**What was the training for?**

Fine-tuned GPT-2, a powerful language model, on your custom dataset of fantasy-themed names and titles like:

* AshBorn the Molten
* BlazeClawKnight
* DragonFlame the Charred
* SmolderStormChampion
* WildWrathWarrior

**Goal of Fine-tuning**

Trained GPT-2 to learn and generate similar fantasy names, titles, and combinations by:

* Understanding common prefixes (Ash, Blaze, Cinder, Dragon, Fury, etc.)
* Combining them with roles (Hunter, Reaper, Knight, etc.)
* And adding epic adjectives (the Charred, the Infernal, the Blazing)

**Why fine-tune GPT-2 on this?**

Fine-tuning helps GPT-2 become *specialized* in your domain. It allows:

* Higher quality, thematically consistent outputs
* Less randomness, more fitting names
* Better autocomplete for fantasy RPGs, games.

Fantasy Name Generator (GPT-2 Fine-Tuned)

This project fine-tunes a GPT-2 language model to generate fantasy names (for characters, places, races, etc.) using a custom dataset.

**It includes:**

Training on a fantasy names dataset

Saving the model

Deploying a FastAPI-based API to generate new names on the fly

**What This Project Does ?**

Fine-tunes a GPT-2 model on a list of ~1000 fantasy-style names

Saves the trained model locally

Deploys an API with endpoints to:

Check status (/status)

Generate new names based on prompts (/generate?prompt=Ash)

**How to Run It**

Local Setup

Bash,Copy

git clone https://github.com/samyak44/fantasy-gpt2-api.git

cd fantasy-gpt2-api

pip install -r requirements.txt

Ensure you’ve already fine-tuned the model and saved it under fantasy-gpt2-model/.

The model was about 1.5 gb therefore need to download the model from google colab file.

Then run:

bash

Copy

uvicorn app:app --reload

Now go to:

http://127.0.0.1:8000/status

<http://127.0.0.1:8000/generate?prompt=Ash>

2. Google Colab

You can follow the instruction from Google colab.

* Upload your dataset (e.g., fantasy\_names\_1000.txt)
* Run fine-tuning code (see Colab notebook)
* Download or save model to Google Drive
* Now will be able to run the model on Vs-code and check the API.

**What I'd Improve with More Time/Resources**

Use LoRA or PEFT to reduce training cost

Add UI (streamlit or HTML) to make a front-end.

Expose model on Hugging Face Spaces or Render.

Include more diverse datasets (e.g., race names, city names, descriptions)