Safar Travel AI Agent – Project Overview Report

Project Goal

The goal of this project is to design and implement an operational AI agent for domestic travel services in Iran. The agent supports ticket booking, cancellation, booking information retrieval, and travel destination suggestions. This system goes beyond a simple chatbot and leverages Large Language Models (LLMs), tool calling, Retrieval-Augmented Generation (RAG), and an interactive user interface.

System Architecture

The project follows a layered architecture consisting of a Chainlit-based user interface, an orchestration layer (AgentManager), an LLM backend, and a modular tools layer. This separation of concerns improves maintainability, scalability, and clarity of the system.

AgentManager (Core Orchestrator)

AgentManager is responsible for managing conversation history, deciding when to invoke tools, handling multi-step reasoning, and safely managing errors. This component is implemented at a professional level and is suitable for both MVP and production-ready systems.

Tools Layer

The tools layer provides all executable actions available to the agent. It includes a simulated ticket management system, a Retrieval-Augmented Generation (RAG) module for company policies, and a destination recommendation tool powered by LLM reasoning. Each tool is exposed to the agent via explicit JSON schemas.

RAG and Knowledge Base

The RAG system is implemented using ChromaDB and is used to answer questions related to company policies such as refunds, cancellations, and baggage allowances. The system is fail-safe and avoids hallucination when no data is available.

System Prompt and Agent Personality

The system prompt defines the agent’s identity, tone, behavioral rules, tool usage guidelines, and language support. A friendly Shirazi-inspired Persian tone is used as a brand differentiator, while maintaining clarity and professionalism. The prompt is designed to minimize hallucinations and enforce correct tool usage.

User Interface and Experience

The user interface is implemented using Chainlit and includes session management, a thinking indicator, and structured message updates. The UI is simple, responsive, and suitable for iterative enhancement.

Stability, Safety, and Error Handling

The system includes robust error handling mechanisms such as environment validation, API error management, loop limits for tool calls, and strict data safety rules. These measures make the project suitable for real-world usage.

Overall Evaluation

This project represents a real-world AI Agent MVP. From architecture and orchestration to prompt engineering and tool integration, the system demonstrates a strong understanding of modern AI agent design principles.