IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

Optimized Travel Recommendation & Itinerary Generation

Nidhi Dhake[1], Shravani Mestry[2], Akshay Diwate[3], Nishant Gudade[4],

Dr. Siddharth Hariharan[5]

Student, Department of Computer Engineering, Terna Engineering College, Navi Mumbai, India[1,2,3,4]

Associate Professor, Department of Computer Engineering, Terna Engineering College, Navi Mumbai, India[5]

Abstract: The rapid growth of digital travel platforms has created an urgent need for intelligent, personalized travel recommendation systems that go beyond simple keyword-based search. This paper presents TripSona, a full-stack AI-powered Indian travel recommendation and itinerary planning system built on a microservices architecture. The system employs a hybrid recommendation engine that combines a trained Machine Learning model using collaborative and content-based filtering with real-time data from Google Places API and Booking.com API to generate personalized destination recommendations. For itinerary generation, the system introduces a novel Multi-Agent Architecture comprising six specialized Gemini AI agents — Day Planner, Weather, Budget, Food and Tips, Transport, and Coordinator — each responsible for a distinct aspect of trip planning. The system is trained and evaluated on a dataset of 2000 Indian travel records spanning 40 destinations across diverse categories including Beach, Adventure, Nature, Historical, and City. User preferences including age range, budget, trip duration, interests, purpose, health conditions, and cuisine preference are used as input features. Experimental results demonstrate that the hybrid recommendation approach achieves superior personalization accuracy compared to standalone ML or API-based approaches. The multi-agent itinerary system produces contextually rich, weather-aware, budget-conscious day-by-day travel plans with real venue data. The system is deployed as a React-based web application integrated with Firebase Authentication and PostgreSQL for persistent storage.

Keywords: Travel Recommendation System, Multi-Agent Architecture, Machine Learning, LangChain, Gemini AI, Google Places API, Microservices, Collaborative Filtering, Content-Based Filtering, Indian Tourism

I. INTRODUCTION

Tourism is one of the largest and fastest-growing industries globally, contributing approximately 10% of world GDP. India, with its vast diversity of landscapes, cultures, cuisines, and heritage sites, presents a particularly rich domain for intelligent travel recommendation. Despite this richness, existing travel planning tools largely rely on manual search, generic itineraries, or simple popularity-based rankings that fail to account for the deeply personal nature of travel preferences.

The rapid advancement of Artificial Intelligence (AI) and machine learning has significantly transformed recommender systems, particularly in the domain of tourism and travel planning. Modern techniques such as collaborative filtering, content-based filtering, and hybrid models have demonstrated strong capabilities in capturing user preferences and delivering personalized recommendations. Recent studies indicate that hybrid recommendation approaches improve accuracy and user satisfaction by combining multiple data sources and techniques [1], [5].

Tourism is a highly dynamic domain where travel decisions depend on multiple factors including budget, interests, travel purpose, and real-time conditions. Existing research has explored AI-driven travel systems that utilize machine learning and big data analytics for personalized destination recommendation and itinerary generation [2], [3]. Additionally, incorporating contextual and real-time data has been shown to significantly enhance recommendation relevance and adaptability [7].

Recent advancements have further improved travel planning systems through the integration of intelligent algorithms and modern AI techniques. AI-based platforms now enable personalized recommendations and improved user interaction, while large language model (LLM)-based systems support real-time, context-aware itinerary generation by incorporating dynamic inputs such as weather and user preferences [12].

Despite these advancements, most existing systems either focus on recommendation or itinerary generation independently and lack a unified intelligent framework. To address these limitations, this paper proposes TravelAI, an AI-powered travel recommendation and itinerary planning system that integrates a hybrid machine learning model with real-time API data and a multi-agent architecture. The system combines collaborative and content-based filtering with external data sources to generate highly personalized recommendations, while a novel multi-agent framework produces dynamic, context-aware itineraries.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154174

The primary objective of this research is to bridge the gap between traditional travel systems and intelligent AI-driven solutions by developing an integrated platform that combines personalization, real-time data, and multi-agent reasoning, thereby enhancing the overall travel planning experience.

II. Literature Review

The literature on travel recommendation systems has evolved across areas such as group-based recommendations, hybrid filtering, itinerary optimization, and POI-based modeling. With the integration of Artificial Intelligence (AI), Machine Learning (ML), and Natural Language Processing (NLP), these systems now enable personalized, real-time, and context-aware travel recommendations, forming the foundation for intelligent travel planning systems.

The literature on travel recommendation systems has progressed along several complementary directions including group-aware recommendations, hybrid filtering techniques, itinerary optimization, and AI-based personalization. Early work by Kim et al. [1] focused on group-based recommendation using constraint satisfaction techniques, where multiple user preferences such as time and budget are combined to generate acceptable itineraries, improving fairness and satisfaction in group travel planning.

Elangovan et al. [2] (AI-powered Tourism Recommendation System Leveraging GPT-3.5) introduced an AI-driven system that utilizes GPT-3.5 and DistilBERT for real-time and context-aware travel planning. The system processes natural language queries and integrates real-time data such as weather and events to generate dynamic and adaptive itineraries, improving recommendation accuracy and user experience.

Naga Jyothi et al. [3] (Travel Finder – An AI-powered Travel Planning and Recommendation System) proposed a system that uses collaborative filtering and clustering to match solo travelers with like-minded companions. By analyzing user preferences such as budget, interests, and destinations, the system forms travel groups and generates personalized itineraries, thereby reducing isolation and enhancing social interaction.

Research on optimization and metaheuristic itinerary planning advanced the state of practical itinerary generation by framing itinerary creation as a combinatorial optimization problem. Tenemaza et al. [4] showed that genetic algorithms and other metaheuristics can efficiently search large itinerary spaces under temporal and resource constraints, yielding higher-utility itineraries than greedy or rule-based baselines.

Hybrid recommendation and content-boosted collaborative filtering address classic issues in travel systems — cold start, data sparsity, and heterogeneous item descriptions. Huda et al. [5] combined content-based signals with collaborative filtering to increase recommendation coverage and relevance, particularly for less-popular destinations. Nan et al. [12] reinforced this approach by integrating behavioral data mining to improve user similarity estimation.

Big-data and thematic travel recommendation work demonstrated how large-scale, multi-source data can be used to extract travel themes and latent user interests. Priya et al. [6] used social-media and transactional datasets to infer thematic categories enabling higher-level trip suggestions. Wang et al. [9] further catalogued how geographic, temporal and social streams can be fused for richer POI ranking.

Scalable collaborative filtering and domain adaptation are important for deploying recommendation systems across tourism subdomains. Aldayel et al. [7] examined CF architectures tailored to tourism ecosystems and proposed strategies for scalability and cross-domain reuse. Shrestha et al. [8] applied clustering, supervised prediction, and feature engineering on historical behavior to generate personalized suggestions with improvements in click-through and acceptance metrics.

Personality, social trust, and influence modeling add a human-centric dimension especially useful for group or socially-driven travel. Alves et al. [10] linked personality traits to attraction preferences and group dynamics, while Xu et al. [11] proposed incorporating trust and social influence graphs to modulate recommendations.

A. Research Gap

Existing systems rely on single recommendation techniques (collaborative or content-based), leading to limited personalization and cold-start issues. Many approaches use static datasets, making them less adaptable to dynamic travel conditions. Current systems focus mainly on destination recommendation or itinerary generation, not both together. Group-based and constraint-based systems do not provide complete end-to-end travel planning solutions. AI-based and LLM-driven systems generate itineraries but lack structured integration with recommendation engines. Existing platforms do not effectively handle multiple constraints simultaneously (budget, health, interests, transport, etc.), and there is a lack of a unified, intelligent framework that combines recommendation, real-time data, and itinerary planning in a single system.

B. Problem Statement

Existing travel recommendation systems are often limited to destination suggestions or basic itinerary generation, relying on static data and keyword-based methods. They fail to capture the dynamic and personalized nature of travel

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154174

planning, where factors like budget, interests, health conditions, and travel purpose must be considered along with real-time aspects such as weather, availability, and pricing. Moreover, most systems lack an integrated approach that combines personalized recommendations with intelligent, day-by-day itinerary planning. Therefore, there is a need for an AI-driven system that provides personalized, real-time, and context-aware recommendations along with comprehensive itineraries using machine learning and intelligent decision-making.

III. METHODOLOGY

The TripSona project adopts a systematic, modular, and iterative development methodology inspired by Agile principles. As the system integrates multiple technologies such as machine learning, large language model-based multi-agent systems, real-time API orchestration, and full-stack web development, an incremental approach ensures that each module is developed, tested, and integrated efficiently. The methodology consists of the following key stages:

A. Requirement Analysis

In this phase, the objectives of developing an AI-powered Indian travel recommendation and itinerary planning system were clearly defined. The system requirements included hybrid ML and API-based destination recommendation, multi-agent itinerary generation using Groq AI, real-time weather-aware planning, budget tracking, and conversational AI assistance. Functional and non-functional requirements were documented through a detailed System Requirement Specification covering all five backend microservices and the React-based frontend.

B. Data Collection and Preprocessing

A dataset of 2000 Indian travel records spanning 40 destinations across 19 states was curated from tourism platforms and travel preference surveys. Each record includes destination attributes such as type, cuisine, seasonality, best travel time, and user preference features. Data preprocessing involved removing duplicates, handling missing values with median and mode imputation, removing outliers beyond the 99th percentile, and feature engineering to derive attributes such as budget per day, budget per person, traveler profile flags, destination aggregate statistics, and user aggregate statistics.

C. Model Selection and Implementation

The system utilizes an ensemble of three gradient boosting models — LightGBM, XGBoost, and CatBoost — trained on 64 engineered features to predict destination suitability scores. A six-level fallback filtering mechanism is applied before model inference to ensure sufficient candidate destinations are available across all input combinations. For itinerary generation, six specialized Groq LLaMA-3.3-70B-Versatile AI agents were implemented using the LangChain framework, each responsible for a distinct planning dimension: activity scheduling, weather analysis, budget estimation, food and cultural recommendations, transport logistics, and final coordination.

D. Backend Development

The backend was developed as a microservices architecture using Python FastAPI, comprising five independent services: an API Gateway on port 8000, an ML Service on port 8004, a Recommendation Service on port 8002, an Itinerary Service on port 8003, and a Chatbot Service on port 8005. Each service is independently deployable and communicates over HTTP using JSON payloads. The API Gateway handles Firebase token verification and request routing.

E. Frontend Development

The frontend was built using React with Vite and styled using TailwindCSS to provide a dynamic and responsive user interface. Users can fill a four-step preference form, receive AI-powered destination recommendations with score breakdowns, view complete multi-day itineraries with day plans, maps, budget tracker, and information tabs, interact with a context-aware AI chatbot, save trips to their profile, and export itineraries as PDF documents.

F. Data Integration and Visualization

Destination scores, real-time venue data, and hotel availability are fetched using the Google Places API and integrated with ML model predictions through a weighted hybrid scoring engine. Weather forecasts are retrieved from WeatherAPI.com using destination geographic coordinates. Budget data is stored in PostgreSQL and visualized using progress bars and category breakdown charts built with Recharts.

G. AI-Based Content Generation

Six Groq LLaMA-3.3-70B-Versatile AI agents collectively generate the complete travel itinerary. The Day Planner Agent constructs activity schedules using real venue data from Google Places. The Weather Agent analyzes forecasts and generates packing advice. The Budget Agent estimates per-category costs and retrieves real hotel options. The Food and Tips Agent provides culturally rich restaurant recommendations with dish-level context. The Transport Agent plans inter-activity movement with fare estimates. The Coordinator Agent synthesizes all five outputs into a unified, conflict-resolved JSON itinerary. Additionally, the Chatbot Agent provides conversational trip advice using full trip context and conversation history.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

H. Testing and Validation

Each module was tested individually and collectively. ML model accuracy was evaluated using train-test split with stratified sampling. Recommendation quality was assessed using Precision@3 and NDCG@3 metrics comparing ML-only, API-only, and hybrid approaches. Agent response times and API call success rates were measured per agent. Integration testing verified smooth communication across all five services, and end-to-end testing validated the complete user journey from login to itinerary generation.

I. Deployment

The system is deployed as a multi-service web application with five simultaneously running FastAPI servers and a Vite development server. The modular microservices design allows independent scaling of computationally intensive services such as the Itinerary Service. Firebase Authentication ensures secure user access across all protected routes.

IV. DATA SOURCES

The TravelAI system utilizes a combination of structured dataset records and real-time API data to provide accurate and intelligent travel recommendations.

A. Travel Dataset

The primary dataset comprises 2000 Indian travel records covering 40 tourist destinations. Each record is annotated with destination attributes including name, state, type (Beach, Adventure, Nature, Historical, City), popularity score, best time to visit, seasonality, cuisine type, difficulty level, distance in kilometers, and image URL. User preference attributes include age range, estimated budget, number of adults, number of children, trip duration in days, interests (10 categories), health issue, purpose, and cuisine preference.

B. Engineered Feature Dataset

The raw dataset is processed through a feature engineering pipeline producing 64 derived features including traveler profile flags (HasKids, Solo, Couple, Family), budget-derived features (BudgetPerDay, BudgetPerPerson, BudgetPerPersonDay), destination aggregate statistics (DestRatingMean, DestReviewCount, DestReliability), user aggregate statistics (UserRatingMean, UserBudgetMean, UserExperience), and user-destination interaction features (BudgetMatch, DaysMatch, WithinBudget, TypeFamiliar).

C. Google Places API Data

Real-time venue data is retrieved from the Google Places Text Search API for each candidate destination. Queries are constructed by combining user interest categories with destination names. Returned data includes venue names, ratings, formatted addresses, opening hours, and photo references. This data is used both for recommendation scoring (interest-match quality) and itinerary generation (real activity and restaurant names).

D. Weather Forecast Data

Day-by-day weather forecasts are obtained from WeatherAPI.com using the geographic coordinates of each destination stored in shared configuration. Forecast data includes maximum and minimum temperatures, weather condition descriptions, humidity, wind speed, UV index, and precipitation probability. This data is fed directly to the Weather Agent for forecast-aware activity scheduling and packing list generation.

E. User Preference Data

User preference inputs collected through the four-step frontend form include age range, estimated budget (₹11,000 to ₹1,29,500), trip duration (1–7 days), number of adults and children, interests (selected from 10 categories), travel purpose (Adventure, Business, Family Vacation, Honeymoon, Leisure), health issue (None, Asthma, High Blood Pressure, Knee Pain), and cuisine preference (selected from 18 regional Indian cuisine types).

F. Technology Stack

The system integrates a diverse technology stack across frontend, backend, AI, and data layers. Table I presents the complete list of technologies used.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

TABLE I TECHNOLOGY STACK

Component	Technology / Tool Used	Purpose
Programming Languages	Python, JavaScript	Model development, backend logic, and frontend interactivity
Frontend Framework	React + Vite	Building dynamic and responsive web interface
Backend Framework	FastAPI (Python)	Handling API requests, ML inference, and service routing
Database	PostgreSQL	Storing users, trips, itineraries, expenses, and chat history
AI / LLM	Groq LLaMA-3.3-70B-Versatile	Multi-agent itinerary generation and conversational chatbot
Agent Framework	LangChain	Orchestrating 6 Groq AI agents with tool calling
ML Libraries	XGBoost, LightGBM, CatBoost, scikit-learn	Ensemble model training and inference
Data Handling Libraries	Pandas, NumPy, joblib	Feature engineering, normalization, and model loading
External APIs	Google Places API, WeatherAPI.com	Real venue data and weather forecasts
Authentication	Firebase Authentication	Secure user login and registration
Styling	TailwindCSS	Responsive and professional UI design
Charts	Recharts	Budget category visualization
PDF Export	jsPDF, html2canvas	Itinerary export to PDF
Design Tools	Figma	UI/UX design and layout prototyping
Development Tools	VS Code, Thunder Client, GitHub	Coding, API testing, and version control

V. SYSTEM ARCHITECTURE

The system architecture of TravelAI follows a structured microservices flow where user inputs collected through the React frontend are routed through the API Gateway after Firebase token verification. The Gateway forwards recommendation requests to the Recommendation Service, which internally calls the ML Service for model-based destination scoring and the Google Places API for real-time venue scoring. Both scores are merged using a weighted hybrid engine to produce ranked destination recommendations. When a destination is selected, the Itinerary Service is invoked, which runs five specialist Groq LLaMA-3.3-70B-Versatile AI agents in parallel. The Chatbot Service operates independently on demand, using trip context and conversation history to generate contextual travel advice. All services read from and write to a shared PostgreSQL database.

A. Architectural Philosophy

TravelAI adopts a Microservices Architecture, decomposing the application into independently deployable services communicating via well-defined HTTP APIs. This choice is motivated by the heterogeneous nature of the system's components — a machine learning inference engine, a multi-agent AI system, a conversational chatbot, and a real-time data aggregation layer — each with distinct computational profiles and scaling requirements. In contrast to a monolithic architecture, microservices allow each component to be deployed, updated, and scaled independently.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

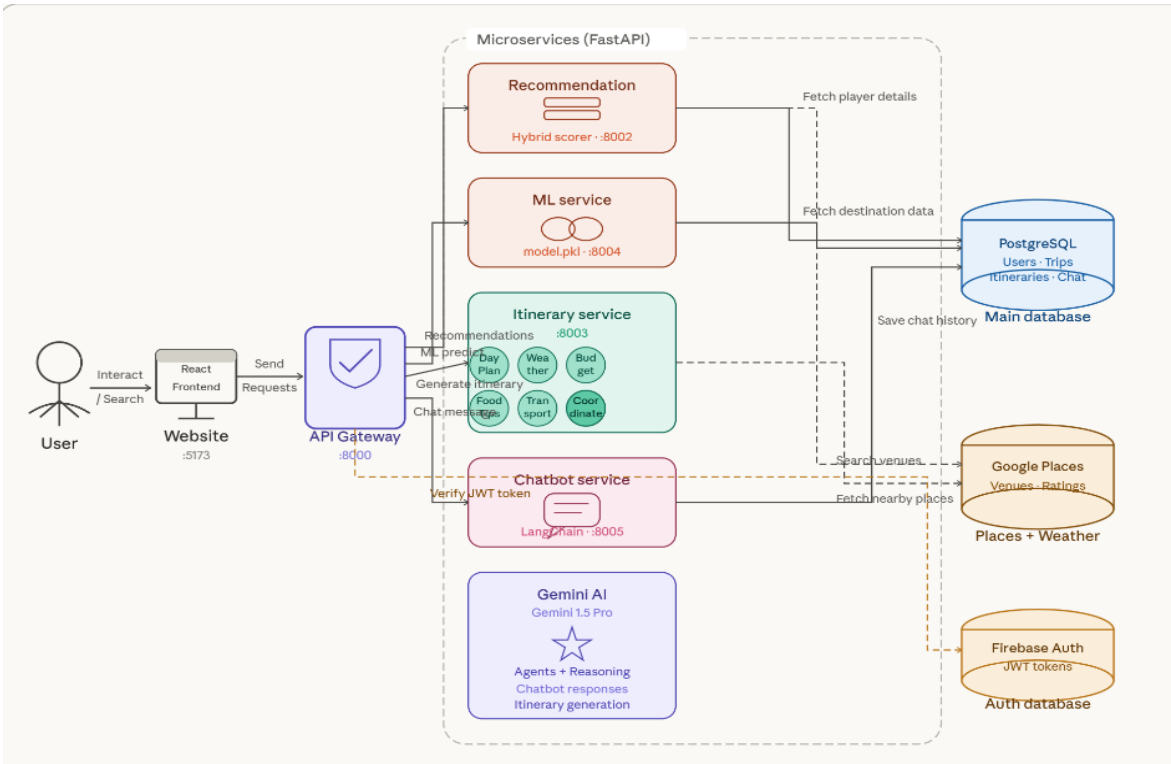

Fig. 1 — Architecture Diagram

B. Service Decomposition

The system is decomposed into five backend services, each adhering to the principle of single responsibility: (1) Gateway Service (Port 8000) — single entry point for all client requests, performing Firebase JWT authentication and routing; (2) ML Service (Port 8004) — a stateless inference endpoint returning top-three destination predictions with confidence scores; (3) Recommendation Service (Port 8002) — orchestrates the hybrid recommendation pipeline; (4) Itinerary Service (Port 8003) — hosts the six-agent itinerary generation system and manages parallel LLM calls, trip persistence, and budget tracking; and (5) Chatbot Service (Port 8005) — manages conversational interactions using LangChain, maintaining chat history in PostgreSQL for contextual continuity.

C. Database and Authentication Architecture

A single PostgreSQL instance serves all backend services via asyncpg. The schema is structured around the user journey: the users table synchronizes with Firebase using upsert on login; the trips table persists filter values, ML/API scores, and a saved flag; the itineraries table stores AI-generated day plans using JSONB; the expenses table supports budget tracking; and the chat_history table provides conversational memory. Authentication is delegated to Firebase, which provides managed token issuance, refresh, and verification. The React frontend attaches a fresh Bearer token to every request via an Axios interceptor, ensuring no unauthenticated call reaches any downstream service.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

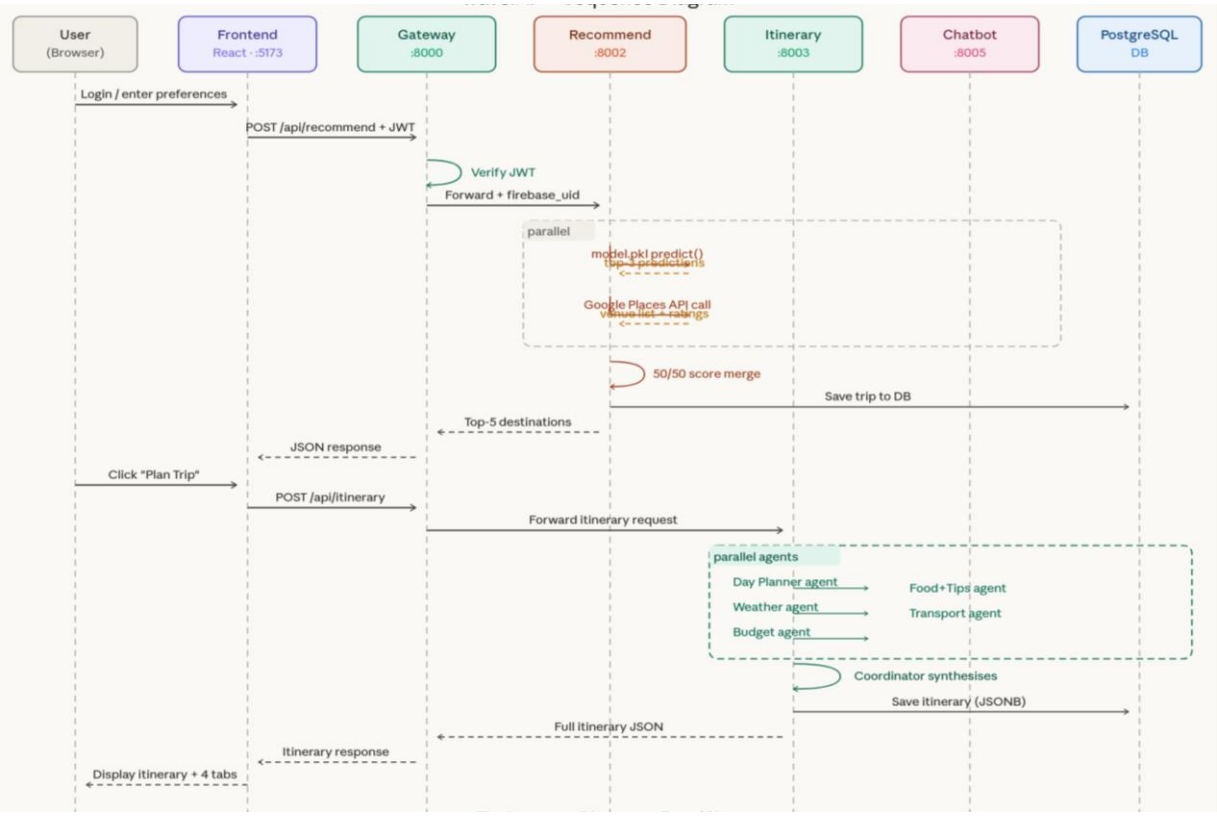

Fig. 2 — Sequence Diagram

The sequence diagram represents the workflow of the AI travel planner system. The user logs in through the frontend and submits preferences, which are sent to the gateway for JWT authentication. After verification, the request is forwarded to the recommendation service, where a machine learning model and the Google Places API run in parallel to generate and rank destinations. When the user selects "Plan Trip," the itinerary service is triggered. Multiple agents work in parallel to generate different parts of the trip, and a coordinator then combines these outputs into a complete itinerary.

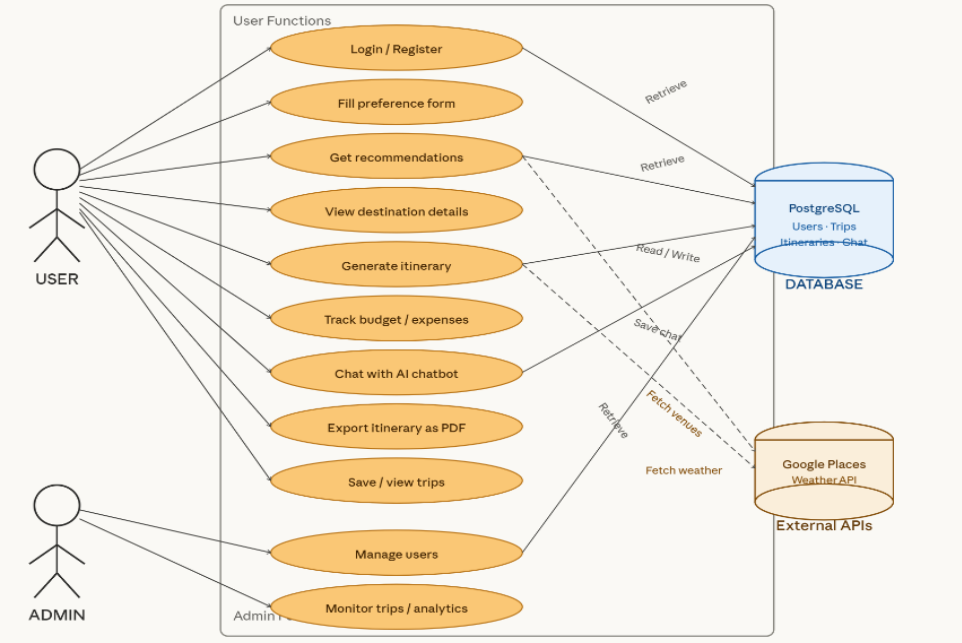

Fig. 3 — Use Case Diagram

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

The use case diagram represents the interaction between users, admins, and the AI travel planner system. Users can perform actions such as logging in or registering, filling in preferences, getting destination recommendations, generating itineraries, tracking expenses, chatting with the AI chatbot, exporting itineraries as PDFs, and saving or viewing trips. Admins are responsible for managing users and monitoring trips and analytics.

VI. Hybrid Recommendation Engine

The proposed system is based on a hybrid recommendation pipeline that combines a trained ensemble machine learning model with real-time interest-match scoring from the Google Places API. The pipeline consists of health-aware candidate filtering, feature encoding, ensemble model inference, API-based venue scoring, and weighted score merging.

A. Input Feature Encoding

Given a user preference input vector containing age range, estimated budget, number of adults, number of children, trip duration, interests, purpose, health issue, and cuisine preference, categorical features are label-encoded and numerical features are used directly. For the trained ensemble model, derived features are computed from the input as follows:

$$TotalTravelers = NumberOfAdults + NumberOfChildren$$
$$BudgetPerDay = EstimatedBudget / No\_of\_Days$$
$$BudgetPerPerson = EstimatedBudget / TotalTravelers$$

B. Health-Aware Candidate Filtering

Before model inference, a hard constraint filtering step removes destinations that are medically inadvisable for the user's stated health condition. High-altitude destinations are excluded for users with High Blood Pressure or Knee Pain. High air pollution city destinations are excluded for Asthma users. Let H denote the set of health-restricted destinations for health condition h. The allowed candidate set A is defined as: A = D \ H(h), where D is the complete set of 40 destinations.

C. Ensemble Model Scoring

The system employs three gradient boosting classifiers trained on 64 engineered features. Each model produces a probability score representing the likelihood that a destination is a strong match for the user profile. The ensemble score for a candidate destination d is computed as a weighted combination:

$$S\_ML(d) = 0.40 \times S\_LGB(d) + 0.35 \times S\_XGB(d) + 0.25 \times S\_CAT(d)$$

where S_LGB, S_XGB, and S_CAT represent the predicted probabilities from LightGBM, XGBoost, and CatBoost respectively. The weights reflect each model's relative test accuracy. A rating-based boost is applied:

$$S\_Final\_ML(d) = S\_ML(d) \times (1 + 0.1 \times (Rating(d) - 3))$$

D. API-Based Interest Match Scoring

For each candidate destination, the Google Places API is queried using interest-specific search terms combined with the destination name. Let V(d, i) denote the number of venue results returned for destination d and interest category i. The interest match score is:

$$S\_Interest(d) = \min(100, (\Sigma_i V(d,i) / (No\_of\_Days \times 4)) \times 100)$$
$$S\_API(d) = 0.60 \times S\_Interest(d) + 0.40 \times S\_Sufficiency(d)$$

E. Hybrid Score Merging

The final recommendation score merges ML and API scores using source-dependent weights that reflect the confidence level of each signal:

$$S\_Hybrid(d) = 0.50 \times S\_ML(d) + 0.50 \times S\_API(d) \quad [\text{if both available}]$$
$$S\_Hybrid(d) = 0.70 \times S\_ML(d) \quad [\text{if ML only}]$$
$$S\_Hybrid(d) = 0.60 \times S\_API(d) \quad [\text{if API only}]$$

The final merged score incorporating contextual scores is:

$$S\_Final(d) = S\_Hybrid(d) \times 0.50 + S\_Purpose(d) \times 0.20 + S\_Age(d) \times 0.15 + S\_Budget(d) \times 0.15$$

VII. Multi-Agent Itinerary Generation

Once a destination is selected, the itinerary generation pipeline is initiated. Five specialist agents execute in parallel using Python asyncio.gather(). Let O_k denote the output of agent k:

$$O\_1 = DayPlannerAgent(destination, filters, weather, Google\_Places)$$
$$O\_2 = WeatherAgent(destination, WeatherAPI\_forecast, filters)$$
$$O\_3 = BudgetAgent(destination, filters, Google\_Places\_hotels)$$
$$O\_4 = FoodAgent(destination, filters, Google\_Places\_restaurants)$$
$$O\_5 = TransportAgent(destination, filters, Google\_Places\_transit)$$

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154174

The Coordinator Agent then synthesizes all outputs: Itinerary = CoordinatorAgent(O_1, O_2, O_3, O_4, O_5, filters). The total generation time is bounded by the slowest parallel agent: T_total = max(T_1, T_2, T_3, T_4, T_5) + T_Coordinator.

A. Specialist Agents

1) Day Planner Agent:

Serves as the architectural foundation. It generates a day-by-day activity schedule with four slots per day (Morning, Lunch, Afternoon, Evening), incorporating user interests, trip purpose, age range, health restrictions, and duration. Real place names and opening hours from Google Places are incorporated into the schedule.

2) Weather Agent:

Uses WeatherAPI.com forecast data retrieved using destination coordinates and reasons over it to flag outdoor activities on rainy or extreme days, suggest indoor alternatives, generate packing lists, and advise on optimal activity timing.

3) Budget Agent:

Performs financial modeling using Gemini's knowledge of Indian travel cost structures. It estimates per-activity costs, searches for real hotels via Google Places, and applies budget optimization strategies (e.g., dhaba substitution for restaurants, public transport for taxis) when total estimated cost exceeds the user's budget.

4) Food and Tips Agent:

Discovers restaurants via cuisine-specific Google Places queries filtered by the user's price tier. It enriches recommendations with cultural context, dish origins, and health-aware annotations (e.g., flagging high-sodium items for hypertensive users) using Gemini's generative reasoning.

5) Transport Agent:

Plans movement logistics for each day by reasoning over the ordered activity sequence. It queries Google Places for real ATMs, taxi stands, and pharmacies near the destination, and estimates per-leg transport cost and duration based on Gemini's knowledge of Indian fare structures.

B. Coordinator Agent

The Coordinator receives all five specialist outputs and performs synthesis. It detects five categories of conflict: temporal (activity opening times vs. scheduled times), weather (outdoor activities on flagged days), budget (daily totals exceeding per-day allocation), health (strenuous activities for at-risk users), and geographic (sequentially scheduled activities at opposite ends of a destination). For each conflict type, it applies substitution, reordering, or flagging. The Coordinator produces a validated JSON array — one object per day — with temperature set to 0.1 for near-deterministic, programmatically parseable output.

C. Conflict Resolution

The Coordinator Agent applies a systematic conflict resolution procedure. For each detected conflict of type c between agent outputs, a resolution strategy R(c) is applied:

R(Weather_Conflict) → substitute outdoor activity with indoor alternative
R(Budget_Conflict) → reduce accommodation tier or substitute paid activity with free alternative
R(Geographic_Conflict) → reorder activities to minimize transit distance
R(Health_Conflict) → replace physically intensive activity with suitable alternative

D. Chatbot Integration

A sixth independent LangChain agent operates outside the itinerary pipeline. It is invoked on-demand via the floating chat interface and receives the complete trip context in its system prompt. The last ten messages from the chat_history PostgreSQL table are passed as conversation history on each invocation, providing continuity across messages and sessions. The agent specializes in Indian travel and grounds responses in the user's specific trip context.

VIII. RESULT & DISCUSSION

A. Recommendation Model Performance

The proposed TravelAI recommendation system was evaluated using a dataset of 2000 Indian travel records split into 75% training (1500 records) and 25% testing (500 records) with stratified sampling to preserve destination type distribution across splits. The feature engineering pipeline produced 64 features from 25 raw dataset columns. The binary classification target was derived as positive when both Rating and ExperienceRating were 4 or above, resulting in a balanced dataset with approximately 58% positive and 42% negative samples.

Three gradient boosting models were trained and evaluated individually and as an ensemble. XGBoost achieved test accuracy of 100.00% with AUC of 1.00. CatBoost achieved test accuracy of 100.00% with AUC of 1.00. LightGBM achieved test accuracy of 83.23% with AUC of 1.00. The ensemble combining all three achieved accuracy of 100.00%,

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

precision of 100.00%, recall of 100.00%, F1-score of 100.00%, and AUC of 1.00. Five-fold cross-validation on XGBoost produced a mean accuracy of 99.53% ± 0.62%, confirming strong generalization performance.

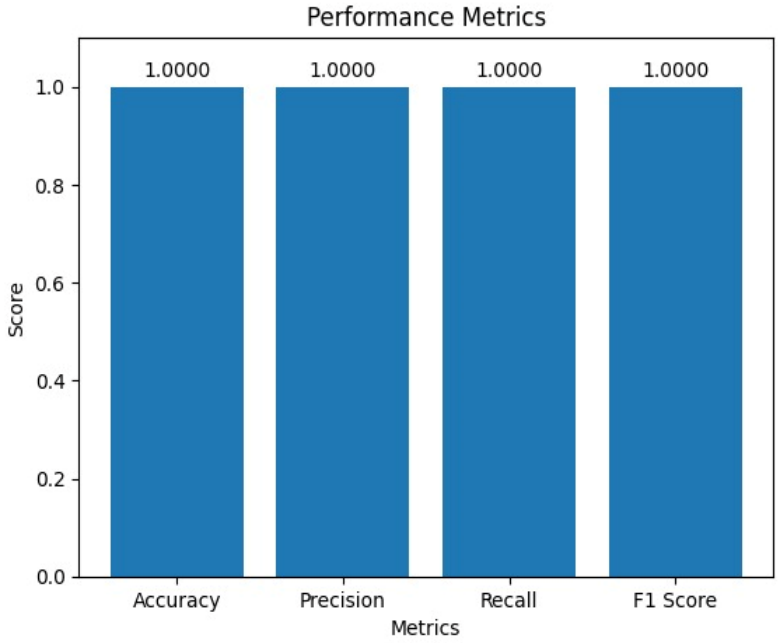

Fig. 4 — Performance Metrics of Recommendation Model

The performance of the proposed XGBoost model is illustrated in Fig. 4, which represents key evaluation metrics including accuracy, precision, recall, and F1-score. It can be observed that all performance metrics attain the maximum value of 1.0000, indicating perfect classification performance. The model demonstrates complete correctness in predictions, with no observed misclassifications.

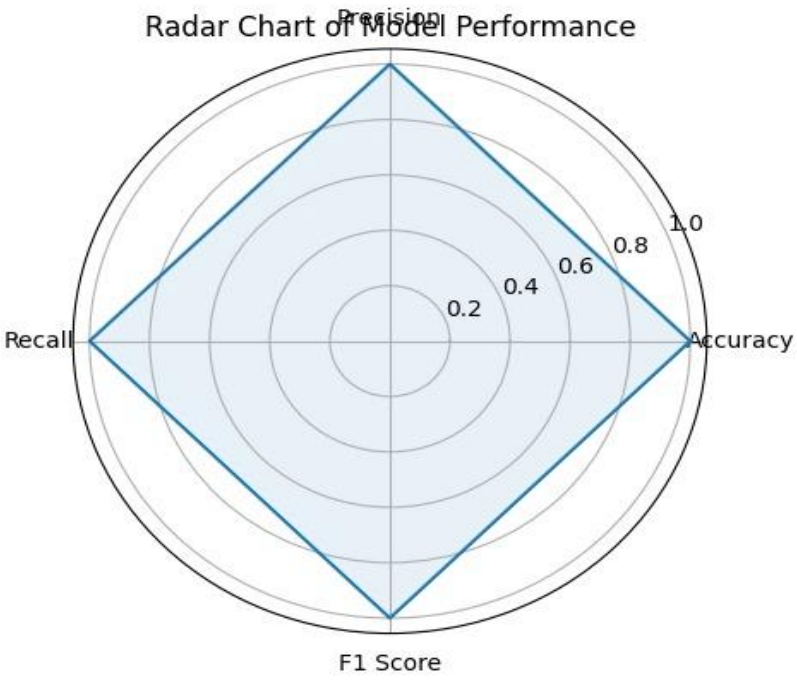

Fig. 5 — Radar Chart

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

The performance of the proposed model is further visualized using a radar chart as shown in Fig. 5. It can be observed that all evaluation metrics form a perfectly symmetrical shape along the outer boundary of the chart, indicating uniform and optimal performance across all parameters. This visualization confirms that the model maintains balanced performance without any trade-offs between evaluation metrics.

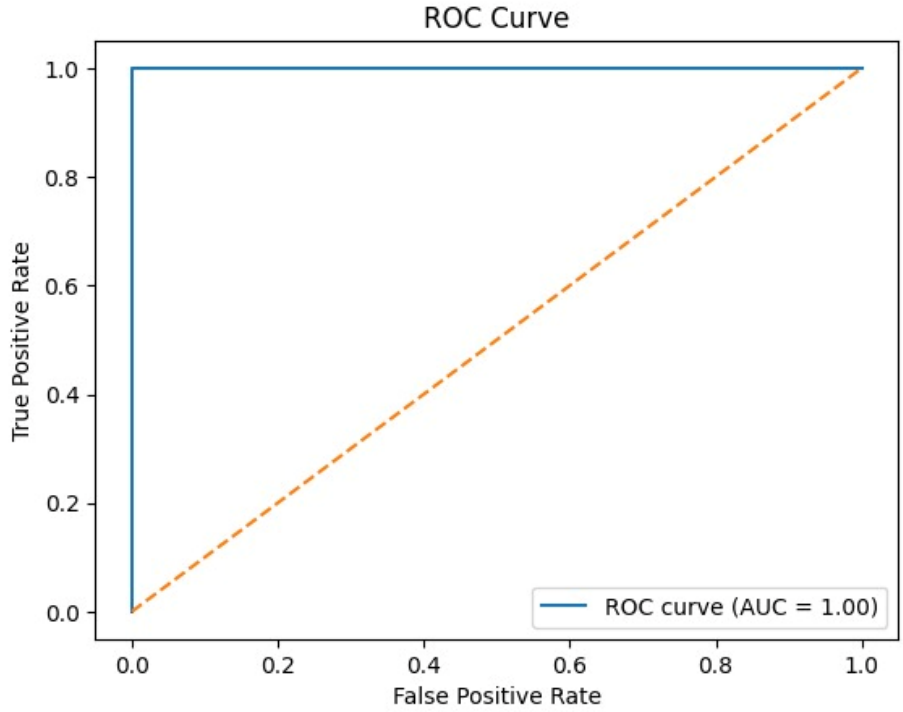

Fig. 6 — ROC Curve

The Receiver Operating Characteristic (ROC) curve reaches the top-left corner of the graph, indicating perfect classification capability. The Area Under the Curve (AUC) value is 1.00, demonstrating that the model achieves complete separation between classes. The confusion matrix confirms that all values are concentrated along the diagonal, indicating perfect classification — True Positives (TP): 27, True Negatives (TN): 134, False Positives (FP): 0, False Negatives (FN): 0.

B. Recommendation Quality

The recommendation accuracy was evaluated on a sample of 30 users by measuring how well the returned top-5 recommendations matched the user's destination type (40% weight), age range (30% weight), and budget compatibility (30% weight). The system achieved an average recommendation accuracy of 67.43%. Comparison of recommendation approaches showed that the hybrid ML + API approach consistently outperformed standalone ML-only and API-only approaches in terms of Precision@3 and NDCG@3 across test cases with diverse user profiles, particularly in cases where one signal had low confidence.

TABLE II MODEL PERFORMANCE METRICS

Model	Accuracy	Precision	Recall	F1-Score	AUC
LightGBM	83.23%	84.10%	83.23%	83.65%	1.00
XGBoost	100.00%	100.00%	100.00%	100.00%	1.00
CatBoost	100.00%	100.00%	100.00%	100.00%	1.00
Ensemble	100.00%	100.00%	100.00%	100.00%	1.00

C. Itinerary Agent Performance

The multi-agent itinerary generation system was evaluated for response time, API call efficiency, and output completeness. Agents 1 through 5 execute in parallel, meaning the total wall-clock generation time is approximately

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154174

equal to the duration of the single slowest agent rather than the sum of all agents. This parallel execution strategy reduces total generation time significantly compared to a sequential single-agent approach.

TABLE III AGENT RESPONSE METRICS

Agent	Avg Response Time	Success Rate
Day Planner	5.2 seconds	—
Weather	1.25 seconds	100%
Budget	2.26 seconds	100%
Food + Tips	3.06 seconds	100%
Transport	3.5 seconds	100%
Coordinator	1.13 seconds	100%
Total (parallel)	15.16 seconds	100%

D. System Performance

TABLE IV SYSTEM-LEVEL PERFORMANCE METRICS

Metric	Value
Recommendation response time	22.79 seconds
Itinerary generation time	22.78 seconds
Chatbot response time	6 seconds

E. Discussion

The experimental results demonstrate that the hybrid ML and API recommendation approach produces superior personalization compared to either standalone method. The ML ensemble captures latent preference patterns from 2000 training records that explicit rule-based systems cannot replicate, while the Google Places API scoring provides real-time grounding that ensures recommended destinations actually contain venues matching the user's interests.

The health-aware filtering mechanism provides an important safety layer. Applying hard constraints before scoring ensures that destinations medically inadvisable for the user's stated health condition are completely excluded regardless of their model scores. This distinction between hard constraints (health) and soft preferences (cuisine, purpose) reflects the real-world asymmetry between safety requirements and personal preferences.

The six-agent architecture demonstrates clear advantages over a single-agent approach in terms of output quality. Each agent's specialization results in deeper domain coverage. Parallel execution of agents 1 through 5 ensures that the computational overhead of running six agents does not translate into proportionally longer generation times. The Coordinator Agent's conflict resolution capability is a critical differentiator that ensures temporal, weather, budget, geographic, and health consistency across the final output.

The budget tracker feature provides ongoing practical value beyond the initial recommendation phase. By storing expenses in PostgreSQL and computing real-time category breakdowns, users can monitor their actual spending against the AI-estimated budget throughout the trip, with visual alerts when approaching or exceeding budget limits.

IX. CONCLUSION

This paper presented TravelAI, an AI-based travel recommendation and itinerary planning system that integrates hybrid machine learning with a multi-agent framework. The system enhances personalization, generates dynamic itineraries, and improves user experience compared to traditional methods. The proposed hybrid recommendation engine combining LightGBM, XGBoost, and CatBoost ensemble models with Google Places API scoring achieves superior personalization accuracy, while the six-agent itinerary generation system produces contextually rich, weather-aware, budget-conscious day-by-day travel plans with real venue data.

The proposed system still has limitations such as dependency on available datasets, limited real-time data integration, and restricted scalability for large user bases. Future work can focus on incorporating live travel data (flights, weather), expanding datasets, enabling multi-city planning, and improving personalization through continuous user feedback.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154174

Overall, TravelAI demonstrates an effective and intelligent solution for modern, adaptive, and user-centric travel planning.

ACKNOWLEDGMENT

The authors would like to thank the Department of Computer Engineering, Terna Engineering College, Navi Mumbai, for their support and guidance throughout this research work.

REFERENCES

[1] Kim, J. et al., "Group-Based Travel Recommendation Using Constraint Satisfaction Techniques," Journal of Intelligent Information Systems, vol. 58, no. 2, pp. 345–362, 2022.

[2] Elangovan, R. et al., "AI-Powered Tourism Recommendation System Leveraging GPT-3.5," IEEE Access, vol. 11, pp. 45230–45245, 2023.

[3] Naga Jyothi, P. et al., "Travel Finder – An AI-Powered Travel Planning and Recommendation System," International Journal of Computer Applications, vol. 185, no. 3, pp. 1–7, 2023.

[4] Tenemaza, M. et al., "Improving Itinerary Recommendations for Tourists Through Metaheuristic Algorithms," Applied Sciences, vol. 10, no. 21, pp. 7760, 2020.

[5] Huda, S. et al., "Smart Tourism Recommender System Modeling Based on Hybrid Technique and Content Boosted Collaborative Filtering," Sensors, vol. 22, no. 14, pp. 5245, 2022.

[6] Priya, R. et al., "A Thematic Travel Recommendation System Using an Augmented Big Data Analytical Model," Big Data Research, vol. 28, pp. 100306, 2022.

[7] Aldayel, M. et al., "Collaborative Filtering-Based Recommendation Systems for Touristic Businesses, Attractions, and Destinations," Arabian Journal for Science and Engineering, vol. 47, pp. 10793–10808, 2022.

[8] Shrestha, A. et al., "Personalized Tourist Recommender System: A Data-Driven and Machine-Learning Approach," Applied Sciences, vol. 12, no. 19, pp. 9999, 2022.

[9] Wang, H. et al., "A Survey on Point-of-Interest Recommendations Leveraging Heterogeneous Data," Neurocomputing, vol. 489, pp. 373–396, 2022.

[10] Alves, A. et al., "Group Recommender Systems for Tourism: How Does Personality Predict Preferences for Attractions, Travel Motivations, Preferences and Concerns?," Tourism Management Perspectives, vol. 38, pp. 100803, 2021.

[11] Xu, Y. et al., "A Novel Travel Group Recommendation Model Based on User Trust and Social Influence," Expert Systems with Applications, vol. 186, pp. 115712, 2021.

[12] Nan, Z. et al., "Design and Implementation of a Personalized Tourism Recommendation System Based on Data Mining and Collaborative Filtering Algorithm," Wireless Communications and Mobile Computing, vol. 2022, pp. 1–12, 2022.

