# LoreBot 📜🤖

LoreBot is a stateful conversational AI agent that allows users to interact with fictional characters grounded in specific narrative context. Using **Retrieval-Augmented Generation (RAG)** and **LangGraph**, it ensures that the AI stays strictly "in-character" by referencing uploaded lore documents (PDFs) instead of relying on generic training data.

## 🚀 Key Features

* **Dynamic Character Persona**: Input any character name to instantly set the persona.
* **RAG-Powered Lore Extraction**: Upload a PDF of story notes, world-building, or character history to provide the bot with a "memory."
* **Stateful Conversation**: Built with **LangGraph** to maintain complex state and strictly enforce character boundaries.
* **Modern AI Stack**: Utilizes Mistral-7B via Hugging Face Endpoints for high-quality, era-appropriate dialogue.
* **Interactive UI**: A clean **Streamlit** interface for seamless chatting and document indexing.

## 🛠️ Tech Stack

* **Frameworks**: LangGraph, LangChain
* **LLM**: Mistral-7B-Instruct-v0.2 (via Hugging Face)
* **Vector Store**: FAISS
* **Embeddings**: Sentence Transformers (all-MiniLM-L6-v2)
* **Frontend**: Streamlit
* **Deployment & Orchestration**: Python, Dotenv, MemorySaver

## 📁 Project Structure

Lorebot/  
├── .venv/ # Virtual environment  
├── .env # API keys (HF\_TOKEN)  
├── .gitignore # Excludes private data  
├── app.py # Main application and Graph logic  
├── requirements.txt # Dependencies  
└── temp.pdf # Temporary processing storage

## ⚙️ Setup Instructions

### 1. Prerequisites

* Python 3.9+
* A Hugging Face API Token (with access to Mistral models)

### 2. Installation

# Clone the repository  
git clone [https://github.com/YOUR\_USERNAME/Lorebot.git](https://github.com/YOUR\_USERNAME/Lorebot.git)  
cd Lorebot  
  
# Set up virtual environment  
python -m venv .venv  
source .venv/bin/activate # On Windows: .venv\Scripts\activate  
  
# Install dependencies  
pip install -r requirements.txt

### 3. Environment Variables

Create a .env file in the root directory:

HF\_TOKEN=your\_huggingface\_api\_token\_here

### 4. Running the App

streamlit run app.py

## 🧠 How It Works

1. **Indexing**: The uploaded PDF is split into chunks and converted into vector embeddings using FAISS.
2. **Retrieval**: When a user inputs a message, the system searches the lore for the top 3 most relevant context snippets.
3. **Generation**: The character name, retrieved context, and chat history are passed to Mistral-7B with a custom prompt to generate a character-appropriate response.

*Created by [Your Name]*