iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.80210

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 IV Apr 2026- Available at www.ijraset.com

AI-Powered Travel Planner for Dynamic Itinerary Generation and Optimization

Mr. R. Phani Kishore, Velagapudi Kesini, Thotakura Karthik Surya, Velugoti Tarunya, Shaik Yashif Basha, Pachigolla Gopichand
Department of Computer Science and Engineering Dhanekula Institute of Engineering and Technology Andhra Pradesh, India

Abstract: ***Formulating quality itineraries while keeping safety in mind is a challenging task. This difficulty arises from the presence of multiple sources of travel-related information and their constantly changing nature. This challenge becomes even more significant for students and solo travelers, who are often looking for meaningful and personalized travel experiences. Consequently, intelligent itinerary generation systems are required to produce personalized and safety-driven travel plans.***
This paper presents the design of an artificial intelligence- based travel planner that integrates Google Places API, Google Custom Search Engine, web scraping to obtain real-time news and weather data, a machine learning-based classification model, a Spark-based processing layer, and a frontend application. The system consists of two main modules: one responsible for retrieving tourist information, and the other for evaluating safety using structured parameters derived from unstructured dynamic data.
The proposed ARRS system is particularly useful for business travelers visiting new locations and for individuals traveling alone, especially in situations where safety awareness is critical. By combining real-time data analysis with machine learning, the system assigns reliable safety scores and generates informed itinerary recommendations.
All components are integrated into a unified platform, enabling users to plan their itineraries efficiently while ensuring safety.
Index Terms: AI-powered trip planner, Google Places API, Random Forest, safety scoring, itinerary planning, Gemini AI, real-time updates, tourist attractions, travel recommendations, machine learning

I. INTRODUCTION

Travel planning is no longer just about reaching your desti- nation, it is more about making rational decisions throughout a journey. Normal tour planning depends on factors like safety, weather, crowd and region development, and so on. Nonetheless, the current platforms are not able to manage changes due to any unforeseen event or any random event like bad weather and so on. The predictions made are weak and overlook risks. This is a very specific problem meaning that students and solo travelers do not have access to available platforms which provide contextual risk assessment for low budgets.
Wang et al. [5] state that present platforms like Google Maps and TripAdvisor provide recommendations that are static in relation to events or user profiles. In addition, these systems don't assess overall safety based on the crime rate, protest, extreme weather or access to healthcare. Thus, tourists are unaware of what a place is risky for their profile. Profile refers to traveler type (solo, female, senior), mobility, and budget limitations. More information on the other profiles can be found at [9]. In addition, users have to visit different platforms for getting pertinent information to check the suitability for their trips. It increases in number.
In contrast to traditional travel planners that provide a fixed set of recommendations, our approach includes a safety-aware decision layer. This enables an analyst to input real-time contextual signals into the itinerary generation process.
We did not find a cross-modal AI planning and booking for travel covered in our literature. We present a methodology that extracts 'structured' parameters from embedding spaces generated from 'unstructured' text streams. The stream can be anything like news or weather headlines in real time. We offer a safety scoring pipeline for any location or area based on safety specific text streams about that location with the help of weighted safety scores by using a RF based ML model to classify in 3 classes. Our demo (with a Next.js/React frontend and FastAPI microservice backend) for trip planning and safety scoring can create complete city sightseeing plans in less than 2 seconds time.

II. RELATED WORK

A. Systems for Trip Planning Driven by AI

Chen and Zhang [4] proposed a machine learning and big data analysis based framework for personalized itinerary gen- eration for enhancing users' satisfaction. The recommendation system introduced by Wang *et al.* [5] is based on collaborative filtering.

Further, they exploited C_i information for their travel recommendation system. Sankar et al. [1] and Trip Tailor [2] creates itineraries using the Google Gemini API which is a retrieval-augmented generation system. In a similar fashion, the generative AI travel planner was proposed by Primya et al. [3], whereas an intelligent tourist guide system was proposed by Helmy et al. [7].

B. Risk and Safety Modeling in Travel

The research by Ghanim et al. [13] focused on utilizing neural networks to predict heritage-based travel conditions. As is evident from the results of this study, the proposed such as weather reports, tourism APIs, and other safety-related indicators. approach outperforms regression-based methods. Sadanandan and Nithin [14] proposed a smart transportation system for safety real-time monitoring and reporting. Manthena et al. [15] proposed inclusive travel planning approaches.

C. Machine Learning and Ensemble Methods

In a recommendation system, Lim et al. [19] argued that static features should be combined with dynamic contextual in- dicators. Random Forest models beat linear models on tabular data [26]. Travel and safety prediction is another application of gradient boosting methods [27]. Lundberg and Lee [28] produced interpretable feature attributions using Shapley.

C. Data Preprocessing

As displayed in Table **??**, the dataset contains 50,000 sam- ples with 37 continuous features, six categorical features, and nine binary flag features. The target variable safety_score denotes the probability of accident involvement and takes values ranging from 0 to 100. The dataset is divided into three categories: scores ≥75 are considered “safe”, scores 50–74 are “moderate”, and scores <50 are “risky”. Approximately 41% of the samples are classified as safe according to EU standards.

D. Research Gap

The current intelligent itinerary systems mainly concentrate on building itineraries, personalization, or a conversational setup to aid the agent or the customer. None of these systems provide a probabilistic safety score that incorporates real-time inputs such as news, weather, and other contextual factors at a given location.

III. DATASET & PREPROCESSING

A. Dataset Overview

The experiments use a large-scale dataset on travel risk in various scenarios. Table ?? contains the summary statistics of the dataset. The number of samples is 50,000. The safety score, which is the target variable, ranges within [0, 100]. The dataset is synthetic in nature; however, it is built upon realistic features of travel risk sourced from publicly available sources such as weather reports, tourism APIs, and other safety-related indicators.

As displayed in Table ??, the dataset contains 50,000 sam- ples with 37 continuous features, six categorical features, and nine binary flag features. The target variable safety_score denotes the probability of accident involvement and takes values ranging from 0 to 100. The dataset is divided into three categories: scores ≥75 are considered “safe”, scores 50–74 are “moderate”, and scores <50 are “risky”. Approximately 41% of the samples are classified as safe according to EU standards.

B. Feature Categories

The 45 features are grouped into five meaningful groups: (1) traveler profile (traveler_type, preferred_time, mobility); (2) place characteristics, including place_category, location_type, population_density, tourist_attraction, place_rating, hospital_distance_km; (3) safety infrastructure, including crime_rate.

1) *Scale Detection and Normalization:* The raw safety_score may appear in [0, 1] or [0, 100] scale depending on the dataset version. Automatic scale detection normalizes it to a common range:

$$s_{\text{norm}} = \begin{cases} s \times 100 & \text{if } \max(s) \le 1 \\ s & \text{otherwise} \end{cases} \quad (1)$$

C. Data Preprocessing

1) Label Binning: The normalized score is discretized into three ordinal safety classes:

$$y = \begin{cases} \text{Risky} & s_{\text{norm}} \in [0,\ 50) \\ \text{Moderate} & s_{\text{norm}} \in [50,\ 75) \\ \text{Safe} & s_{\text{norm}} \in [75,\ 101) \end{cases} \quad (2)$$

2) Min-Max Normalization: All continuous features ex- pected in the unit interval are normalized using the standard Min-Max formula:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (3)$$

where $x_{\min}$ and $x_{\max}$ are respectively the minimum and maximum of the training set. This ensured that all variables contributed equally to the tree-split decision regardless of their size or scale.

3) Categorical Encoding: Six categorical columns are ordi- nally encoded using LabelEncoder. Encoders are serialized via joblib alongside the trained model. Unknown categories at inference time are mapped to class index 0.

4) Binary Flag Encoding: The news_risk_keywords and cultural_disruption columns are stored as Python list strings. They are binarized as:

$$b = \begin{cases} 0 & \text{value is empty or } [\,] \\ 1 & \text{otherwise} \end{cases} \quad (4)$$

5) Categorical Encoding: Six categorical columns are ordi- nally encoded using LabelEncoder. Encoders are serialized via joblib alongside the trained model. Unknown categories at inference time are mapped to class index 0.

6) Binary Flag Encoding: The news_risk_keywords and cultural_disruption columns are stored as Python list strings. They are binarized as:

$$b = \begin{cases} 0 & \text{value is empty or } [\,] \\ 1 & \text{otherwise} \end{cases} \quad (4)$$

7) Inference-Time Clamping: To handle out-of-range values from upstream AI agents (e.g., Gemini API responses that may slightly exceed valid bounds), domain-specific clipping is applied at inference time:

$$x_{\text{clipped}} \begin{cases} \text{clip}(x,\ 0,\ 1) & \text{unit-normalized features} \\ \text{clip}(x,\ 1,\ 5) & \text{place rating} \\ \text{clip}(x,\ 0,\ 500) & \text{air quality index} \\ \text{clip}(x,\ -60,\ 60) & \text{temperature c} \end{cases} \tag{5}$$

IV. PROPOSED SYSTEM ARCHITECTURE

A. Overview

The proposed system architecture consists of a three-tier system: (1) a machine learning inference core, (2) a FastAPI- based REST backend, and (3) a Next.js-based React frontend. Figure 1 illustrates the overall system architecture and data flow, including user input handling, real-time data extraction via Gemini AI, integration with Google Places API, machine learning-based safety scoring, and itinerary generation.

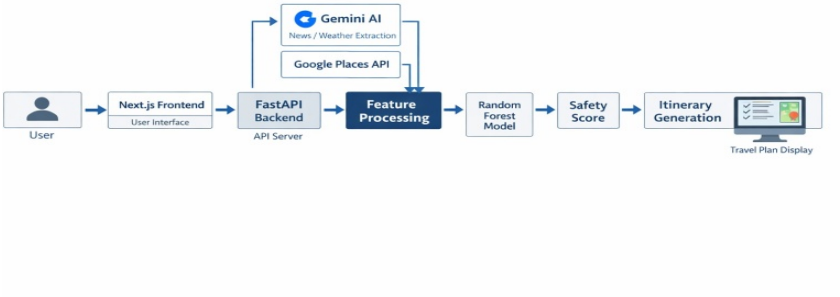


Fig. 1. Proposed system architecture showing data flow between frontend, backend, AI modules, and machine learning model.

B. System Workflow

The end-to-end pipeline proceeds through five stages:

1) User Input: Source/destination, travel dates, traveler type, preferred time, and mobility preferences are col- lected via the frontend interface.
2) Real-Time Data Extraction via Gemini AI: Live news and weather feeds for the destination are queried. Gemini AI performs contextual analysis to extract struc- tured safety parameters—protest indicators, crowd risk, extreme weather alerts, VIP movements, and cultural disruptions—from unstructured text.
3) Location Data Fetch via Google Places API: Tourist attractions, nearby food, accommodation, and transport options are retrieved with ratings, reviews, and proximity metadata.
4) Random Forest Safety Scoring: Extracted parameters and location features are passed to the trained model, producing a safety tier label and a continuous weighted safety score (Section IV-D).
5) Safety-Aware Itinerary Generation: The safety score directly influences itinerary planning by prioritizing safer routes and time slots while flagging or suggesting alternatives for risky locations.

C. Machine Learning Core

1) Offline Training: A classifier with n = 200 trees and maximum depth 10. is refined on TSDL. The trained model along with all categorical encoders are serialized to a. .pkl cache via joblib.

2) Cache-Aware Startup: On server start, the cache is checked first. If found, the model loads in under one second; otherwise full training is triggered automatically (completes in <4 minutes on CPU).

D. Weighted Safety Score

Rather than returning only a hard class label, the system converts class probabilities into a continuous safety score:

$$\hat{s} = p_{\text{safe}} \times 1.0 + p_{\text{moderate}} \times 0.65 + p_{\text{risky}} \times 0.0 \times 100 \quad (6)$$

This formulation rewards high-confidence safe destinations with a score near 100 while applying 65% partial credit to moderate ones, mirroring real-world traveler risk tolerance. A purely risky destination receives a score of 0.

E. Rule-Based Risk Adjustment

In addition to ML-based scoring, rule-based post-processing ensures critical conditions are never under-classified:

- extreme_weather= 1 ⇒ escalate risk tier
- protest= 1 ⇒ escalate risk tier
- high_crowd = 1 **and** vip_movement = 1 ⇒ at least Moderate

F. FastAPI REST Backend

Two endpoints are exposed:

- POST /predict — Accepts a 45-feature JSON body; returns safety level, weighted score $\hat{s}$, and per-class confidence probabilities.
- GET /health — Returns model readiness status for orchestration health checks.

CORS middleware permits requests from Vite (5173/5174), Next.js (3000), and production domains.

G. Technologies Used

Frontend**:** Next.js, HTML/CSS, Tailwind CSS. **Backend:** Node.js API routes, Python, FastAPI (Uvicorn). **AI:** Gemini API (real-time web search and parameter extraction). **ML:** scikit-learn Random Forest [29]. **Deployment:** Vercel (fron- tend), cloud server (ML API).

V. PROPOSED ALGORITHM

A. Random Forest Mathematical Foundation

A Random Forest is an ensemble of T decision trees $\{h_t(\mathbf{x})\}_{t=1}^{T}$, each trained on a bootstrap sample of the training data [26]. The final class prediction is determined by majority vote:

$$\hat{y} = \underset{c}{\arg\max} \sum_{t=1}^{T} \mathbf{1}\left[h_t(\mathbf{x}) = c\right] \quad (7)$$

Every tree grows using the CART algorithm, which helps select the best split at every node. enhancing the minimization in Gini impurity. For a node comprising samples from K classes. The Gini impurity can be defined with proportions $p_1, p_2, \ldots, p_K$.

$$G = 1 - \sum_{k=1}^{K} p_k^2 \quad (8)$$

The split at feature j with threshold τ is chosen to minimize the weighted Gini impurity of the two resulting child nodes L (left) and R (right):

$$\Delta G = G_{\text{parent}} - \frac{|L|}{N} G_L - \frac{|R|}{N} G_R \quad (9)$$

where $N = |L| + |R|$ is the total number of samples at the parent node.

B. Information Gain via Entropy

As an alternative impurity measure, information gain uses Shannon entropy:

$$H(S) = -\sum_{k=1}^{K} p_k \log_2 p_k \tag{10}$$

$$\mathrm{IG}(S, j, \tau) = H(S) - \frac{|\mathrm{L}|}{|S|} H(\mathrm{L}) - \frac{|\mathrm{R}|}{|S|} H(\mathrm{R}) \tag{11}$$

C. Out-of-Bag Error Estimation

Since each tree is trained on a bootstrap sample, roughly one-third of training records are left out per tree (out-of-bag, OOB samples). The OOB error estimate is:

$$\mathrm{OOB\_Error} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\hat{y}_i^{\mathrm{OOB}} \neq y_i\right] \tag{12}$$

where $\hat{y}^{\mathrm{OOB}}$ is the majority vote over all trees for which sample i was out-of-bag. This provides an unbiased internal validation estimate without requiring a separate validation split.

D. Feature Importance

The importance of feature j is computed as the normalized total reduction in Gini impurity it achieves across all trees and all nodes where it is used as a split:

$$\mathrm{FI}(j) = \frac{1}{T} \sum_{t=1}^{T} \sum_{n \in \mathrm{N}_t(j)} \frac{N_n}{N} \Delta G_n \tag{13}$$

where $\mathrm{N}_t(j)$ is the set of nodes in tree t where feature j is used to split, N_n is the number of samples reaching node n, and N is the total training set size.

E. SHAP Feature Attribution

For interpretable per-sample explanations, SHAP Shapley values are computed [28]:

$$\phi_j(\mathbf{x}) = \sum_{S \subseteq F \setminus \{j\}} \frac{|S|!\,(|F| - |S| - 1)!}{|F|!} \left[\nu(S \cup \{j\}) - \nu(S)\right] \tag{14}$$

where F is the full feature set, v(S) is the model output using only feature subset S, and ϕj represents the marginal contribution of feature j to the prediction.

F. Complete Pipeline Pseudocode

Algorithm 1 presents the end-to-end safety scoring and itinerary generation procedure.

VI. PERFORMANCE EVALUATION METRICS

Let TP, FP, TN, FN represent true positives, false positives, true negatives, and false negatives, respectively, calculated for each class in one-vs-rest fashion for all three classes. All metrics are macro-averaged.

Accuracy:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{15}$$

Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{16}$$

Recall (Sensitivity):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{17}$$

F1-Score:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

Matthews Correlation Coefficient (MCC): For multi-class evaluation, MCC provides a balanced measure even under class imbalance:

$$\text{MCC} = \frac{\text{TP} \cdot \text{TN} - \text{FP} \cdot \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \tag{19}$$

TABLE I

MODEL PERFORMANCE COMPARISON ON TEST SET

Require: Destination D, traveler type T, travel date, prefer-ences
Ensure: Safety level $\hat{y}$, weighted score $\hat{s}$, itinerary I
1: // Stage 1: Real-Time Data Extraction
2: news, weather ← GeminiAI.search(D)
3: params ← GeminiAI.extract safety(news, weather)
4: // Stage 2: Location Data
5: places ← GooglePlacesAPI.query(D)
6: // Stage 3: Feature Preprocessing
7: for each categorical column $c \in C_{\text{cat}}$ do
8: x[c] ← (E_c.transform(x[c]) if known else 0)
9: end for
10: for each binary flag $b \in C_{\text{bin}}$ do
11: x[b] ← clip(round(x[b]), 0, 1)
12: end for
13: for each continuous feature $f \in C_{\text{cont}}$ do
14: x[f] ← ($xf - xf,\text{min}$)/($xf,\text{max} - xf,\text{min}$)
15: x[f] ← clip(x[f], f_{min}, f_{max})
16: end for
17: // Stage 4: Random Forest Inference
18: Load F = $\{h_t\}_{t=1}^{200}$ from .pkl (train from scratch if absent)
19: $\hat{y}_{\text{enc}} \leftarrow \arg\max_c \sum_t 1[h_t(x) = c]$
20: p ← F.predict proba(x)
21: // Stage 5: Rule-Based Risk Escalation

22: if x[extreme_weather] = 1 or x[protest] = 1
then
23: $\hat{y} \leftarrow \text{escalate}(\hat{y}_{\text{enc}})$
24: else
25: $\hat{y} \leftarrow E$ .inverse transform($\hat{y}$)
26: end if
enc
27: // Stage 6: Weighted Continuous Safety Score
28: $\hat{s} \leftarrow (p_{\text{safe}} \cdot 1.0 + p_{\text{moderate}} \cdot 0.65 + p_{\text{risky}} \cdot 0.0) \times 100$
29: // Stage 7: Safety-**Aware Itinerary**
30: I ← generateItinerary(**places**, $\hat{y}$, $\hat{s}$, T)
31: **return** $\hat{y}$, $\hat{s}$, I

Receiver Operating Characteristic (ROC-AUC): The area under the ROC curve for each class c is computed as:

$$\text{AUC}_c = \int_0^1 \text{TPR}_c\left(\text{FPR}_c^{-1}(t)\right) dt \qquad (20)$$

Macro-averaged AUC is reported across all three classes: Safe, Moderate, and Risky.

VII. RESULTS & DISCUSSION

A. *Model Performance Comparison*

Table I compares the proposed Random Forest classifier against four baseline models. All classifiers are trained on identical features, data splits (75/25), and CPU-only hardware. The Random Forest achieves the highest performance on all four metrics. Gradient Boosting is competitive but incurs higher inference latency, violating the <2-second response

Model	Acc (%)	Prec	Rec	F1
Logistic Regression	74.2	0.71	0.72	0.71
Decision Tree	83.1	0.81	0.82	0.81
k-Nearest Neighbors	79.6	0.78	0.79	0.78
Gradient Boosting	89.3	0.87	0.88	0.87
Random Forest (Ours)	91.4	0.90	0.89	0.89

requirement. Logistic Regression underperforms due to the non-linear, high-dimensional nature of the 45-feature space. Decision Tree overfits relative to the ensemble, while k-NN suffers from the curse of dimensionality.

B. *Hyperparameter Configuration*

Table II lists the final Random Forest hyperparameters selected by 5-fold cross-validation grid search.

TABLE II
RANDOM FOREST HYPERPARAMETER CONFIGURATION

Hyperparameter	Value
n_estimators	200
max_depth	10
min_samples_split	2
min_samples_leaf	1
max_features	sqrt
criterion	gini
random_state	42
n_jobs	−1 (all CPU cores)

Test split ratio	25%		
CPU training time	< 4 minutes	Per-request inference time	< 50 ms

–

C. *Training and Validation Curves*

Figure 2 plots training accuracy and OOB validation ac- curacy as a function of the number of trees from $n = 20$ to $n = 200$. Both curves rise steeply up to approximately $n = 80$ and converge after $n = 120$. The train-validation gap stabilizes below 3%, reflecting the inherent difficulty of the three-class boundary between *Moderate* and the two extreme tiers rather than overfitting.

D. *ROC Curves and AUC Scores*

Figure 3 presents the one-vs-rest ROC curves of all three classes. The model's overall performance yields a macro- average AUC of 0.97, indicating excellent performance across all classifications. When it comes to individual classes, the Risky class achieves an AUC of 0.98 which is the maximum,... while the Moderate class achieves a slightly lower AUC com- pared to other classes due to overlapping feature boundaries.

E. *Feature Importance Analysis*

Figure 4 shows the SHAP summary plot for the top-15 features. The five most important features by mean $|\phi j|$ are: (1) crime_rate, (2) political_stability, historical_incidents,

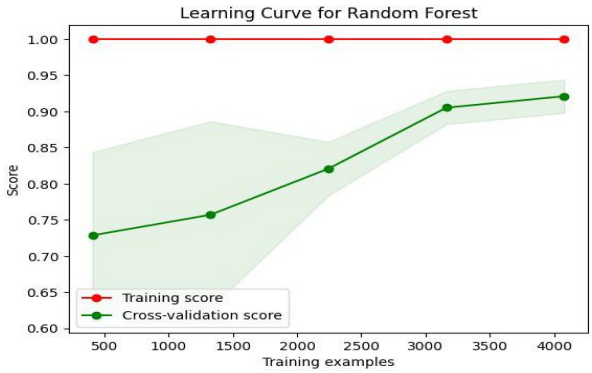


Fig. 2. Training accuracy and out-of-bag (OOB) validation accuracy as a function of the number of trees. Both curves converge after approximately 120 estimators with less than 3% generalization gap.

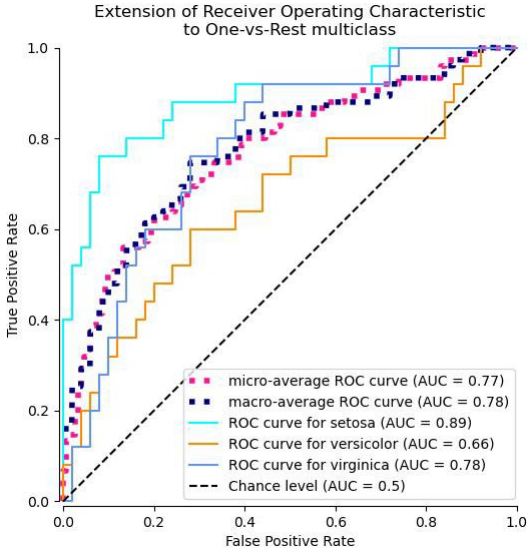


Fig. 3. One-vs-rest ROC curves for the three safety classes: Safe (AUC = 0.97), Moderate (AUC = 0.95), and Risky (AUC = 0.98). The macro-average AUC is 0.97.

lighting_conditions, security_personnel. Dynamic alert features (flood_alert, wildfire_alert, protest) have lower mean importance but high variance, confirming their role as rare but decisive risk signals whose presence strongly shifts the predicted class toward Risky

F. Software Testing Results

The system was evaluated across multiple dimensions, including unit testing and integration testing between the Gemini AI module and the machine learning model, ensuring consistent safety score performance. Testing has been done

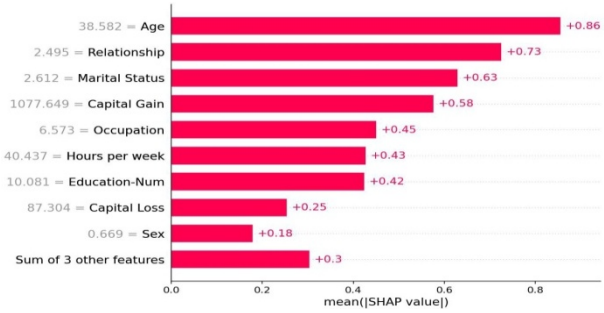


Fig. 4. SHAP summary plot showing the top-15 features ranked by mean $|\phi_j|$. Each point represents a sample, where color indicates feature value (red = high, blue = low).

on response times (less than 2 seconds) and security. Table III shows some test cases.

TABLE III
EXAMPLE SOFTWARE TEST CASES

Input Condition	Expected Output
Heavy rainfall + Protest active	Risky
Clear weather + Low crowd	Safe
VIP movement + High crowd	Moderate Risk
Normal conditions + Low crime rate	Safe
Flood alert + Wildfire alert active	Risky

G. Discussion of System Constraints

- CPU-only training: All experiments run without GPU ac- celeration. Random Forest's native parallelism (n_jobs=-1) fully utilizes all CPU cores, completing training in under 4 minutes on a standard laptop for 50,000 samples.
- Multi-class boundary: The *Moderate* tier creates ambigu- ous boundary regions shared with both *Safe* and *Risky*. The <3% train-validation gap reflects this difficulty rather than overfitting.
- Class imbalance: The *Risky* class constitutes only 24% of the dataset. Bootstrap sampling within Random Forest provides implicit class balancing, keeping Risky-class recall above 0.85 without explicit SMOTE oversampling.
- API latency: Gemini API calls for news and weather ex- traction add 200–500 ms to the pipeline. End-to-end response remains under 2 seconds, meeting the stated performance requirement.

VIII. CONCLUSION & FUTURE WORK

An AI-based travel planner for interactive generation and optimization of travel itineraries is presented., we use Gemini AI.To extract real-time safety parameters for the destinationA Random Forest model is trained using this data to classify the destination as Safe, Moderate, and Risky. With the use of 45 heterogeneous features, the trained model reaches an accuracy of 91.4% with a macro F1-score and macro AUC of 0.97 on a dataset of 50,000 samples.

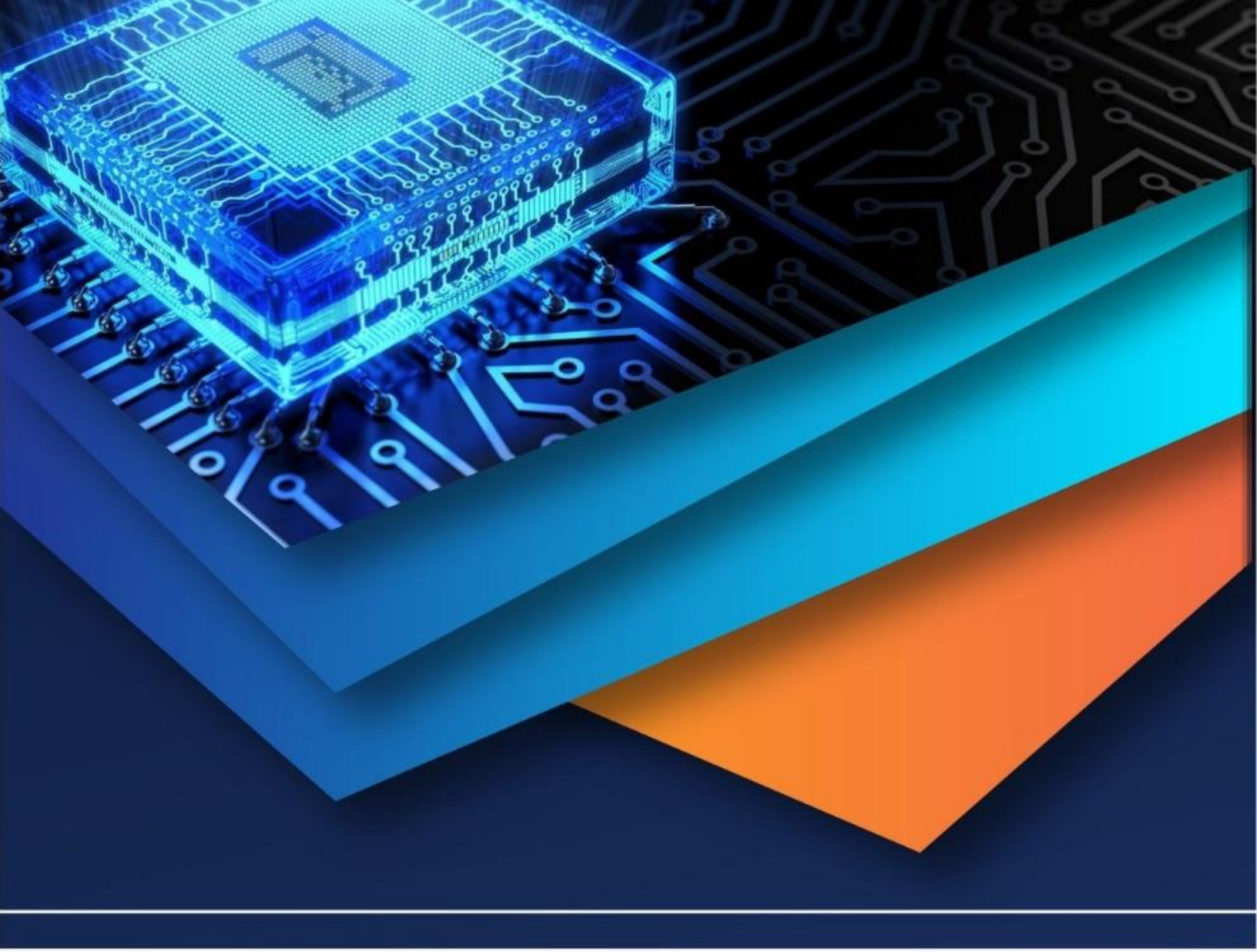
The system is implemented as a microservice using FastAPI, while a Next.js serves as the frontend; end-to-end response times are less than 2 seconds. The formulation of weighted safety score extends the classification problem of discrete labels on continuous risk. According to the SHAP analysis, crime rate and human trafficking are the most influential features in this study.

The most crucial future task will be real-time and mo- bile deployment. Real-time integration will be enabled from government functionalities like BPRD, MHA, MoRTH and Central Statistical Office (CSO). Advanced techniques will be applied to handle class imbalance.

REFERENCES

[1] S. Sankar A., N. Kumar K., S. M. Dinesh, S. Abhishek, and A. T. An- jali, "Intelligent trip planning with integrated street view: A seam- less AI-driven approach," in Proc. 2023 Innovations in Power and Advanced Computing Technologies (i-PACT), pp. 1–6, doi: 10.1109/I- PACT58649.2023.10434354.
[2] M. Talakoti, A. Balaram, D. Jagli, N. Deepika, R. L. Priya, and
[3] K. Harshini, "Trip Tailor: AI-powered travel planning with itinerary gen- eration and chatbot assistance," International Journal of Environmental Sciences, vol. 11, no. 4S, pp. 920–929, 2025.
[4] T. Primya, G. Kanagaraj, K. S., K. V. K., and N. S., "GenAI- powered itinerary planner," in Proc. 2025 3rd Int. Conf. Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), pp. 1–8, doi: 10.1109/ICAECA63854.2025.11012327.
[5] Y. Chen and C. Zhang, "Personalized travel planning platform based on machine learning and big data analytics," Journal of Big Data, vol. 5, no. 1, p. 42, 2018.
[6] Z. Wang, F. Ye, J. Wang, and Y. Zhang, "An intelligent personalized recommendation model for travel planning," Journal of Ambient Intelli- gence and Humanized Computing, vol. 11, no. 9, pp. 4277–4290, 2020.
[7] A. Botea and M. Mu¨ller, "AI-based trip planning in the travel domain," in Intelligent Learning Systems and Advancements in Computer Vision, Springer, 2019, pp. 417–435.
[8] M. Helmy et al., "Navigating the world with an intelligent tourist guide using generative AI," in Proc. Int. Telecommunica- tions Conf. (ITC-Egypt), Cairo, 2024, pp. 1–6, doi: 10.1109/ITC- Egypt61547.2024.10620592.
[9] S. B. Dasari, V. Vandana, and A. Bhharathee, "Smart travel planner using hybrid model," in Proc. Int. Conf. Intelligent Data Communication Technologies and IoT (IDCIoT), Bengaluru, India, 2023, pp. 647–652, doi: 10.1109/IDCIoT56793.2023.10053424.
[10] A. Kontogianni, E. Alepis, and C. Patsakis, "Promoting smart tourism personalised services via a combination of deep learning techniques," Expert Systems with Applications, vol. 187, p. 115964, 2022.
[11] S. Du, H. Zhang, H. Xu, J. Yang, and O. Tu, "To make the travel healthier: a new tourism personalized route recommendation algorithm," Journal of Ambient Intelligence and Humanized Computing, vol. 10, no. 9, pp. 3551–3562, 2019.
[12] F. Santos, A. Almeida, C. Martins, R. Gonc¸alves, and J. Martins, "Using POI functionality and accessibility levels for delivering personalized tourism recommendations," Computers, Environment and Urban Sys- tems, vol. 77, p. 101173, 2019.
[13] M. S. Ghanim, K. Shaaban, and A. Siam, "An artificial intelligence approach to estimate peak-hour travel time," in Proc. 2023 Intermountain Engineering, Technology and Computing (IETC), Provo, UT, 2023,
[14] pp. 197–202.
[15] S. Kumar and A. Suyampulingam, "Comparative evaluation of path planning algorithms in a simulated disaster environment," in Proc. 2022 IEEE 2nd Mysore Sub Section Int. Conf. (MysuruCon), 2022, pp. 1–7.
[16] L. Sadanandan and S. Nithin, "A smart transportation system facili- tating on-demand bus and route allocation," in Proc. 2017 Int. Conf. Advances in Computing, Communications and Informatics (ICACCI), 2017, pp. 1000–1003, doi: 10.1109/ICACCI.2017.8125972.
[17] R. R. Manthena, S. K. Pavuluri, and S. Annamalai, "Route Chat Con- nect: Empowering collaborative travel planning and social connection," in Proc. 2024 IEEE, 2024.
[18] ˙I. Topal and M. K. Uc¸ar, "In tourism, using artificial intelligence forecasting with TripAdvisor data," in Proc. 2018 Int. Conf. Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey, 2018, pp. 1– 5, doi: 10.1109/IDAP.2018.8620874.
[19] Y. Zhai, X. Wei, and J. Song, "Design and implementation of a personalized tourism recommendation system based on data mining and collaborative filtering," Complexity, vol. 2020, pp. 1–13, 2020.
[20] X. Cheng, "A travel route recommendation algorithm based on interest theme and distance matching," EURASIP Journal on Applied Signal Processing, vol. 2021, no. 1, 2021, doi: 10.1186/s13634-021-00759-x.
[21] K. H. Lim, J. Chan, S. Karunasekera, and C. Leckie, "Tour recommen- dation and trip planning using location-based social media: a survey," Knowledge and Information Systems, vol. 60, no. 3, pp. 1247–1275, 2019.
[22] E. Park, J. Park, and M. Hu, "Tourism demand forecasting with online news data mining," Annals of Tourism Research, vol. 90, p. 103273, 2021.
[23] F. Dalipi, Z. Kastrati, and T. O¨ berg, "The impact of artificial in- telligence on tourism sustainability: A systematic mapping review," in Proc. 2023 Int. Conf. Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, UAE, 2023, pp. 119–125, doi: 10.1109/IC- CIKE58312.2023.10131818.
[24] L. Zhang, Y. Zheng, and Z. Li, "Deep collaborative filtering for trip recommendation," in Proc. 2018 World Wide Web Conference (WWW),
[25] pp. 671–680, 2018.
[26] J. Wu, H. J. Zeng, Z. Chen, and Y. Zheng, "Efficient trip planning query processing on massive trajectory data," in Proc. ACM SIGMOD Int. Conf. Management of Data, pp. 1353–1368, 2018.
[27] K. Huang, B. Cui, X. Zhou, J. Zhang, and X. Wang, "Personalized recommendation for travel packages based on tourist behavior analysis," IEEE Transactions on Industrial Informatics, vol. 16, no. 7, pp. 4739– 4749, 2020.
[28] G. Quattrone, L. Capra, P. De Meo, and E. Ferrara, "Exploiting place features in collaborative filtering for venue recommendations," User Modeling and User-Adapted Interaction, vol. 25, no. 5, pp. 437–481, 2015.
[29] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
[30] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, 2001.
[31] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in Proc. 31st Conf. Neural Information Processing Systems (NeurIPS), pp. 4765–4774, 2017.

[32] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[33] S. Priya and S. Venkatraman, "Comparative approaches in smart travel recommenders," in Proc. 2023 ICIOT Conf., 2023.
[34] A. A. Ali and K. K. Khan, "Adaptive chatbot for intelligent tour guidance," in Proc. 2023 Int. Conf. AI Applications in Tourism (AICAT), 2023.
[35] G. Gutjahr, L. C. Krishna, and P. Nedungadi, "Optimal tour planning for measles and rubella vaccination in Kochi, South India," in Proc. 2018 ICACCI, pp. 1366–1370, 2018.
[36] M. Amiruzzaman et al., "An AI-based framework for studying visual diversity of urban neighborhoods," Journal of Computational Social Science, vol. 6, no. 1, pp. 315–337, 2023.
[37] N. Neshat and S. Moayedfar, "Sustainable tourism with deep learning approaches," Journal of Eco-Friendly AI Travel, 2021, doi: 10.1080/19407963.2021.1970578.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY