🏨 Hotel Booking Analytics with RAG System

A full-stack, intelligent analytics system for hotel booking data — combining traditional metrics with Retrieval-Augmented Generation (RAG) to answer natural language questions about trends, revenue, and guest behavior.

📦 Features

- 📊 Automated analytics generation from CSV data
- 🤖 Natural language Q&A powered by Mistral-7B (via HuggingFace)
- 🔍 Semantic search over custom knowledge base using FAISS
- 📈 Interactive dashboards and visualizations with Streamlit
- 🌐 REST API via FastAPI for backend integration

Setup Instructions

1. Clone the Repository

```bash
git clone https://github.com/your-username/hotel-booking-analytics-rag.git
cd hotel-booking-analytics-rag
```

2. Install Dependencies

Create a virtual environment (recommended):

```
python -m venv venv
source venv/bin/activate  # or venv\Scripts\activate on Windows
```

Install all required packages:

```
pip install -r requirements.txt
```

> **Note**: Ensure Python 3.10 or 3.11 is installed.

🚀 Running the App

Option 1: Streamlit UI

streamlit run app.py
Or this link-
https://buyogo-llm-powered-booking-analytics-app-system-mmq6jzvhqgr2el.streamlit.app/

- Upload `hotel_bookings.csv`
- Browse dashboards or ask natural language questions (e.g., *"Which month had the highest cancellations?"*)

Option 2: FastAPI Backend

uvicorn main:app --reload --port 8000

API Endpoints:

- `POST /upload-csv`: Upload hotel booking data
- `GET /analytics`: Get computed metrics (ADR, cancellations, etc.)
- `POST /query`: Ask a natural language question

API Root:
This confirms the app is running:
`http://127.0.0.1:8080/`

Analytics Endpoint:

To get hotel booking analytics:
`http://127.0.0.1:8080/analytics`
(This should be accessed via POST, and you need to provide a body with filters and visualizations option)

Ask Endpoint:
To ask questions to the RAG system:
`http://127.0.0.1:8080/ask`
(This also requires a POST request with the question and options like including sources)

Visualizations:
You can view the generated visualizations (assuming the images are available in the static folder):

- `http://127.0.0.1:8080/visualizations/monthly_adr`
- `http://127.0.0.1:8080/visualizations/cancellation_by_country`

Health Check:
To check the health status of your application:
`http://127.0.0.1:8080/health`

🧪 Sample Questions

- What is the average daily rate by month?
- Which countries have the highest cancellation rates?
- What are the peak check-in days?
- How much revenue was generated in August?

🔐 Environment Variables

Create a `.env` file in the root directory:

HUGGINGFACEHUB_API_TOKEN=your_hf_token

Optional: You can cache the vector store to skip repeated embedding generation.

🧱 Folder Structure

```
.
├── app.py                 # Streamlit frontend
├── api.py                 # FastAPI backend
├── HotelBookingPipeline.py # Data preprocessing and analytics
├── HotelBookingRAG.py      # Embedding + Q&A logic
```

```
├── data/                # Uploaded CSVs and processed outputs
├── images/              # Visualizations
├── faiss_store/         # Vector DB
├── requirements.txt
└── README.md
```

🧭 Roadmap

- Add live integration with hotel PMS APIs
- Enable real-time dashboards via WebSocket
- Replace FAISS with Pinecone for scale
- Add user login/authentication

Contact

📧sharmakanishka1604@gmail.com
🔗 LinkedIn
