

# ***Cura Medica – AI Medical Consultant Chatbot***

**Name:** Joel Mugisha

**Date:** October 2025

**Course:** Machine Learning Techniques I

Links

App demo: [Cura Medica – Health Symptom Assistant](#)

Github repo: [jmugisha1/cura-medica-source](#)

Video link: [cura-medica](#)

---

## **2. Abstract**

The Cura Medica chatbot is an AI-powered medical assistant designed to help users identify possible diseases based on their symptoms and recommend the right medical specialist for consultation.

The model was developed and fine-tuned on Gemma 3 270M LLM

to understand medical queries and generate accurate, domain-specific responses.

Evaluation was done with BLEU and Perplexity metrics and they showed balanced and fair performance in predicting the likelihood of a disease and point towards the right specialist , in first-level medical consultation.

---

## **3. Introduction**

### **Problem Statement:**

Accessing the right healthcare professional is often time-consuming and confusing for patients who lack medical knowledge. Many start with general consultations that take long, waste money and time. This leads to patients often experiencing difficulty identifying which doctor to consult when symptoms appear. This causes inefficiency and delays in treatment.

So solving such a problem is beneficial because accurate disease identification and specialist recommendations can improve patient outcomes and reduce the in general

In the future the same solution design for the general public's healthcare will become useful also for doctors to make it more disease prediction reducing time wastage inaccurate results among others .

### **Objectives:**

1. Analyze symptoms and predict potential diseases.
2. Recommend the most suitable medical specialist.  
Serve as a preliminary AI consultant for users before visiting a doctor.

---

## **4. Dataset Collection & Preprocessing**

**Data Sources:**Datasets were collected and combined from multiple sources which were latter combined into a single dataset for training :

1. [FreedomIntelligence/Disease Database · Datasets at Hugging Face](#)
2. [Doctor's Specialty Recommendation](#)
3. [zaimasohail/symptom-disease-dataset · Datasets at Hugging Face](#)

**Web Scrapping:** apart from the traditional data collection some data were collected using web scraping to extract medical or in this case doctor specialists and add them to the initial dataset which were only the mapping of symptoms to diseases.

### **Files Used:**

1. cura-medica-data.ipynb – Data cleaning, normalization, and merging
2. generate.ipynb – Web scraping and doctor-specialist mapping
3. cura-medica-v1.ipynb – Final model training and chatbot implementation

### **Preprocessing Steps and tools :**

1. Cleaned and removed missing or duplicate records.
  2. Normalized text data (lowercasing, removing noise).
  3. Tokenized using the model tokenizer (Gemma 3).
  4. Combined datasets into a unified symptom–disease–doctor format.
  5. Libraries/Tools: Python, Pandas, BeautifulSoup, Hugging Face Datasets, TensorFlow, Unsloth
-

## 5. Model Selection & Fine-Tuning: gemma 3 270m via unsloth

**Reason for Selection:** the gemma 3 270m is a Lightweight generative model suitable trained on only 270m tokens and was about 550m in disk size meaning fine tuning would take a short period of time and be source efficient .

### Implementation Summary:

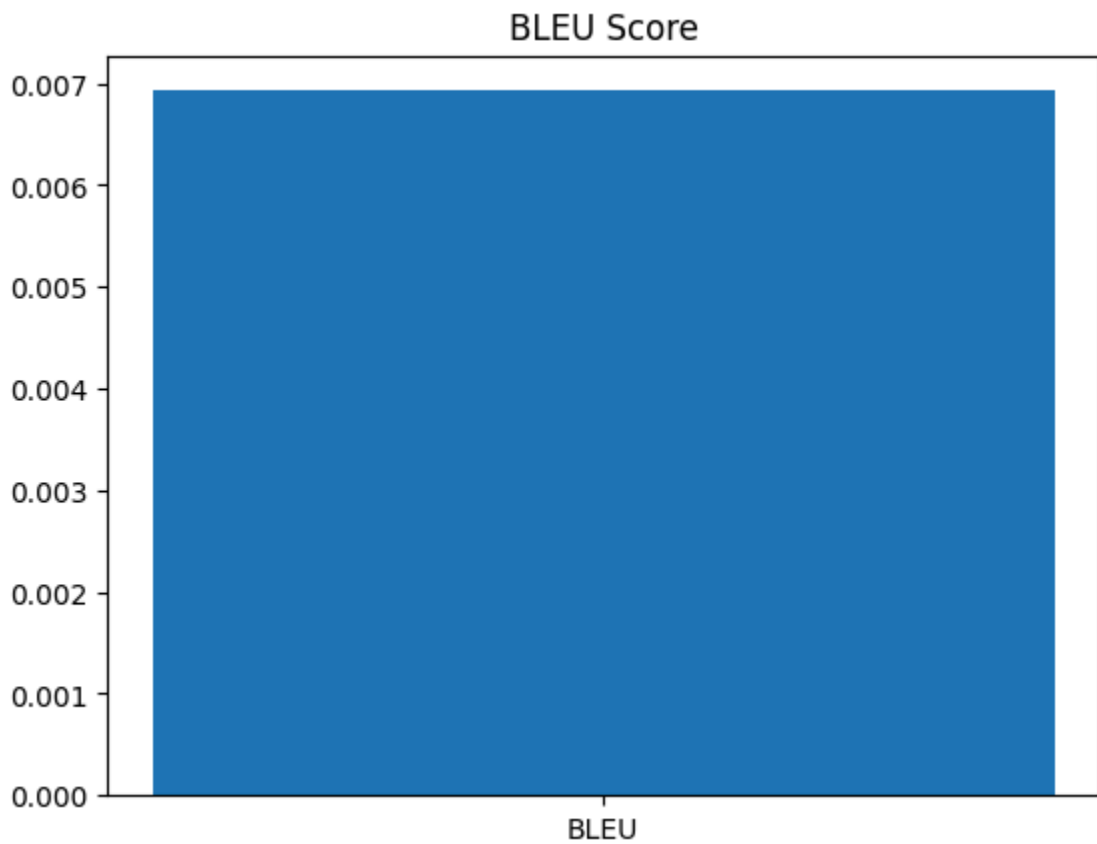
1. Loaded base model from unsloth/gemma-3-270m
2. Added LoRA adapters fine-tuning.
3. Trained on 80% of the dataset, validated on 20%.
4. Tokenized using formatting function
5. And added trainer with different hyperparameters
  - a. **model & tokenizer** – The base language model and tokenizer being fine-tuned.
  - b. **train\_dataset** – The processed training data used for fine-tuning.
  - c. **num\_train\_epochs=3** – Number of full passes through the dataset.
  - d. **learning\_rate=2e-4** – Speed at which the model learns.
  - e. **optim="adamw\_8bit"** – This sets the optimizer used for training.
  - f. **warmup\_steps=50** – Gradually increases LR at the start for stability.

### Hyperparameters:

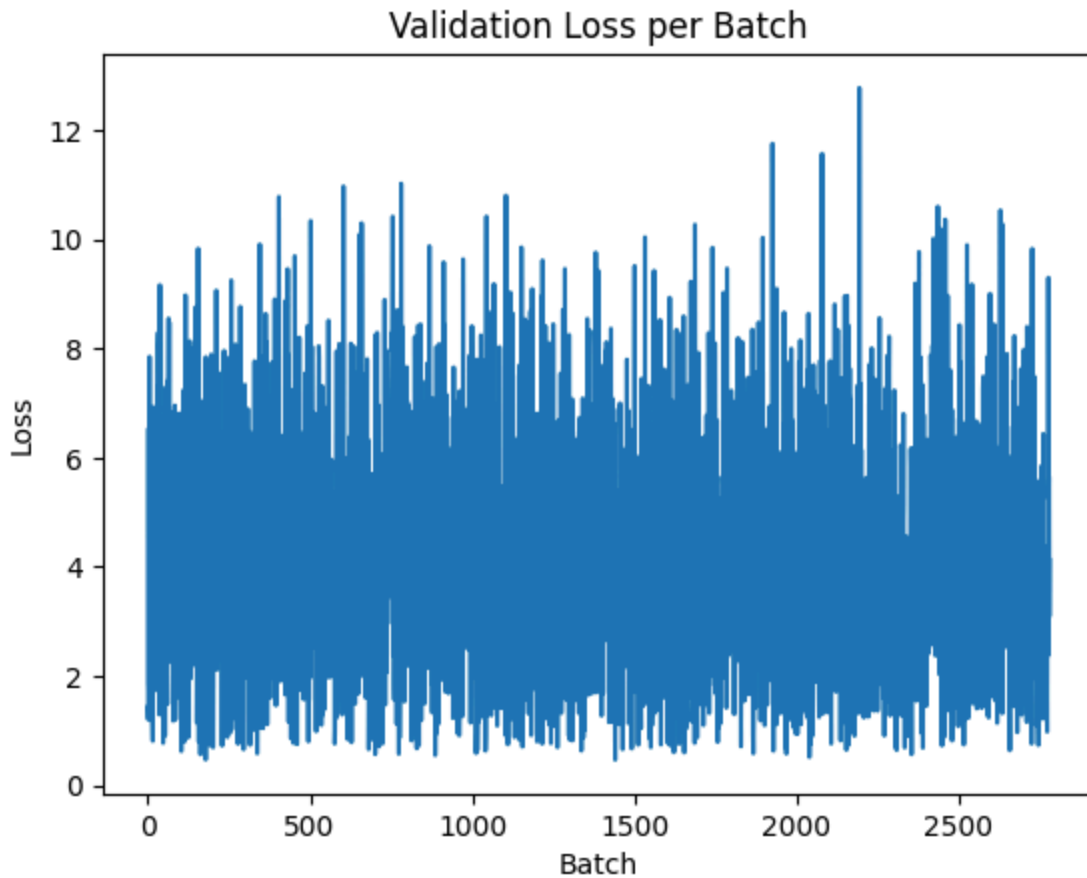
Parameter	Value
Learning Rate	2e-4
Batch Size	4
Epochs	3
Optimizer	AdamW (8-bit)
Gradient Accumulation	8
Warmup Steps	50

### Evaluation Metrics Results and visualization:

- **BLEU Score:** Evaluated text accuracy.: BLEU Score: 0.0069273106279722635



- **Perplexity:** Measured model fluency and confidence. Perplexity: 57.43403602718883



---

## 5. Deployment

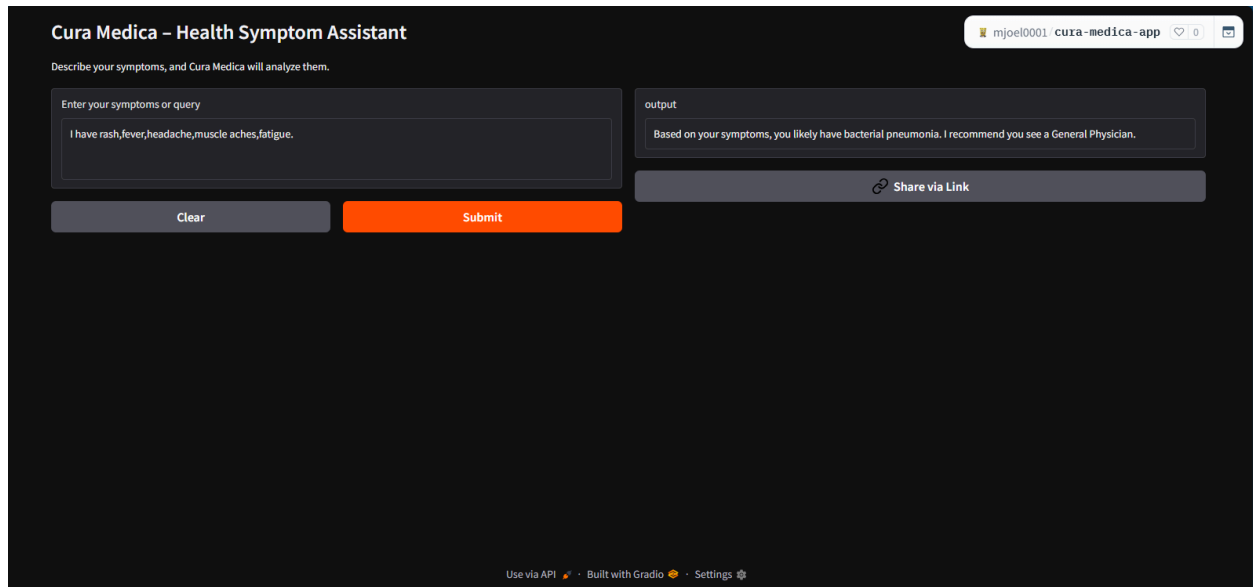
The chatbot was deployed using Gradio integrated with Hugging Face spaces

This setup provided a lightweight and user-friendly web interface where users can interact with the chatbot directly in their browser.

This was archived app.py what creates gradio app and loads the model

### User Interaction Workflow:

1. The user enters a query related to the medical domain in the Gradio input box.
2. The chatbot processes the query using the fine-tuned Transformer model.
3. The response is generated and displayed instantly below the input field.
4. Users can continue chatting or reset the interface for a new session.
5. Some sample symptoms from training dataset
6. For usage use the user content without double ""



```
{"messages": [{"role": "user", "content": "I have fever,cough,sore throat,runny or stuffy nose,muscle aches,headache,fatigue."}, {"role": "model", "content": "Based on your symptoms, you likely have flu. I recommend you see a General Physician."}]}
```

```
{"messages": [{"role": "user", "content": "I have cough,mucus production,shortness of breath,chest pain."}, {"role": "model", "content": "Based on your symptoms, you likely have bronchitis. I recommend you see a General Physician."}]}
```

```
{"messages": [{"role": "user", "content": "I have fever,cough,shortness of breath,chest pain,fatigue."}, {"role": "model", "content": "Based on your symptoms, you likely have pneumonia. I recommend you see a General Physician."}]}
```

---

7. Demo Video Summary The demo video (5–10 minutes) walks through the entire project lifecycle:

1. **Data Collection:** Explanation of medical conversational dataset.
2. **Data Processing:** Cleaning, tokenization, and formatting for the model.
3. **Model Creation:** Fine-tuning the Transformer model (Hugging Face + unsloth for gemma 3).
4. **Model Evaluation:** Performance metrics (BLEU & Perplexity).  
**Model Deployment:** Integration into Gradio for user access.

5. **Chatbot Interaction:** Live demonstration of chatbot answering domain-specific queries using the Gradio interface.
- 

## 8. Conclusion & Future Work

### Summary of Outcomes:

The project successfully implemented a **domain-specific medical chatbot** using Transformer model (gemma 3) and Gradio. The chatbot can handle user queries effectively, demonstrating the potential of fine-tuned language models in healthcare-related NLP applications.

### Limitations:

1. Responses may vary in accuracy for complex medical queries.
2. Limited dataset size may restrict generalization.
3. The chatbot currently handles text input only.

### Future Work:

1. Integrate image classification for analyzing medical images (e.g., X-rays).
2. Expand dataset with more medical dialogues.
3. Improve response confidence estimation and feedback handling.