# My LLM Project

## Introduction

This project is a custom-built Large Language Model (LLM) designed to answer predefined questions based on a JSON dataset. The model is fine-tuned to respond only to queries present in the dataset, ensuring relevant and accurate responses. If a question is not available, the chatbot informs the user that the topic is still under development.

## Project Features

• \*\*Fine-Tuned LLM\*\*: The model is trained using a selected dataset for domain-specific knowledge.  
• \*\*Custom JSON Dataset\*\*: Answers are fetched from a predefined dataset stored in JSON format.  
• \*\*User-Friendly Interface\*\*: The application provides a simple and interactive UI for user interaction.  
• \*\*Filtered Responses\*\*: The chatbot only responds to known queries and provides a default message for unknown ones.  
• \*\*Gradio Web UI\*\*: The project uses Gradio to create a lightweight and easy-to-use web interface.  
• \*\*Local Execution\*\*: The model runs locally without relying on external APIs, ensuring privacy and security.

## How It Works

1. \*\*Loading the Model and Tokenizer\*\*: The fine-tuned model and tokenizer are loaded from the local directory.  
2. \*\*Loading the JSON Dataset\*\*: The predefined question-answer pairs are loaded into a dictionary for quick lookup.  
3. \*\*Processing User Input\*\*: The input is normalized by converting it to lowercase and removing punctuation.  
4. \*\*Matching Against Dataset\*\*: The chatbot checks if the cleaned input matches any predefined questions.  
5. \*\*Returning a Response\*\*: If a match is found, the corresponding answer is returned. Otherwise, a default message is displayed.  
6. \*\*Gradio UI\*\*: The user interacts with the chatbot through a Gradio-powered web interface.

## Technologies Used

• \*\*Python\*\* - Main programming language  
• \*\*Hugging Face Transformers\*\* - Used for fine-tuning the LLM  
• \*\*Gradio\*\* - For creating the user interface  
• \*\*JSON\*\* - For storing question-answer pairs  
• \*\*PyTorch\*\* - For model training and inference  
• \*\*GitHub\*\* - For version control and project management

## Hosting and Deployment

The application can be run locally on a personal computer. It is currently set up for execution in a local environment, but it can be hosted using services like Vercel, Hugging Face Spaces, or a cloud server. Since the model is large, Git Large File Storage (LFS) is used for handling model weights in GitHub.

## Future Improvements

• Expanding the dataset with more domain-specific knowledge.  
• Improving model efficiency to reduce memory usage and inference time.  
• Adding a feedback mechanism to refine responses.  
• Exploring deployment options for cloud-based hosting.  
• Enhancing UI features to improve user interaction.

## Conclusion

This project demonstrates the power of fine-tuned LLMs for domain-specific applications. By using a predefined dataset, it ensures that responses are accurate and reliable. Future developments will focus on enhancing model capabilities and making deployment more scalable. The project serves as a foundation for building AI-powered assistants tailored to specific business or personal needs.