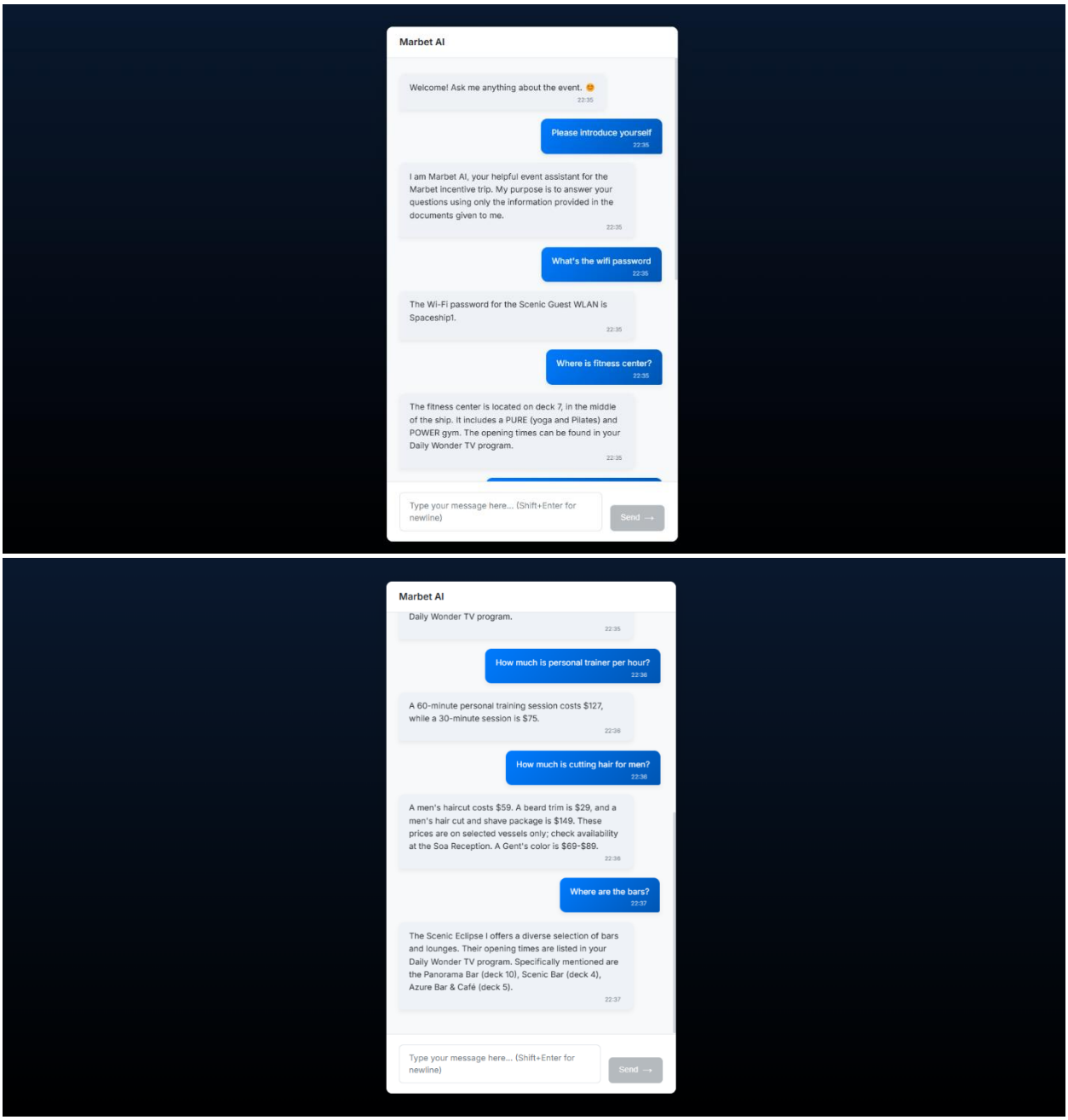
Marbet AI Assistant Report

1. Demo Screenshot

Below you can see the screenshot of a sample chat.



2. Introduction

This project developed a specialized RAG chatbot to assist Marbet incentive trip attendees by providing accurate information drawn exclusively from event documentation (schedules, packing lists, policies).

The solution offers:

- Flexible model usage: Either local Ollama (for privacy) or Google Gemini (for enhanced capabilities)
- Easy configuration through environment variables and config file
- LangChain for orchestration and ChromaDB for vector storage
- Enhanced guest experience while maintaining data privacy

The system features PDF ingestion, intelligent chunking, vector embeddings, context-aware retrieval, chat history awareness, grounded responses, and source attribution - all accessible through both CLI and web interfaces.

3. Chatbot Design & Prompt Engineering

The RAG pipeline processes queries through seven key steps:

1. User query + chat history intake
2. Context-aware query reformulation
3. Relevant document retrieval from ChromaDB
4. Context formatting with source metadata
5. Response generation using only provided information
6. Citation processing for attribution
7. Final response delivery to user

The system includes two carefully engineered prompt components:

- A contextualizing prompt that transforms chat history into standalone queries
- A QA prompt that defines the assistant's persona while strictly limiting responses to provided context

The system works with either local or cloud models:

Component	Ollama (Local)	Gemini (Cloud)
LLM	deepseek-r1:32b	gemini-1.5-flash-latest
Embeddings	mxbai-embed-large	models/embedding-001

For factual accuracy, the temperature is set to 0.0 with MMR retrieval implementation to balance relevance with diversity in results.

4. Knowledge Base Structuring

The project implements a robust document processing pipeline:

Document Loading:

- Enhanced PDF parsing using UnstructuredPDFLoader with hi-res strategy
- Metadata enrichment with page numbers and source information

Text Processing:

- Recursive character splitting with configurable chunk size (default: 128) and overlap (20)
- Start index tracking for better source tracing

Vector Storage:

- Local ChromaDB persistence
- Configurable embeddings based on selected model
- Smart indexing with rebuild capabilities when needed

5. Conclusion & Future Improvements

The RAG chatbot successfully delivers on its core objective: providing accurate event information using only authorized materials. Its flexible architecture allows deployment in various privacy contexts while maintaining consistent performance.

Recommendations for Enhancement:

Priority	Improvement
High	Fine-tune document extraction for better table handling
High	Implement hybrid search combining semantic and keyword approaches
Medium	Add document-specific filtering options
Medium	Create systematic evaluation framework
Medium	Enhance UI with source previews and feedback mechanisms
Low	Improve error handling and monitoring
Low	Benchmark additional models for performance comparison

With these improvements, the system could deliver even more precise responses while maintaining its core strengths of accuracy and privacy.