

Intelligent Travel & Visa Assistance Platform

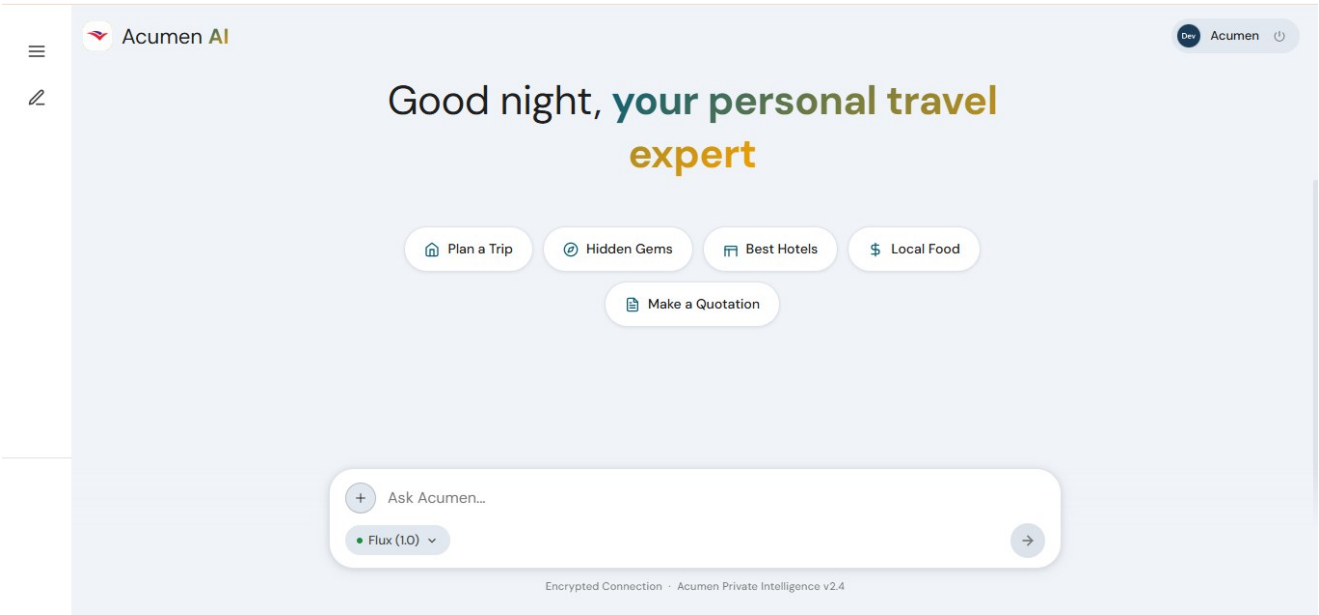


Figure 1. Actual Acumen AI main workspace screenshot supplied for the case study.

Role AI Engineer - Intern	Organisation Acumen Travels (India) Private Limited
Core Interface Conversational AI workspace with travel workflows	Architecture React.js + Vite frontend, FastAPI + Python backend

This case study is grounded in the supplied Acumen AI project documentation and the supplied application screenshot. Project claims are kept aligned with the documented implementation and testing scope.

01 PRODUCT CONTEXT & PROBLEM

Why Acumen AI was developed

Travel planning and visa-related assistance can involve destination selection, itinerary planning, accommodation, local attractions, flight information, visa-related requirements and quotation generation. The project documentation identifies a fragmented workflow in which users may have to consult multiple platforms or communication channels and manually organize information before taking the next step.

Acumen AI was designed to bring these interactions into a unified web-based environment. The intended experience combines conversational natural-language interaction with structured travel workflows, so that a user can begin with an open-ended request and continue into specific tasks such as itinerary generation, flight exploration, quotation preparation and document delivery.

Core product objective

Conversational access Allow users to describe travel or visa requirements in natural language.	AI resilience Use multiple configured AI services with a fallback-oriented routing path.
Workflow continuity Maintain relevant context across multi-step interactions.	Travel assistance Support planning and recommendation-oriented travel workflows.
Structured quotations Collect travel requirements and transform them into structured quotation data.	Document output Generate travel quotation/package documents in PDF form.
Secure access Provide authentication and session-related functionality.	Integrated services Coordinate databases, external APIs and document services through the backend.

What makes the problem technically interesting

The challenge is not only generating a useful AI response. The system must connect natural-language input with downstream application processing while keeping provider-specific AI and external-service logic behind a backend boundary. This creates an engineering problem around request routing, persistence, authentication, error handling and the conversion of conversational information into structured outputs.

02 SYSTEM ARCHITECTURE

Layered architecture and request flow

The documented implementation follows a client-server architecture. React.js provides the user-facing application interface, Vite supports the frontend development/build workflow, and FastAPI with Python acts as the central application-processing layer. Persistent data and external-service communication are kept behind the backend boundary.

USER INTERFACE React.js + Vite Conversational workspace Navigation · Itinerary · Flights · Documents · Billing	BACKEND PROCESSING FastAPI + Python Request handling · validation · application logic · service orchestration
AI SERVICE LAYER Configured AI services Provider routing · fallback path · conversational response processing	DATA LAYER Persistent application data Backend-controlled storage/retrieval · session/application context
DOCUMENT WORKFLOW Quotation processing Structured quotation data -> document generation -> preview/download	EXTERNAL SERVICES Travel/application-specific services Controlled backend communication and error handling

Figure 2. High-level architecture representation based on the supplied project documentation.

AI routing and fallback

AI-related requests originate in the frontend and are transmitted to the FastAPI backend. The backend routes the request through the configured AI service architecture. Where a preferred processing path cannot complete the request, the documented fallback-oriented architecture can attempt an alternative configured service. If no configured service can complete the request, a controlled error can be returned.

Backend as the integration boundary

External services are accessed through the backend rather than directly from the browser. This keeps provider-specific communication, sensitive configuration and service-level error handling outside the presentation layer and makes individual integrations easier to modify without redesigning the complete frontend.

03 IMPLEMENTATION & USER EXPERIENCE

From conversation to structured travel output

The implementation is organized around a persistent application shell and a central working area. The user can move between a new conversation, travel-planning workflows, flight exploration, documents, billing and settings without leaving the overall application context.

Main conversational workspace Provides the primary natural-language interaction surface and serves as the starting point for travel assistance.	Smart Itinerary Presents structured destination and day-wise travel information generated through the itinerary workflow.
Flight Exploration Displays flight information in a comparison-oriented interface with a selection flow.	Quotation workflow Collects travel information and converts it into structured quotation data before document generation.
Combined Travel Package Presents generated travel material through the document workflow with preview/download access.	Billing & Plans Shows usage/subscription information, including Free and Pro plan presentation.
Authentication Provides controlled access to the personalised application workspace.	Settings & theme Keeps account and interface preferences within the same application structure.

Functional workflow

User request -> React interface -> FastAPI backend -> selected workflow/service -> structured result -> frontend presentation. For a travel workflow, the result may become a day-wise itinerary, flight information, quotation data or a generated document. This layered flow keeps presentation, application logic and service-specific processing separate.

Implementation evidence

The project testing documentation reports successful end-to-end paths for authentication, navigation, AI chat, Smart Itinerary, Flight Exploration, quotation processing, PDF generation and Billing/Plans. It also records controlled handling for AI-provider failure, invalid or incomplete input, and unsuccessful processing scenarios.

04 TESTING, OUTCOME & FUTURE DIRECTION

What was evaluated

Test Area	Expected Outcome	Evaluation
Authentication	Access to the personalised application workspace	Passed
AI Chat	Travel-related request returns an AI response or workflow action	Passed
AI Fallback	Alternative configured processing path is attempted	Passed / controlled
Smart Itinerary	Structured destination and day-wise plan is displayed	Passed
Flight Exploration	Comparable flight information is presented	Passed
Quotation	Travel information is converted into structured quotation data	Passed
PDF Generation	Generated travel document is available	Passed
Billing & Plans	Plan and usage information is displayed	Passed

Current outcome

The documented evaluation positions Acumen AI as an integrated application rather than a collection of isolated pages. Authentication leads into the application workspace; AI requests produce responses; travel requests connect to itinerary or flight interfaces; and quotation requests can progress to document output. The architecture and testing approach therefore demonstrate integration across the frontend, backend, AI services, data operations and document workflows.

Future direction

The project documentation identifies deeper integration between conversational and transactional workflows as a practical next step. Other future directions include stronger observability around request latency and provider failures, deeper personalisation based on travel preferences and interaction history, broader real-time travel-service integrations, and continued improvement of reliability and scalability.

Professional summary

Acumen AI demonstrates a modular approach to AI-assisted travel software: conversational interaction at the front, backend-controlled orchestration in the middle, and structured travel/document workflows behind it. The result is a system designed to turn open-ended travel requests into practical, traceable application actions.

Source basis: supplied Acumen AI project report, including the documented project scope, architecture, implementation, testing, future scope and references, plus the supplied main-workspace screenshot.