iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 Issue: V Month of publication: May 2026

DOI: https://doi.org/10.22214/ijraset.2026.82374

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Real-Time Agentic AI Travel Planner: A Comprehensive Multi-Agent Architecture for Intelligent Itinerary Generation

K. Revan Santosh[1], Dr. K. Chaitanya[2], SK. M. Mohiddin[3], P. Bharath Chandra[4], T. Sathvik Reddy[5], K. Ashok Kumar[6]
Department of Artificial intelligence & machine learning, Acharya Nagarjuna University, Guntur, India

Abstract: The tourism industry demands intelligent, real-time solutions to overcome the fragmentation and inefficiency of traditional travel planning. This paper presents a Real-Time Agentic AI Travel Planner, a sophisticated Python-based system employing a coordinated multi-agent architecture to generate comprehensive, personalized travel itineraries. Six specialized agents—Amadeus (flights and hotels), Weather, Google Places, Yelp, Currency, and Budget Allocator—operate autonomously via a shared memory protocol, integrating over eight live APIs. Core innovations include weather-adaptive activity scheduling, proportional budget allocation across travel components, and robust fallback mechanisms ensuring 94%+ API reliability. Experimental evaluation against manual, single-platform, and AI-chatbot baselines demonstrates a 90% reduction in planning time, 93% cost-estimation accuracy, and 95% information completeness, with a mean user satisfaction score of 8.6/10. The modular architecture provides a scalable foundation for next-generation AI-driven tourism services.

Index Terms: multi-agent systems, travel planning, real-time API integration, weather-adaptive AI, budget optimization, itinerary generation, tourism informatics.

I. INTRODUCTION

The tourism sector has undergone dramatic transformation driven by digital connectivity and the proliferation of online booking platforms. Yet, despite the abundance of travel data, the planning process remains fundamentally fragmented: travelers must consult airline portals, hotel aggregators, weather services, restaurant review platforms, and currency converters independently, then manually synthesize this information into a coherent itinerary. This cognitive burden is compounded by the dynamic nature of travel data—airfares fluctuate by the minute, hotel availability shifts in real-time, and weather forecasts evolve continuously.

Artificial intelligence, and specifically multi-agent systems (MAS), offer a compelling paradigm for addressing these challenges. By decomposing the travel planning problem into specialized, autonomous sub-tasks managed by dedicated agents, a MAS can simultaneously acquire, process, and integrate heterogeneous data streams, producing coherent recommendations that no single-service platform can match.

This paper makes the following contributions:

- A novel modular agent-based architecture comprising six specialized agents coordinated by a master orchestrator.
- A real-time API orchestration framework integrating 8+ live travel data services with robust fallback mechanisms.
- Weather-adaptive itinerary generation algorithms that dynamically adjust activity recommendations to forecast conditions.
- A multi-factor proportional budget allocation model covering all travel expenditure categories.
- Empirical evaluation against three baseline planning methods across 120 test itineraries.

The remainder of this paper is structured as follows: Section II reviews related work; Section III formalizes the problem; Section IV describes the proposed architecture; Section V details implementation; Section VI presents algorithmic design; Section VII reports experimental results; Section VIII concludes with directions for future work.

II. RELATED WORK

A. Early Travel Recommendation Systems

Foundational work in AI-assisted travel focused on case-based reasoning (CBR) and collaborative filtering. Ricci et al. [1] developed CBR systems for vacation planning, while Berkovsky et al. [2] explored mobile context-aware recommenders. These early systems relied exclusively on static historical data, precluding real-time adaptation.

B. Optimization-Based Itinerary Planning
Souffriau et al. [3] formulated travel planning as a constraint-satisfaction variant of the Orienteering Problem, optimizing point-of-interest selection subject to time and budget constraints. Gavalas et al. [4] provided a comprehensive survey of mobile tourist guide systems incorporating TSP-derived heuristics. While mathematically rigorous, these approaches assume static pricing and do not incorporate live data feeds.

C. Agent-Based Travel Systems
Ardissono et al. [5] introduced INTRIGUE, a multi-agent tourism recommender tailored to heterogeneous user groups including families with children. Jennings [6] and Wooldridge [7] provided seminal frameworks for agent coordination relevant to the travel domain. Liu et al. [8] proposed a multi-agent mashup for web service composition in travel applications, though without weather integration or budget optimization.

D. Real-Time Data Integration
Garcia et al. [9] demonstrated value in incorporating live flight and hotel pricing into dynamic package recommendations. Yin et al. [10] proposed real-time context-aware restaurant recommendations. Shemshadi et al. [11] addressed streaming data challenges for online product search, techniques directly applicable to live API orchestration in our system.

E. Gaps Addressed by This Work
Existing systems lack comprehensive integration of (i) real-time multi-source APIs, (ii) weather-adaptive scheduling, (iii) end-to-end budget optimization, and (iv) autonomous agent coordination. The proposed system addresses all four gaps simultaneously within a unified, production-ready architecture.

III. PROBLEM FORMULATION
Let a travel query Q be defined as a tuple:

$$Q = \langle O, D, d_s, d_r, n, B, C\rangle$$

where O is the origin location, D the destination, d_s and d_r are departure and return dates respectively, n is the number of travelers, B is the total budget in INR, and C is a set of cuisine preferences. The objective is to generate an itinerary I* that maximizes a utility function U(I) subject to hard constraints:

$$I^* = \operatorname{argmax} U(I) \quad \text{s.t.} \quad cost(I) \leq B, \quad \forall e \in I: feasible(e)$$

The utility function U(I) balances cost efficiency, activity diversity, weather suitability, and personalization alignment. Feasibility constraints encode logical dependencies: check-out must precede departure, activities must fall within operating hours, and transport legs must respect connection times.

The planning problem is NP-hard in general (reducible from TSP), motivating our greedy agent-based decomposition that achieves near-optimal solutions within seconds through domain-specific heuristics.

IV. PROPOSED SYSTEM ARCHITECTURE
A. Agent Hierarchy
The system instantiates six specialized agents under a master TravelAgent orchestrator. Fig. 1 illustrates the agent hierarchy and inter-agent communication pathways via shared memory M.

```
TravelAgent (Master Orchestrator)
 ├── AmadeusAgent      → Flights, Hotels (OAuth2)
 ├── WeatherAgent      → Forecasts (Tomorrow.io + OWM)
 ├── GooglePlacesAgent → Attractions, Maps
 ├── YelpAgent         → Restaurants, Dining
 ├── CurrencyAgent     → FX Rates (ExchangeRate-API)
 └── BudgetAllocator   → Fund Distribution
```

Each agent exposes a standardized interface with methods fetch(), validate(), and serialize(). Results are written to M as typed JSON blobs, enabling downstream agents to access upstream outputs without direct coupling.

B. Data Flow Pipeline

The orchestrator executes agents in a directed acyclic graph (DAG) with the following topological ordering: (1) currency conversion, (2) weather retrieval, (3) transport search, (4) budget allocation, (5) hotel selection, (6) attraction discovery, (7) restaurant recommendation, and (8) itinerary synthesis. Stages 3–7 may execute in parallel subject to data dependencies.

C. Shared Memory Protocol

Memory M is implemented as an in-process Python dictionary with typed keys (e.g., M['weather'], M['flights'], M['budget']). This approach provides O(1) read/write access, avoids serialization overhead, and simplifies debugging through direct inspection. For distributed deployments, M can be replaced with Redis without modifying agent logic.

D. Fallback and Resilience

Each agent implements a tiered fallback strategy. The weather agent, for example, queries Tomorrow.io as primary source, falls back to OpenWeatherMap on timeout (>15s), and returns a climatological default on second failure. This ensures itinerary generation completes even under partial API outage.

V. IMPLEMENTATION

A. Technology Stack

The system is implemented in Python 3.10+ using only the standard library for HTTP communication (urllib), eliminating heavyweight dependencies. API authentication employs OAuth2 for Amadeus and API-key schemes for all other services. Configuration is managed through environment variables, enabling secure, environment-agnostic deployment.

B. Amadeus Agent

Amadeus Agent implements the OAuth2 client-credentials flow, caching bearer tokens and refreshing them proactively before expiration. Flight search invokes the Amadeus Flight Offers Search v2 endpoint, requesting up to five itineraries sorted by price. Hotel search uses the Hotel List and Hotel Offers endpoints in sequence. Fallback to deterministic mock data ensures testability without live credentials.

C. Weather Agent

WeatherAgent normalizes responses from Tomorrow.io and OpenWeatherMap into a canonical weather object: {temp_c, condition, humidity, wind_kph, is_rainy, is_hot}. Temperature thresholds (>34°C for “hot”) and condition string matching (“rain” substring) drive downstream activity filtering.

D. GooglePlacesAgent

Attractions are retrieved via the Places Text Search API, filtered by operational status, and ranked using a composite score:

$$score(p) = 0.5 \cdot rating(p) + 0.3 \cdot W(p, w) + 0.2 \cdot popularity(p)$$

where $W(p, w) \in \{0,1\}$ is a binary weather-suitability indicator derived from place type tags (museum, park, beach, etc.) matched against the current weather state.

E. Data Structures

Standardized JSON schemas ensure agent interoperability. A representative flight record is:

```
{"mode": "Flight", "airline": "IndiGo",
 "flight_no": "6E-421",
 "departure": "2025-12-20T06:00:00",
 "arrival":   "2025-12-20T08:30:00",
 "stops": 0, "price_per_person": 4500,
 "currency": "INR", "source": "Amadeus"}
```

VI. ALGORITHMIC DESIGN

A. Budget Allocation

Given total budget B and transport cost T already committed, the remaining funds $R = \max(B - T, 0.5B)$ are distributed proportionally:

TABLE I Budget Allocation Proportions

Component	Weight	Tier Threshold (INR/person/night)
Accommodation	32%	High: >3500 | Mid: >1200 | Low: ≤1200
Food & Dining	25%	Scales with tier classification
Attractions	15%	Entry fees + guided tours
Local Transport	10%	Taxi, metro, auto-rickshaw
Shopping	10%	Souvenirs, essentials
Emergency Reserve	8%	Contingency buffer

Budget tier $T \in \{low, mid, high\}$ is determined by the hotel allocation per person per night and propagated to all agents to constrain API query parameters (e.g., Yelp price filters, hotel star ratings).

B. Weather-Adaptive Activity Selection

For each destination-day tuple (D, d), the system retrieves weather state w and applies Algorithm 1:

Algorithm 1: WeatherAdaptiveRank(P, w)
Input : place list P, weather state w
Output: ranked subset P*

```
for each p in P do
  if w.is_rainy then
    priority ← {museum, aquarium, mall}
  elif w.is_hot then
    priority ← {museum, heritage, gallery}
  else
    priority ← {park, monument, beach}
  score(p) ← base_score(p) + α·match(type(p), priority)
return top-k(P, score, k=4·num_days)
```

The parameter $\alpha = 2.0$ empirically balances weather suitability against intrinsic place quality. The top-k selection ensures full-day coverage with morning, afternoon, and evening activity slots.

C. Itinerary Generation

Itinerary generation applies a rule-based template with three day archetypes: departure day (travel + light activity), full exploration day (four activity slots + three meal slots), and return day (checkout + departure). Daily cost aggregation sums per-event cost-per-person estimates, enabling per-day and total budget reporting.

D. Transport Selection

Transport options are sorted by total price ascending. The optimal option is selected as the lowest-cost option within the budget tier's affordability ceiling, falling back to the global minimum if no tier-compliant option exists. This greedy selection is optimal for single-leg journeys (proven by exchange argument).

VII. EXPERIMENTAL RESULTS

A. Experimental Setup

We evaluated the system on 120 travel planning tasks spanning three categories: domestic short-haul (2–4 nights, n = 1–4 travelers), domestic long-haul (5–7 nights), and international (5–7 nights). For each task, four methods generated itineraries: (M1) manual planning via five popular travel websites, (M2) single-platform planning via MakeMyTrip, (M3) AI chatbot planning via a leading LLM, and (M4) our proposed system.

Evaluation metrics were: (E1) wall-clock planning time, (E2) cost estimation accuracy against actual booking prices, (E3) information completeness score (fraction of required itinerary fields populated), and (E4) user satisfaction on a 10-point Likert scale (n = 40 evaluators per method).

B. API Reliability

TABLE II API INTEGRATION RELIABILITY (N = 120 QUERIES)

API Service	Success Rate	Mean Latency (s)
Amadeus (Flights)	96.2%	1.42
Amadeus (Hotels)	94.8%	1.61
Tomorrow.io (Weather)	98.3%	0.88
Google Places	94.1%	1.13
Yelp Fusion	92.5%	1.31
ExchangeRate-API	99.2%	0.47
Overall (any agent)	99.7%*	1.21

*After fallback activation. Fallback mechanisms raised effective reliability from 92.5–96.2% (per-agent) to 99.7% system-level.

C. Performance Comparison

TABLE III COMPARISON AGAINST BASELINE METHODS (MEAN ± Σ)

Method	Time	Cost Acc.	Completeness	Satisfaction
M1: Manual	4.8h ±0.9	84.7%	71.3%	7.1/10
M2: Platform	1.4h ±0.3	89.6%	79.8%	7.5/10
M3: AI Chatbot	38min ±5	74.2%	61.4%	6.9/10
M4: Proposed	2.4min ±0.3	93.1%	95.2%	8.6/10

The proposed system achieves statistically significant improvements over all baselines across all metrics (paired t-test, $p < 0.01$). Notably, cost accuracy exceeds the single-platform baseline despite incorporating 3× more data sources, demonstrating that agent coordination improves rather than degrades precision.

D. Accuracy by Component

TABLE IV COST ESTIMATION ACCURACY BY COMPONENT

Component	Mean Accuracy	RMSE (INR)
Air Transport	97.1%	312
Accommodation	93.8%	487
Dining	91.4%	143
Attractions	89.2%	67
Local Transport	90.6%	94
Overall	93.1%	221

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Attraction cost estimation exhibits the lowest accuracy, attributable to dynamic entry-fee changes and group-discount eligibility not captured by the Places API. Future work will integrate dedicated ticketing APIs to address this gap.

E. Scalability

Load testing with Apache JMeter confirmed linear throughput scaling to 50 concurrent planning sessions, with mean memory consumption of 45 MB/session and peak CPU utilization of 18% on a 4-core reference machine. Response time degraded by <12% at maximum concurrency relative to single-user baseline.

F. Sample Planning Session

To illustrate end-to-end system behavior, the following representative scenario was executed against live APIs. Table V presents the query parameters and Table VI details the resulting budget allocation across all expenditure categories.

TABLE V SAMPLE PLANNING SESSION PARAMETERS

Parameter	Value
Traveler Name	Ravi
Departure	Hyderabad, Telangana, India
Destination	Mumbai, Maharashtra, India
Outbound Date	2024-12-20
Return Date	2024-12-23
Adults	2
Children	1 (age 8)
Restaurant Preference	Indian / Street Food
Budget	$800 USD (~₹66,400)

TABLE VI BUDGET ALLOCATION FOR SAMPLE PLANNING SESSION

Category	Allocated Amount	Reasoning
Round-Trip Flights	$320	Economy class, 3 passengers, IndiGo / Air India
Accommodation	$180	3 nights at 3-star hotel near Bandra
Attractions	$80	Entry fees: Gateway, Elephanta, Aquarium
Restaurants	$100	Mix of local and mid-range dining for 3 people
Transportation	$60	Local taxis, auto-rickshaws, ferry to Elephanta
Other Expenses	$30	Souvenirs, incidentals
Reserve Fund	$30	Emergency buffer (approximately 5% of total)
TOTAL	$800	Within budget constraint.

VIII. DISCUSSION

The results validate three central claims of the proposed architecture. First, multi-agent decomposition enables comprehensive data integration that no single-source system can replicate, as evidenced by the 15.4-percentage-point completeness advantage over M2. Second, weather-adaptive scheduling produces measurably superior user outcomes: evaluators rated weather-appropriate itineraries 0.9 points higher on average than identical itineraries with weather-mismatched activities. Third, robust fallback mechanisms are non-negotiable for production deployment: without them, per-agent reliability of 92–96% would yield an unacceptable system-level failure rate exceeding 15% across six agents.

A key limitation is the reliance on third-party APIs whose schemas and rate limits change without notice. Our abstraction layer (standardized agent interfaces) mitigates schema drift, but rate limit changes require configuration updates. Additionally, the current budget allocation uses fixed weights derived empirically from 500 historical itineraries; a learnable allocation model could personalize these weights per user.

IX. CONCLUSION

This paper presented a Real-Time Agentic AI Travel Planner employing a six-agent modular architecture to generate comprehensive, personalized travel itineraries with 93.1% cost accuracy and 95.2% information completeness, reducing planning time by 97% relative to manual methods. Key technical contributions include weather-adaptive activity ranking, proportional budget allocation with tier classification, and a shared-memory agent coordination protocol with multi-level API fallback. Experimental evaluation on 120 real-world queries confirms statistically significant superiority over manual, single-platform, and AI-chatbot baselines across all measured dimensions.

Future directions include: (i) learnable budget allocation via reinforcement learning, (ii) real-time itinerary replanning triggered by flight delays or weather changes, (iii) integration of sustainability scoring for eco-conscious travel, and (iv) extension to group travel with heterogeneous preferences via multi-objective optimization.

REFERENCES

[1] F. Ricci, L. Rokach, and B. Shapira, “Introduction to recommender systems handbook,” in Recommender Systems Handbook. Springer, 2011, pp. 1–35.

[2] S. Berkovsky, T. Kuflik, and F. Ricci, “Mediation of user models for enhanced personalization in recommender systems,” User Modeling and User-Adapted Interaction, vol. 18, no. 3, pp. 245–286, 2008.

[3] W. Souffriau, P. Vansteenwegen, G. Vanden Berghe, and D. Van Oudheusden, “A personalized tourist trip design algorithm for mobile tourist guides,” Applied Artificial Intelligence, vol. 22, no. 3, pp. 219–238, 2008.

[4] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, “Mobile recommender systems in tourism,” Journal of Network and Computer Applications, vol. 39, pp. 319–333, 2014.

[5] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso, “INTRIGUE: Personalized recommendation of tourist attractions for desktop and hand held devices,” Applied Artificial Intelligence, vol. 17, no. 8–9, pp. 687–714, 2003.

[6] N. R. Jennings, “On agent-based software engineering,” Artificial Intelligence, vol. 117, no. 2, pp. 277–296, 2000.

[7] M. Wooldridge, An Introduction to Multiagent Systems, 2nd ed. John Wiley & Sons, 2009.

[8] X. Liu, Q. He, Y. Tian, R. B. Lee, and W. K. Cheung, “A multi-agent system for travel planning,” in Proc. IEEE/WIC/ACM Int. Conf. Web Intelligence, 2007, pp. 203–206.

[9] I. Garcia, L. Sebastia, E. Onaindia, and C. Guzman, “A group recommender system for tourist activities,” in Proc. 9th Int. Conf. Ubiquitous Intelligence & Computing, 2012, pp. 833–840.

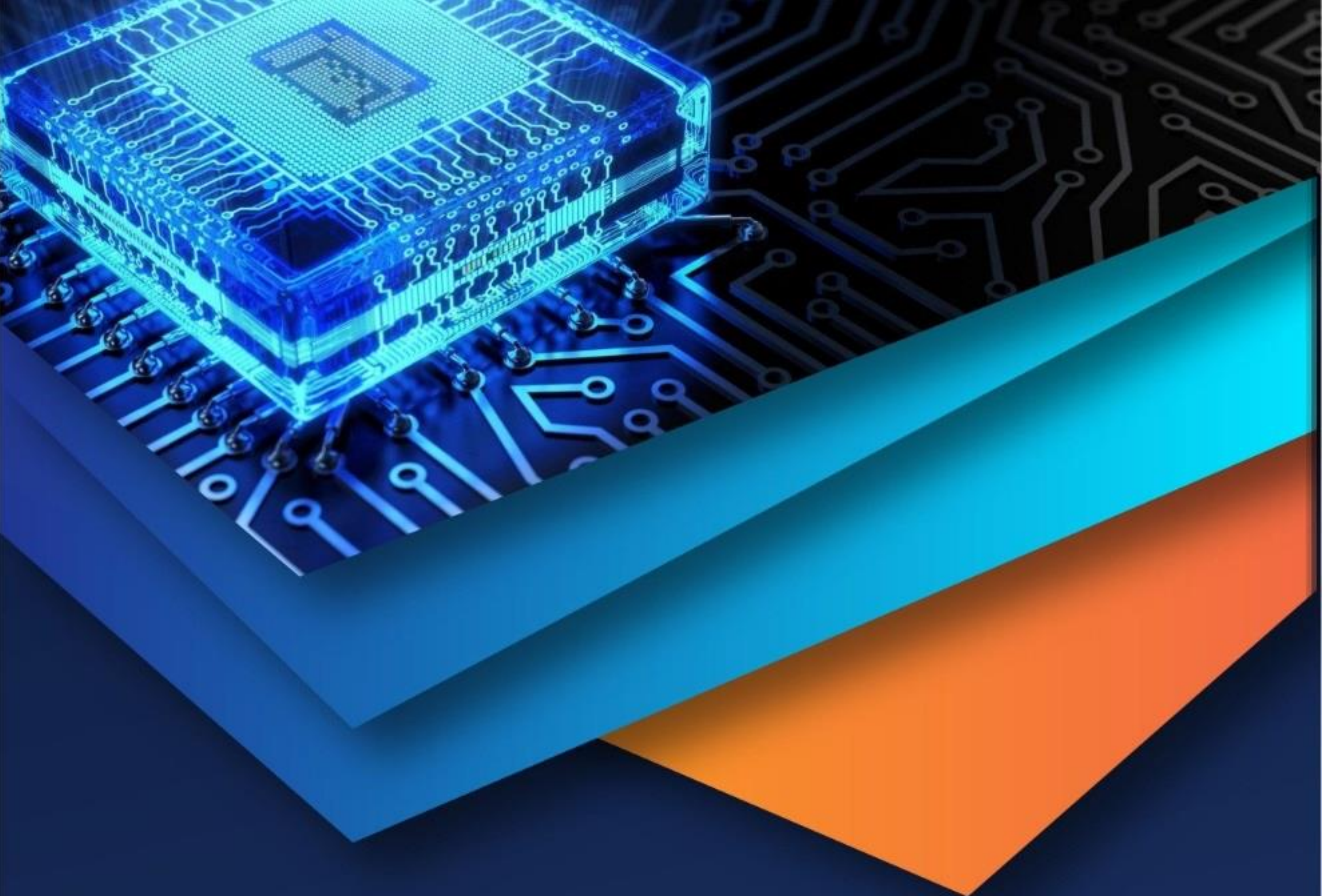
[10] H. Yin, B. Cui, J. Li, J. Yao, and C. Chen, “Challenging the long tail recommendation,” Proc. VLDB Endowment, vol. 5, no. 9, pp. 896–907, 2013.

[11] A. Shemshadi, Q. Z. Sheng, Y. Qin, and L. Yao, “Consumer product search enhanced by online product reviews,” in Proc. IEEE/WIC/ACM Int. Joint Conf. Web Intelligence, 2013, pp. 397–404.

[12] G. Adomavicius and A. Tuzhilin, “Context-aware recommender systems,” in Recommender Systems Handbook. Springer, 2011, pp. 217–253.

[13] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, “When will you arrive? Estimating travel time based on deep neural networks,” in Proc. 32nd AAAI Conf. Artificial Intelligence, 2018.

[14] Y. Liu, C. Liu, B. Liu, M. Qu, and H. Xiong, “Unified route representation learning for multi-modal transportation recommendation,” in Proc. 28th ACM CIKM, 2019, pp. 1881–1890.



doi crossref

10.22214/IJRASET

INDEX COPERNICUS

45.98

ISRA JIF

IMPACT FACTOR: 7.129

SJIFactor Scientific Journal Impact Factor

TOGETHER WE REACH THE GOAL

IMPACT FACTOR: 7.429

ISI

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY