {examples}
    Reference data: {reference}

    {format_instructions}
```

Dataset_generation_chain:



```
data_generation_chain = ({"prompt": RunnablePassthrough(),"reference":RunnablePassthrough()}
                          | PROMPT
                          | model
                          | parser)
```

The output of this chain will consist of the generated data samples as outlined in the parser.

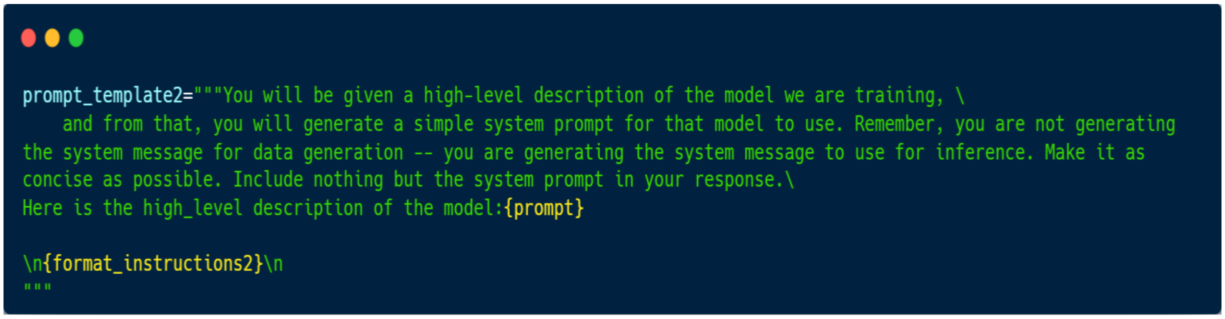


```
datasamples=[DataSample(prompt='Can you explain how Tracified uses blockchain technology?', response='Tracified
utilizes blockchain technology to create a tamper-proof platform that enhances the traceability and trust in supply
chains. It does this by providing direct proofs on the blockchain, establishing a web of trust, and using security
deposits to incentivize the fair distribution of gains and ensure the authenticity of products.'),
DataSample(prompt='What are the main components of the Tracified product line?', response='The Tracified product line
includes Tracified Web Portals such as the Admin Portal, Configs Portal, and Insights Portal, as well as Tracified
Mobile Applications including the Field Officer Application, Consumer Application, and Wallet Application.'), ....]
```

2.System Message Generation:

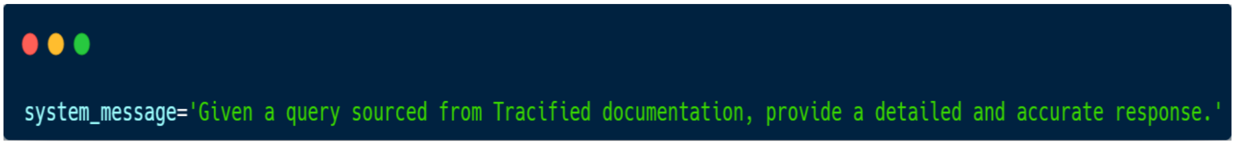
In this stage, we provide a clear and precise prompt to generate system messages for use in inference. We then utilize the output parser to structure the output more consistently and systematically.

System message generation prompt:



```
prompt_template2="""You will be given a high-level description of the model we are training, \
    and from that, you will generate a simple system prompt for that model to use. Remember, you are not generating
the system message for data generation -- you are generating the system message to use for inference. Make it as
concise as possible. Include nothing but the system prompt in your response.\
Here is the high_level description of the model:{prompt}

\n{format_instructions2}\n
"""
```

Output of system message generation chain:

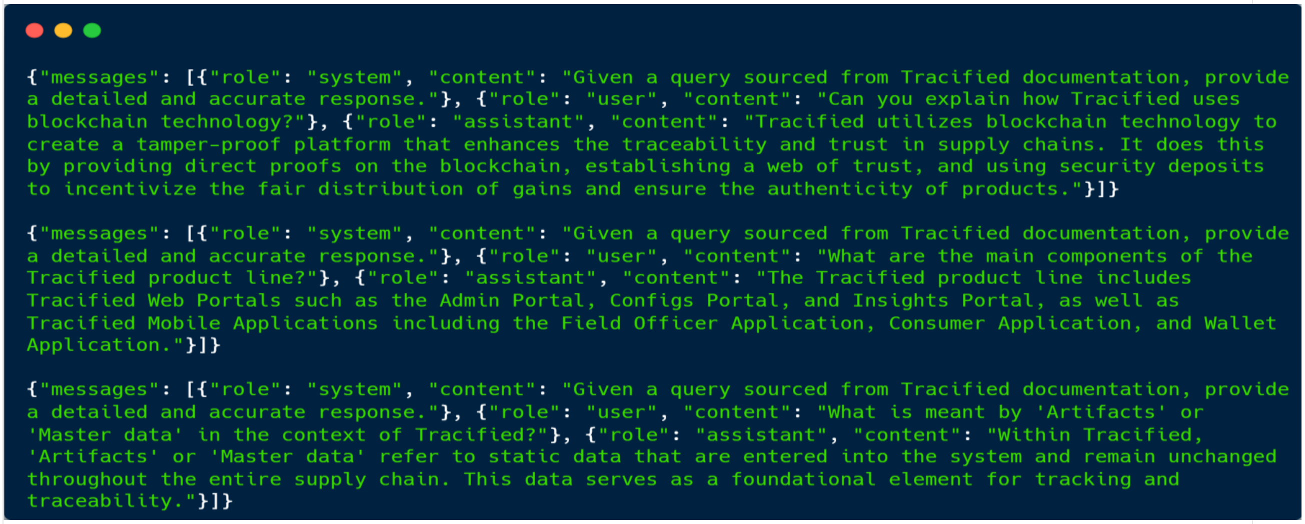


```
system_message='Given a query sourced from Tracified documentation, provide a detailed and accurate response.'
```

3.Fine-Tuning:

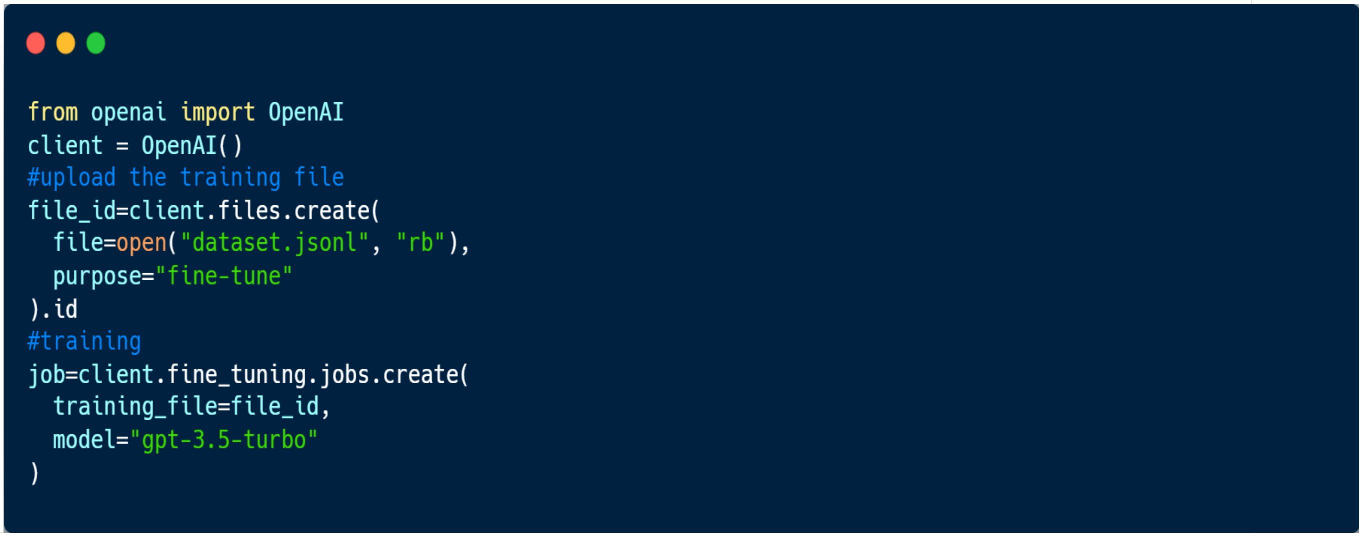
Prior to fine-tuning the model, it is necessary to format the dataset in the following manner For the OpenAI API, the data must be stored in jsonl format.:

Prepared training dataset for fine-tunning:



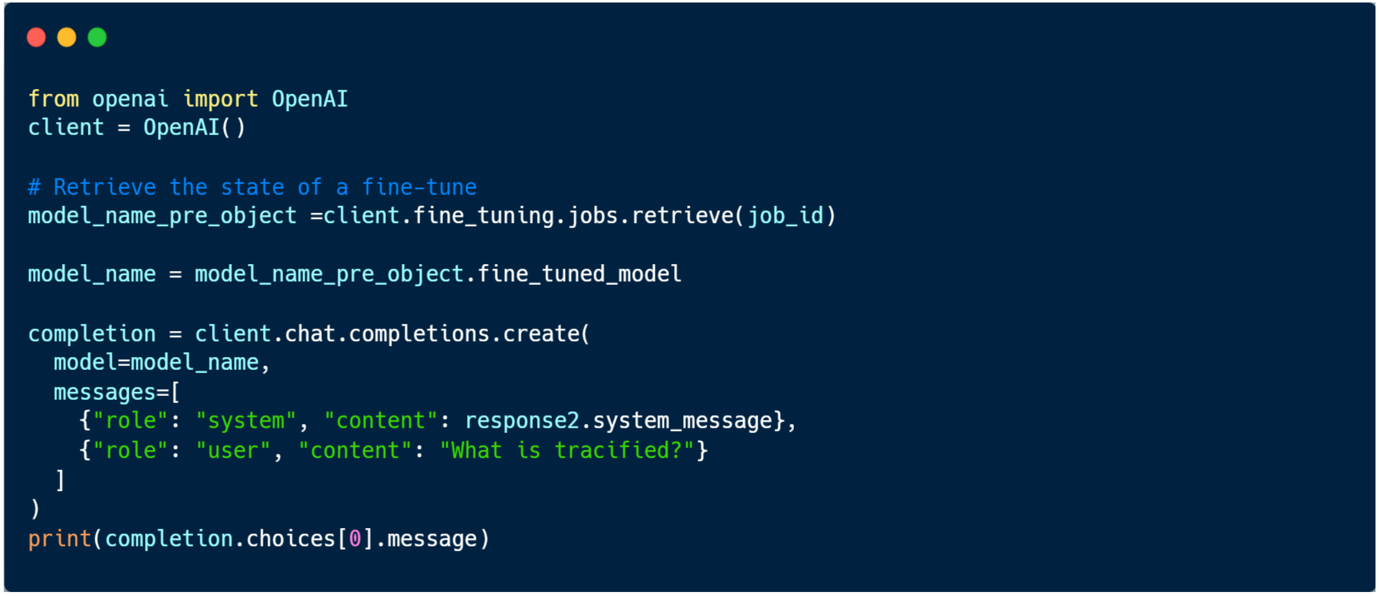
```
{"messages": [{"role": "system", "content": "Given a query sourced from Tracified documentation, provide
a detailed and accurate response."}, {"role": "user", "content": "Can you explain how Tracified uses
blockchain technology?"}, {"role": "assistant", "content": "Tracified utilizes blockchain technology to
create a tamper-proof platform that enhances the traceability and trust in supply chains. It does this
by providing direct proofs on the blockchain, establishing a web of trust, and using security deposits
to incentivize the fair distribution of gains and ensure the authenticity of products."}]}

{"messages": [{"role": "system", "content": "Given a query sourced from Tracified documentation, provide
a detailed and accurate response."}, {"role": "user", "content": "What are the main components of the
Tracified product line?"}, {"role": "assistant", "content": "The Tracified product line includes
Tracified Web Portals such as the Admin Portal, Configs Portal, and Insights Portal, as well as
Tracified Mobile Applications including the Field Officer Application, Consumer Application, and Wallet
Application."}]}

{"messages": [{"role": "system", "content": "Given a query sourced from Tracified documentation, provide
a detailed and accurate response."}, {"role": "user", "content": "What is meant by 'Artifacts' or
'Master data' in the context of Tracified?"}, {"role": "assistant", "content": "Within Tracified,
'Artifacts' or 'Master data' refer to static data that are entered into the system and remain unchanged
throughout the entire supply chain. This data serves as a foundational element for tracking and
traceability."}]}
```

Training:Fine-tuning using the generated dataset:



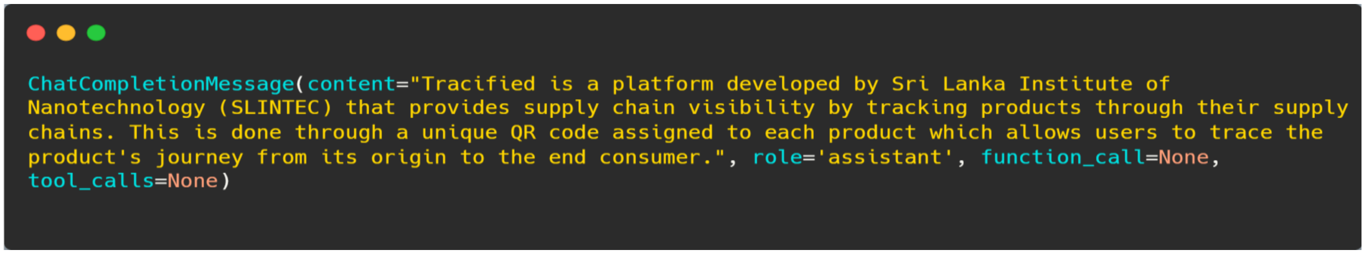
```
from openai import OpenAI
client = OpenAI()
#upload the training file
file_id=client.files.create(
  file=open("dataset.jsonl", "rb"),
  purpose="fine-tune"
).id
#training
job=client.fine_tuning.jobs.create(
  training_file=file_id,
  model="gpt-3.5-turbo"
)
```

Let's try it out!



```
from openai import OpenAI
client = OpenAI()

# Retrieve the state of a fine-tune
model_name_pre_object =client.fine_tuning.jobs.retrieve(job_id)

model_name = model_name_pre_object.fine_tuned_model

completion = client.chat.completions.create(
  model=model_name,
  messages=[
    {"role": "system", "content": response2.system_message},
    {"role": "user", "content": "What is tracified?"}
  ]
)
print(completion.choices[0].message)
```

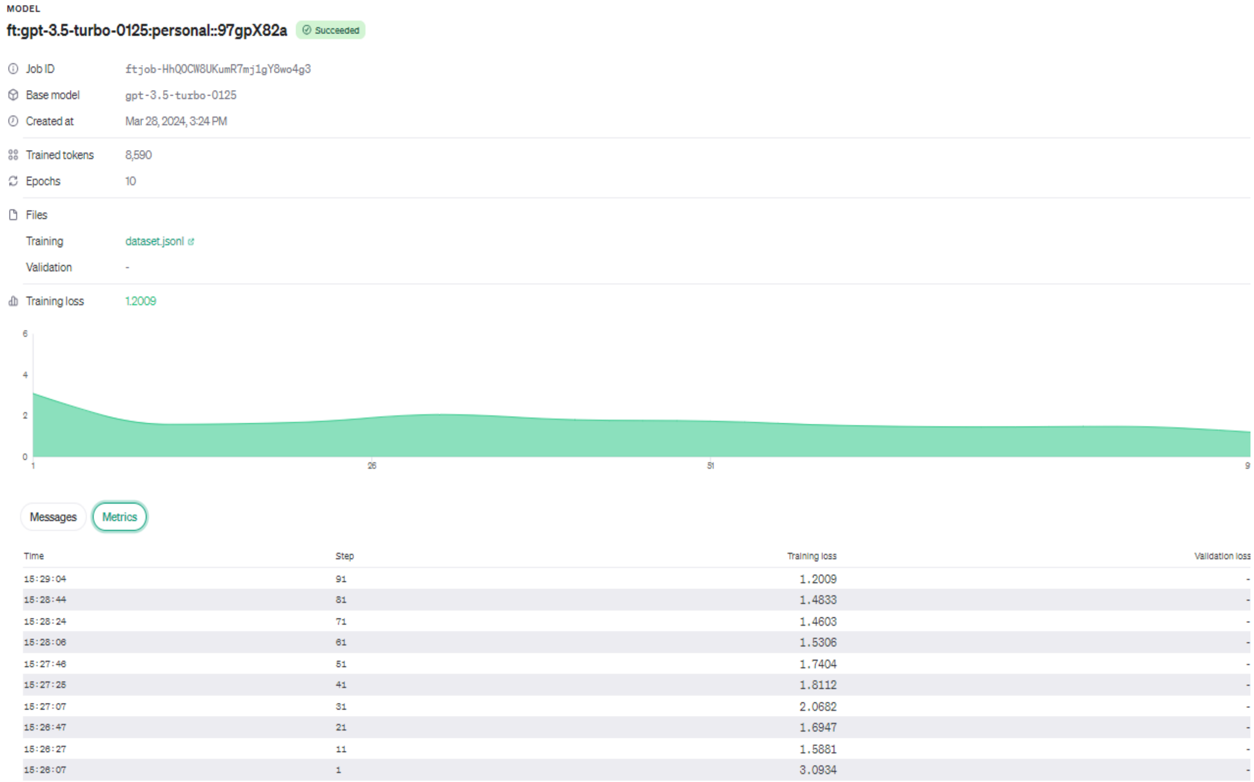
Output of our finetuned model:



```
ChatCompletionMessage(content="Tracified is a platform developed by Sri Lanka Institute of
Nanotechnology (SLINTEC) that provides supply chain visibility by tracking products through their supply
chains. This is done through a unique QR code assigned to each product which allows users to trace the
product's journey from its origin to the end consumer.", role='assistant', function_call=None,
tool_calls=None)
```

Analyze fine-tuned model

OpenAI offers key training metrics such as training loss, token accuracy for training, test loss, and test token accuracy. These metrics help ensure training is progressing as expected, with loss decreasing and token accuracy improving.



MODEL

ft:gpt-3.5-turbo-0125:personal::97gpX82a

Job ID	ftjob-HhQOCW8UKumR7mj1gY8wo4g3
Base model	gpt-3.5-turbo-0125
Created at	Mar 28, 2024, 3:24 PM
Trained tokens	8,590
Epochs	10
Files	
Training	dataset.jsonl
Validation	-
Training loss	1.2009

Time	Step	Training loss	Validation loss
15:29:04	91	1.2009	-
15:28:44	81	1.4833	-
15:28:24	71	1.4603	-
15:28:06	61	1.5306	-
15:27:46	51	1.7404	-
15:27:25	41	1.8112	-
15:27:07	31	2.0682	-
15:26:47	21	1.6947	-
15:26:27	11	1.5881	-
15:26:07	1	3.0934	-

Note: Model performance can be enhanced by generating additional data samples using more document content. I utilized this solely for demonstration purposes, so please do not place too much emphasis on accuracy.