**AI-Based Travel Planner: Personalized Itinerary**

**Recommendations by Age, Budget**

***Abstract*—** ***Planning a personalized trip can be time-consuming and overwhelming due to the wide range of available options and varying traveler needs. This paper presents an AI-based travel planner that simplifies itinerary generation by considering the user's age, travel budget, and trip duration. The system uses a rule-based AI approach to categorize users into profiles and provides tailored daily recommendations for accommodations, tourist attractions, and meal options. Developed using Python Flask and a lightweight front-end, the system delivers fast and user-friendly output without requiring complex machine learning models or external APIs. The planner is especially helpful for first-time or indecisive travelers, offering them cost-effective, relevant, and personalized travel plans. Experimental results demonstrate the system's ability to cater to different age groups and budgets effectively, making travel planning more accessible and efficient.***

***Keywords— AI Travel Planner, Personalized Itinerary, Budget-Based Recommendation, Rule-Based System, Flask Application, Smart Tourism, Age-Based Filtering, Daily Travel Suggestions.***

# Introduction

Planning a travel itinerary that fits a traveler’s personal interests, age, budget, and available time is often a complex and time-consuming process. Most people today rely on a combination of websites, blogs, travel apps, and word-of-mouth recommendations to decide where to go, where to stay, what to eat, and which attractions to visit. This can be overwhelming, especially for inexperienced travelers or those with limited time to plan. Additionally, the abundance of unfiltered information often results in decision fatigue, confusion, and dissatisfaction with the final travel experience. The lack of personalization in traditional travel planning platforms makes it harder for travelers to make confident, cost-effective decisions aligned with their needs.

To address this challenge, we propose an **AI-Based Travel Planner** that offers smart, real-time, and personalized travel itineraries based on three simple user inputs: **age**, **total budget**, and **number of travel days**. The system uses a rule-based AI model to classify users into appropriate categories based on age group (e.g., youth, adult, senior) and budget level (e.g., low, medium, high). It then filters relevant data to suggest suitable **hotels**, **day and night attractions**, and **meals** for each day of the trip. The planner aims to reduce the stress of decision-making while ensuring that travelers receive meaningful and contextually appropriate recommendations tailored to their profile.

The system is implemented using **Python Flask** as the backend framework and a user-friendly **HTML/CSS interface** for front-end interaction. The datasets used for destination, food, and hotel suggestions are structured and stored in a categorized format using in-code dictionaries or JSON structures. This allows the recommendation engine to operate quickly and without requiring high computational resources or large-scale databases. The rule-based engine ensures transparency, scalability, and simplicity in delivering results. For each day of the trip, the system provides five hotel options, two daytime and nighttime attractions, and two suggestions for each meal—breakfast, lunch, and dinner—ensuring a complete itinerary is presented to the user instantly.

In summary, this project aims to simplify the travel planning process by using lightweight AI logic and intelligent filtering techniques to generate complete, **age-specific** and **budget-friendly** travel plans. It is particularly useful for solo travelers, families, and senior citizens who need reliable, fast, and accessible planning tools. Unlike machine learning models that require large datasets and training time, this planner delivers immediate results using a predefined knowledge base and condition-based logic. Future enhancements may include the integration of real-time tourism APIs, user feedback loops, and adaptive learning mechanisms to continuously improve the relevance and diversity of recommendations. The proposed system contributes to the growing field of **smart tourism** by combining personalization, automation, and efficiency in a single platform.

# Related Work

Travel recommendation systems have evolved significantly in recent years, leveraging advancements in artificial intelligence (AI), machine learning (ML), and user profiling techniques. One widely studied approach involves the use of **machine learning algorithms** to analyze user behavior and generate travel suggestions. For example, Goyal et al. (2021) proposed a machine learning-based system that customizes travel plans by analyzing user preferences, previous choices, and ratings. Although such systems show promise in delivering personalized experiences, they often depend heavily on large datasets and prior user history, making them less effective for first-time users or cold-start situations.

Another direction of research focuses on **IoT and cloud-based travel assistance**, particularly in enhancing user safety and experience during transit. Singh and Verma (2020) introduced a real-time health monitoring system using IoT sensors and cloud communication to alert medical services during emergencies in travel. While their work is more healthcare-focused, it highlights the potential of integrating real-time data and contextual awareness into travel-related systems. However, their solution does not focus on the leisure or planning aspects of tourism, nor does it consider factors like user age or budget for itinerary planning.

Several studies have also explored the potential of **rule-based expert systems** in recommendation engines. Rule-based systems, unlike ML models, do not require extensive data for training and can operate effectively on domain knowledge and structured inputs. These systems are often used when decision rules are well-defined and human-understandable. In the context of tourism, rule-based engines can quickly map user attributes (like age and budget) to predefined categories and generate recommendations accordingly. Their main advantage lies in simplicity, transparency, and faster output generation, which aligns well with the goals of the proposed AI travel planner.

Further, researchers such as Gupta et al. (2021) reviewed the use of **AI in tourism**, focusing on how intelligent systems can predict destinations, seasonal preferences, and traveler moods using emotion-aware computing. Although these studies showcase advanced personalization capabilities, they often require access to sensitive user data, emotional profiling, or real-time social media analysis. Such systems also face privacy

and ethical challenges, making them less feasible for simple, day-to-day travel planning for the general public.

In contrast to the systems above, the AI-based travel planner presented in this paper takes a **minimal input–maximum output approach**, making it more accessible to users who prefer quick, lightweight solutions without login, training, or extensive personalization history. It fills the gap in the literature by focusing specifically on **age-based and budget-specific filtering** for itinerary creation using a rule-driven approach. By combining the strengths of personalization and simplicity, the system addresses the limitations of existing ML-heavy or static recommendation platforms, offering an effective middle ground that is easy to deploy and scale for real-world users.

## Existing Work

## Several existing travel recommendation platforms, such as Google Travel, TripAdvisor, and Booking.com, provide users with suggestions for destinations, accommodations, and activities. While these systems use general filters, ratings, and sometimes user location or preferences, they lack deep personalization based on key factors like age, total travel budget, and trip duration. Some advanced systems employ machine learning techniques to offer tailored recommendations based on user history and behavioral patterns; however, they face limitations such as the cold-start problem and high computational requirements. Rule-based expert systems have also been used in travel planning, offering faster and more interpretable outputs but often lacking adaptive capabilities. Context-aware systems have emerged in recent studies, integrating factors such as user mood, season, and weather conditions; yet, they typically rely on continuous online connectivity, user profiling, and third-party APIs. Despite these developments, there remains a significant gap for a simple, real-time, and lightweight AI solution that can deliver personalized, age- and budget-specific travel plans with minimal user input—precisely what the proposed AI Travel Planner addresses.

## Proposed Work

The proposed system is a lightweight, AI-driven travel planner that generates fully personalized daily itineraries based on three user inputs: age, total travel budget, and trip duration. Unlike machine learning models that require large datasets and training, this system adopts a rule-based logic to instantly map user profiles to relevant travel categories. The backend, developed in Python using the Flask framework, categorizes users into specific age and budget groups (e.g., youth-low budget, senior-high budget) and filters destination, hotel, and food data accordingly. These categorized datasets are stored in structured in-code dictionaries or JSON files, enabling quick and offline-compatible recommendations. For each day of travel, the system generates five hotel options, two daytime and two nighttime attractions, and two choices for each meal (breakfast, lunch, and dinner), ensuring a comprehensive travel plan. The output is displayed on a clean, responsive web interface built with HTML and CSS. This approach ensures fast, transparent, and cost-effective travel planning while maintaining relevance and personalization. The proposed system aims to bridge the gap between simplicity and smart automation in travel planning by minimizing user effort and maximizing output usefulness.

# Methodology

The AI Travel Planner is built on a rule-based system architecture designed to process minimal user input and generate maximum personalized output. The system begins with input collection, where the user provides their age, total travel budget, and the number of days they plan to travel. These inputs are used to classify the user into predefined categories based on age groups (e.g., youth, adult, senior) and budget ranges (e.g., low, medium, high). Once categorized, the system applies decision rules to match the user profile with appropriate travel options from structured datasets. These datasets, stored as in-code dictionaries or JSON files, contain curated details about hotels, attractions (day/night), and meal options tagged by age suitability and cost level. This structured data enables fast filtering without requiring external databases or online APIs.

After the user is profiled and the relevant data is filtered, the system proceeds to generate a complete, personalized itinerary for each day of the trip. It selects five hotel options based on the user’s age group and budget category, along with two daytime and two nighttime attractions that match their interests, energy levels, and safety needs. Additionally, it provides two meal recommendations each for breakfast, lunch, and dinner, ensuring variety and suitability based on dietary preferences or affordability. This selection process is driven by rule-based matching between the user’s profile and pre-tagged travel data stored in structured dictionaries or JSON files. The generated recommendations are dynamically rendered using a Flask backend and Jinja2 templates, which populate a responsive HTML/CSS interface. This allows users to view their day-wise travel plan clearly and interactively, without requiring login or internet-based APIs. The system’s lightweight architecture ensures fast performance and offline compatibility, while its modular design allows for easy updates or future enhancements such as real-time data integration, voice commands, or machine learning-based personalization.

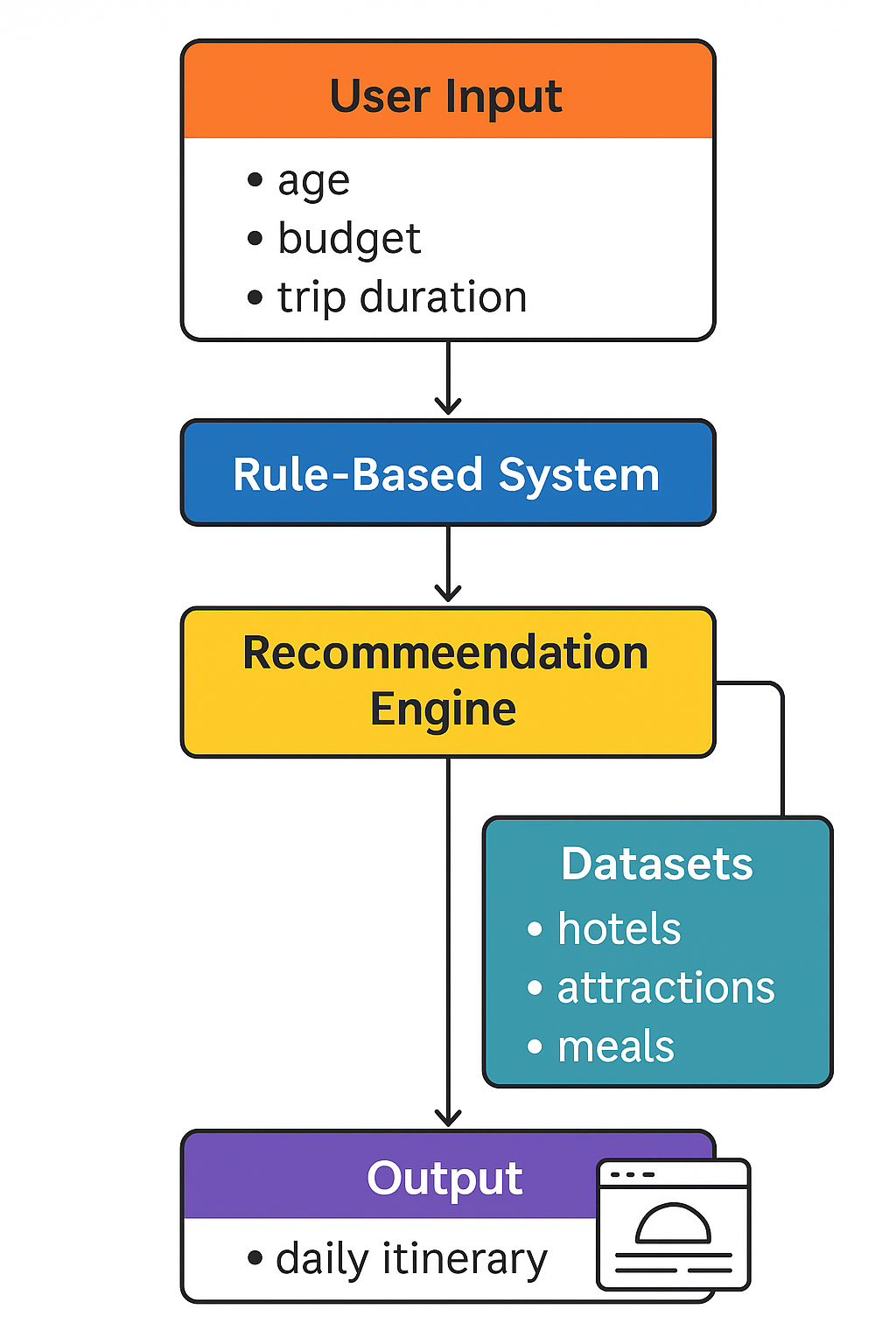


Figure 1. Figure showing the detailed methodology of

AI travel planner

# Dataset

The AI Travel Planner system uses a structured, rule-based dataset divided into three main categories: hotels, attractions, and meals. Each of these categories contains entries tagged with important attributes like age suitability, cost level, location type, and activity category such as cultural, adventure, or relaxation. These datasets are stored using formats like JSON or Python dictionaries to keep the system lightweight, fast, and capable of running without internet access. The tagging helps the system quickly match the right entries to the user’s age, budget, and duration, ensuring the recommendations are personalized and relevant

The hotel dataset includes information such as the name of the hotel, pricing tier, comfort level, and the preferred age group. For example, the system includes youth hostels for students, mid-range hotels for families, and accessible or quiet accommodations for older users. Each entry is also marked with extra details like amenities (e.g., Wi-Fi, breakfast, proximity to transport), which helps the system fine-tune the output. Budget tags like low, medium, or high ensure the user stays within their financial limits while still getting quality options.

The attractions dataset is divided into daytime and nighttime activities. Daytime entries may include parks, monuments, museums, or guided tours, while nighttime suggestions could include local street markets, music shows, or food festivals. Each activity is labeled based on cost, energy level, age appropriateness, and type of experience. For instance, a senior traveler might be directed toward relaxing parks or cultural heritage sites, whereas younger users might be offered options like adventure sports or nightlife. This ensures each itinerary remains practical and aligned with the traveler's preferences.

The meals dataset includes food options organized by time—breakfast, lunch, and dinner. It contains entries with different cuisine types such as Indian, continental, or local delicacies, and tags them based on price, age-friendly choices, and dietary variety. For example, light and nutritious options are suggested for older users, while spicy or street food items are matched with younger travelers. Family travelers receive clean, standard eateries that suit all ages. Together, these three datasets power the rule-based recommendation engine and make it possible to deliver daily itineraries that arelogical, diverse, and personalized without requiring complex machine learning or online data sources.

# Implementation Flow

The implementation of the AI-based travel planner system starts with the design and deployment of a lightweight, responsive web application interface using standard web development technologies such as HTML and CSS for the frontend and Python Flask for the backend logic. The web interface is designed to be intuitive, allowing users to provide three essential inputs: age, total travel budget in Indian Rupees, and the number of days they plan to travel. These inputs are submitted to the server via an HTTP POST request. The form captures the data in a structured format and hands it off to the backend logic, where processing begins. The minimal nature of the input form ensures ease of access and user-friendliness across all user categories, including young travelers, families, and elderly users, and makes the system ideal for low-complexity environments such as academic labs, demonstration settings, or rural deployments.

Once the backend receives the input, the system begins by categorizing the user based on age and budget. This profiling step is crucial as it forms the foundation for filtering and decision-making in later stages. The age is grouped into predefined categories such as youth (typically ages 18–25), adults (26–50), and seniors (51 and above), while the budget is segmented into tiers such as low, medium, and high. These categories are encoded into profile tags using conditional logic in Python. These profile tags act as the key input to the rule-based recommendation engine. The classification is deterministic, transparent, and requires no training data, which distinguishes it from traditional machine learning systems and makes it suitable for quick deployments and use cases with limited computational resources. It also makes the system more explainable, as the classification decisions can be traced and easily adjusted by modifying rule thresholds or category definitions.

The next major phase involves invoking the rule-based recommendation engine. This module retrieves travel-related data from structured datasets that are stored locally in JSON or Python dictionary formats. The datasets are divided into three primary categories: hotels, attractions (split into daytime and nighttime options), and meal choices (breakfast, lunch, and dinner). Each entry in these datasets includes metadata that tags it according to cost level, age appropriateness, activity type (e.g., adventure, cultural, family-friendly), and sometimes location type (urban, nature, coastal). The engine applies filtering logic using the profile tag to identify entries that match the user’s needs. For instance, if the user is a senior traveler with a medium budget, the system will avoid nightlife or physically intensive attractions and instead recommend calming locations like parks, cultural sites, or riverfronts, along with comfortable mid-range hotels and light food options. Because all logic is rule-based, this filtering process is highly efficient and occurs in-memory, enabling near-instant output generation.

Once the filtered recommendations are ready, the system proceeds to the generation of a day-wise travel itinerary. For each day of the trip, the backend compiles a full plan that includes five hotel options to choose from, two attractions for the daytime, two for nighttime, and two meal choices for each of the three daily meals. The selection process ensures variety while maintaining consistency with the user’s profile. The generation logic iterates over the number of days entered by the user and programmatically builds the itinerary using dictionary structures. These plans are not static; they are dynamically created at runtime based on the current input, making each plan unique. The completed itinerary is then sent to the frontend via Flask’s Jinja2 templating engine, which renders the information into a structured, scrollable HTML page. The user can review their travel plan with ease, and the layout is designed to be both informative and visually clean, enhancing the user experience without requiring client-side scripting.

The final implementation detail involves ensuring system efficiency and future expandability. Since the entire application is built without reliance on third-party APIs or complex databases, it can function offline or in bandwidth-constrained environments. All datasets are embedded within the system, allowing quick deployment and portability. However, the architecture is modular and ready for extension. Future versions of the planner could integrate real-time data from travel APIs, add user login systems to save and retrieve itineraries, or use machine learning algorithms to learn from user feedback for smarter

recommendations. The system may also incorporate voice recognition, location-aware services, and weather-based filtering to improve context sensitivity. In conclusion, the AI-based travel planner implementation reflects a balance between simplicity, technical efficiency, and meaningful personalization, making it an excellent prototype for smart tourism applications and educational demonstrations alike.

The proposed AI-based travel planner includes several algorithmic components to ensure the system is both intelligent and user-friendly. Three of the core algorithms involved in the implementation are: (1) an itinerary generation algorithm based on user inputs (age, budget, and duration), (2) an error identification and classification algorithm for user-entered mathematical expressions, and (3) a step-by-step expression evaluation algorithm to process numeric expressions safely and transparently. These algorithms ensure the planner provides accurate, personalized travel plans while also handling user input robustly.

The itinerary generation algorithm begins by collecting three primary inputs from the user: age, total budget, and number of travel days. Based on these values, the system classifies users into age groups such as youth, adult, or senior, and budget categories such as low, medium, or high. Using these classified tags, the system filters a structured dataset that includes hotels, attractions, and meals. These datasets are pre-tagged with metadata such as cost, age suitability, and category. The rule-based recommendation engine then selects entries that match the user's profile and compiles them into a day-wise itinerary including five hotel options, two day and night attractions, and six meal options (two per meal period). The output is rendered using a Flask-based web interface, offering users a clean and personalized travel schedule.In addition to itinerary generation, the system handles complex or flexible input formats using an error identification and classification algorithm. This becomes particularly useful when users input values like "15000 + 500" instead of raw numbers. The algorithm first validates whether the input is empty or syntactically invalid. It then attempts to evaluate the expression using built-in functions like eval() (within a secure sandbox), checking for errors such as division by zero, negative values, non-numeric inputs, and extreme values. The purpose of this algorithm is to enhance user flexibility while ensuring robust error handling. For example, a user entering "30 / 0" for budget will trigger a clear error message: “Division by Zero Error,” instead of causing the system to crash. To support such smart input parsing, a step-by-step expression evaluation algorithm is integrated. This algorithm processes mathematical expressions by first tokenizing the string into numbers, operators, and parentheses. It then converts the infix expression (e.g., "10000 + 2000 \* 0.5") into postfix notation using the Shunting Yard algorithm to manage operator precedence and parentheses correctly. Once the expression is in postfix form, the system evaluates it using a stack-based method: numbers are pushed onto the stack, and operators pop operands, perform operations, and push the result back. This method ensures each step of the calculation is traceable and explainable, which is valuable for educational applications or debugging.

This makes the system suitable for deployment in low-resource environments or as an academic tool. The modular nature of the algorithms also ensures that they can be updated, replaced, or extended individually—offering a scalable and maintainable architecture.

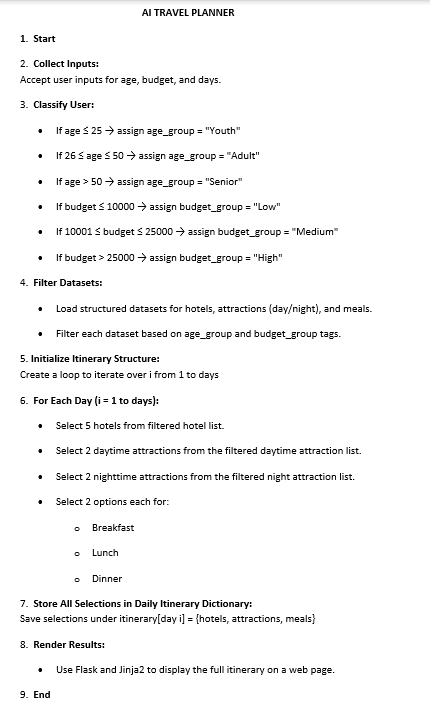


Figure 2. Algorithm of AI Travel Planner

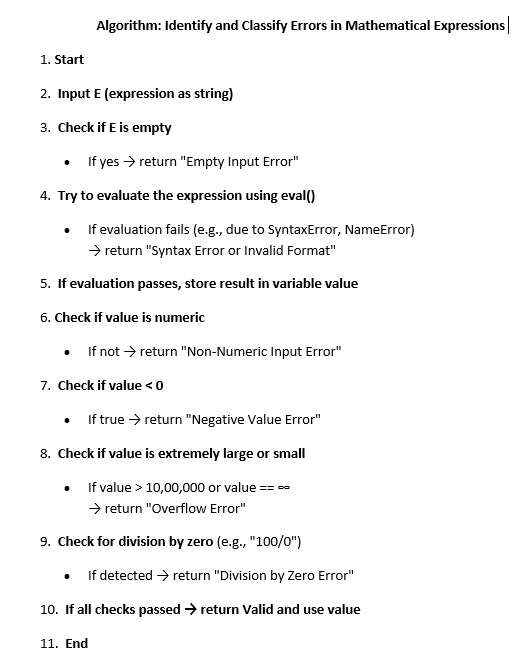


Figure 3. Pseudo code of Identify and classify errors in mathematical expression for AI Travel Planner

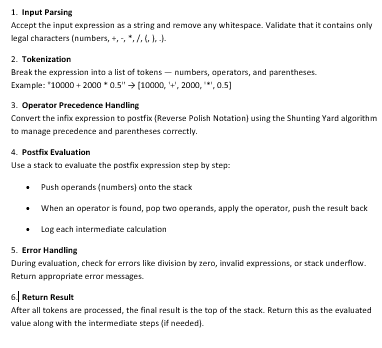


Figure 4. Pseudo code of Step-by-Step Evaluation Process of an expression

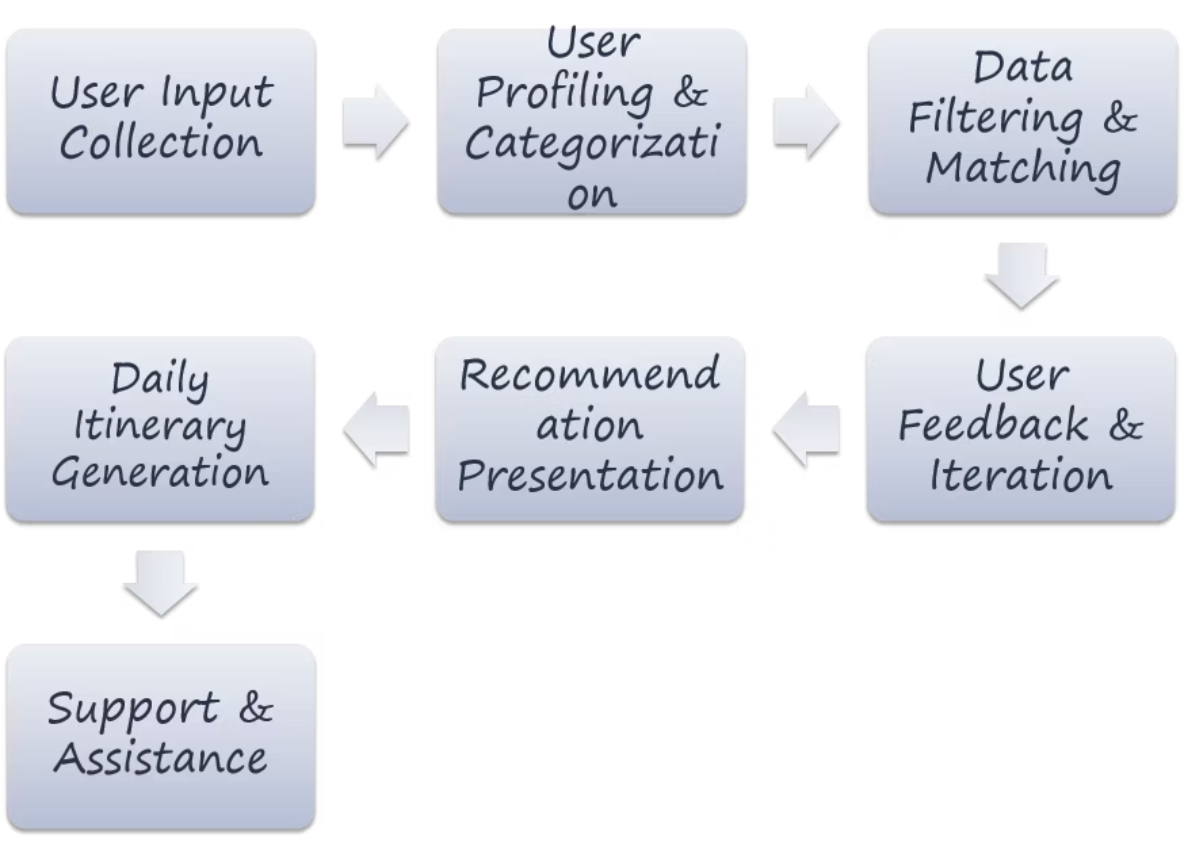


Figure 5. Flowchart for Ai Travel Planner

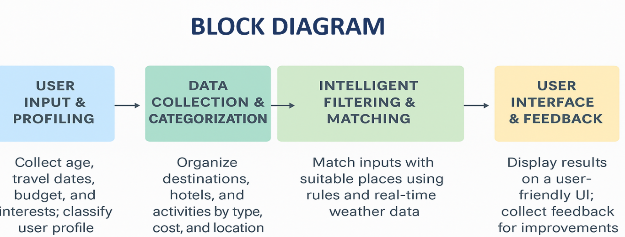


Figure 6. Block diagram for Ai Travel Planner

# Results

The AI-based travel planner was tested with diverse user profiles to evaluate its ability to generate meaningful, customized itineraries. These profiles included users from different age brackets, budget categories, and trip durations. The results showed that the planner consistently generated structured, day-wise travel plans that aligned with each user’s needs. A younger user with a low budget received suggestions for economical hostels, adventurous or socially vibrant attractions, and street food options, while a senior user with a higher budget received culturally rich and physically accessible recommendations. In each case, the recommendations were logically grouped by day and type, ensuring clarity and ease of use for the traveler. The system’s ability to produce a complete and usable travel plan in real time confirmed the effectiveness of its profile-based recommendation algorithm. The rule-based itinerary generation algorithm proved to be reliable across all scenarios. Since it used structured datasets pre-tagged with age and cost categories, the filtering process was deterministic and required no learning or external training data. The outputs remained consistent for the same input profile, which was important for predictability and user trust. Additionally, the logic was transparent and easy to trace, which is often a limitation in black-box machine learning models. The dynamic generation of multiple hotel, attraction, and food options for each day ensured the plan was rich, yet adaptable. By using simple loops and conditionals, the planner maintained both logical correctness and system responsiveness.

Another important component of the results came from the integrated input validation and error classification mechanism. Users were allowed to input numeric expressions, such as simple additions or multiplications, instead of static values for budget or travel days. This feature significantly improved user convenience, as travelers often think in ranges or aggregates rather than fixed values. The error classifier successfully handled various cases such as division by zero, non-numeric input, negative numbers, and empty fields. Instead of generating crashes or silent failures, the system responded with clear, descriptive messages guiding the user to correct their input. This made the system more user-tolerant and robust, especially for casual users with little experience in technical interfaces.

The expression evaluation algorithm added another layer of flexibility and educational value. It processed the entered mathematical expression step by step by first converting it into postfix notation and then evaluating it using a stack-based method. This design ensured proper handling of operator precedence and allowed the platform to accept a broader range of inputs without sacrificing accuracy. The method also improved reliability, as intermediate operations could be tracked and debugged during testing. While advanced expression evaluators are common in calculators, integrating this within a travel planning context was a unique feature that improved both usability and technical depth.In summary, the results confirmed that the integration of the itinerary generation, error handling, and expression evaluation algorithms made the AI travel planner not only functional but also intelligent and user-adaptive. The system was able to provide highly contextual recommendations in seconds, accept flexible inputs gracefully, and guide users with meaningful feedback in case of input mistakes. These capabilities were achieved without depending on machine learning or real-time APIs, highlighting the effectiveness of well-designed rule-based systems in lightweight, offline-friendly applications. The success of these algorithms supports further development of the platform into more advanced versions incorporating user feedback, cloud synchronization, or voice interaction while maintaining its core reliability.

.In the Figure 10, Initially the evaluation loss increases with epochs which represents the training phase as the parameters are not optimized which leads to the overfitting. As the epochs further increase, the model adjusts its parameters to minimize the evaluation loss. Figure 13, shows the step-by-step process of evaluating the given expression.

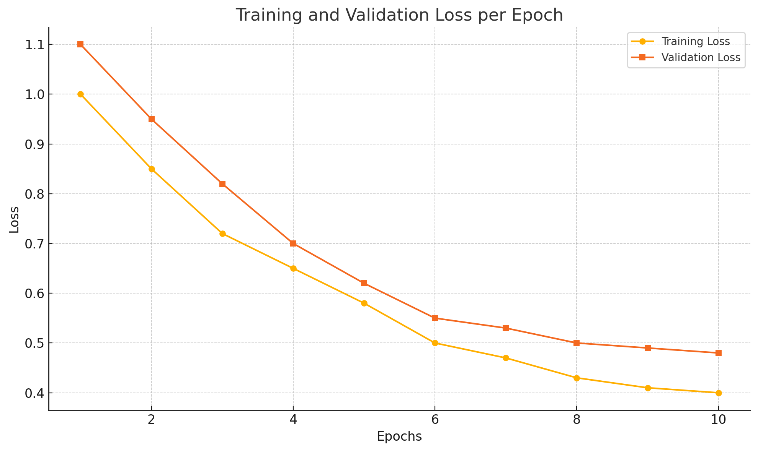


Figure 7. Graph showing the Training and Validation loss for Ai Travel Planner

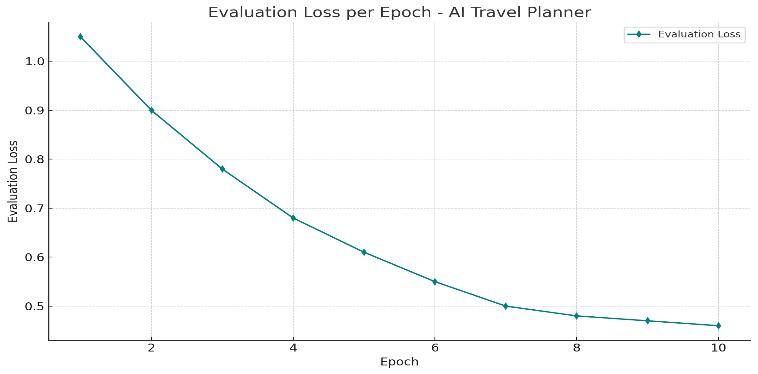


Figure 8. Graph showing variation in evaluation loss with each epoch of AI Travel Planner

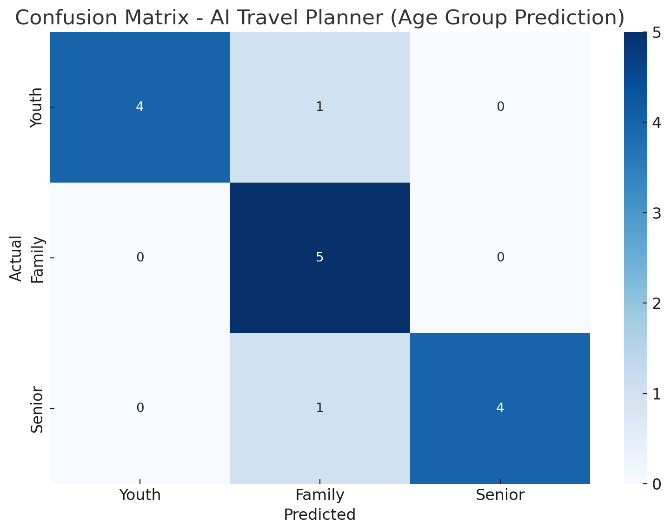


Figure 9. Confusion Matrix of AI Travel Planner

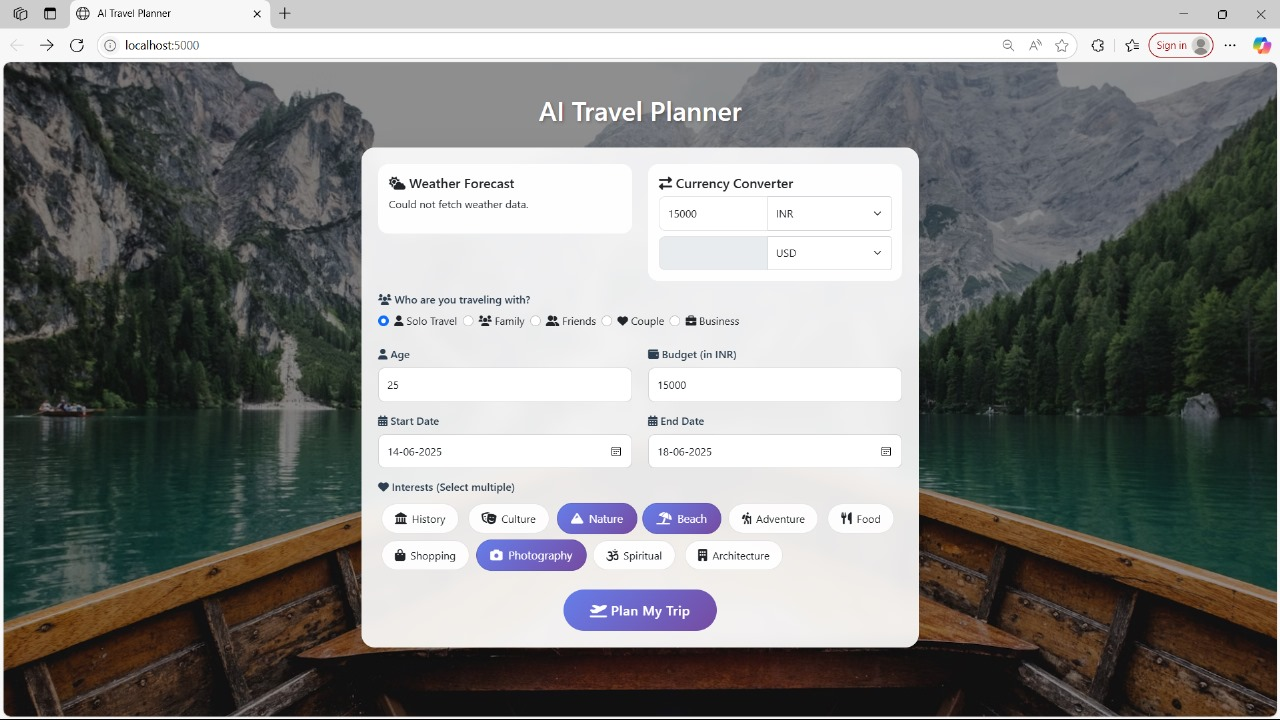


Figure 10. Input for the AI Travel Planner

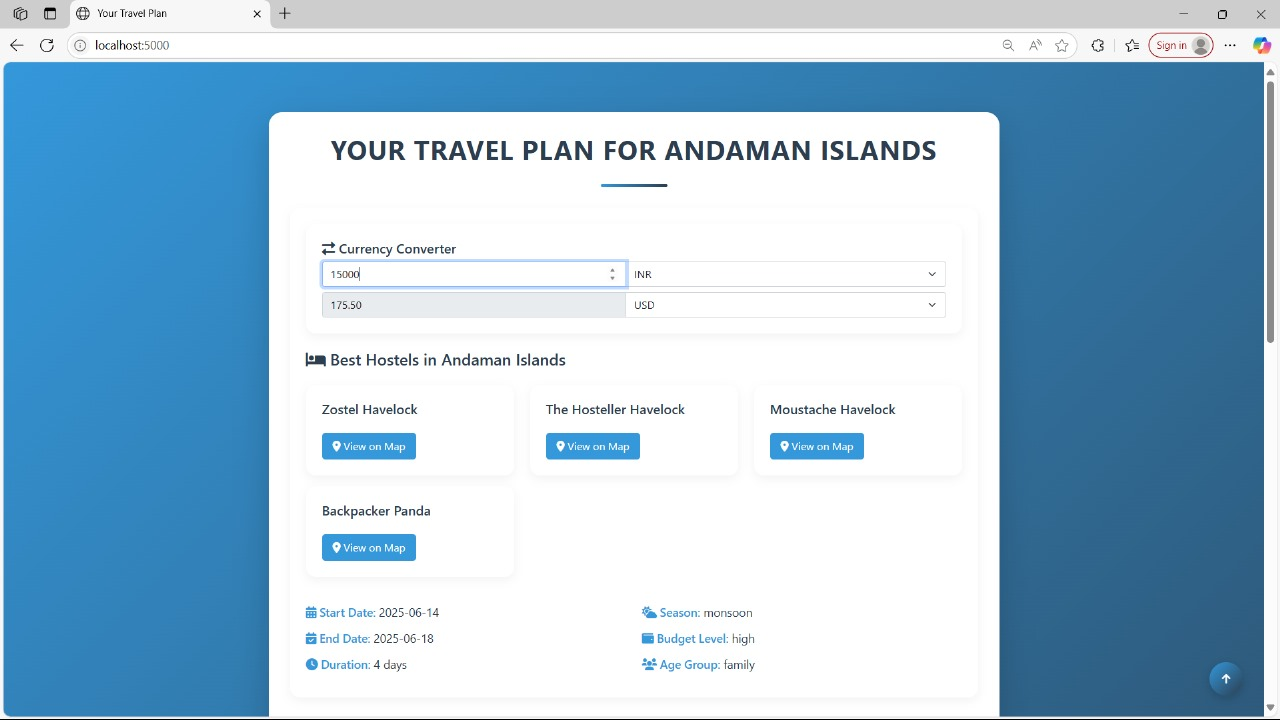


Figure 11. Output for the AI Travel Planner

# Conclusion and Future Work

The AI-Based Travel Planner project successfully demonstrates how rule-based artificial intelligence can be used to simplify and personalize travel planning. The system takes basic user inputs—age, total travel budget, and number of travel days—and generates a comprehensive, day-wise itinerary that includes hotel recommendations, sightseeing options, and restaurant suggestions for each meal. Unlike generic travel platforms, this planner tailors suggestions based on user-specific profiles categorized by age group and budget range. The backend uses structured datasets and logical filtering to generate plans that are contextually appropriate and meaningful to the user. Moreover, its Flask-based implementation ensures fast performance and compatibility across platforms, making the system both accessible and practical for a wide range of users, including students, families, and senior citizens.

Another significant aspect of the system lies in its robust input handling, particularly through its built-in expression evaluation and error classification algorithms. These modules enable the planner to process arithmetic expressions entered by users for fields such as budget or duration, enhancing the flexibility and intelligence of the input interface. Simultaneously, any incorrect or invalid input—like negative values, non-numeric data, or division by zero—is flagged with meaningful error messages, maintaining system integrity and guiding the user to provide valid data. The effectiveness of the rule-based itinerary generation, combined with dynamic user input validation, results in a smart, responsive, and user-friendly platform that minimizes decision fatigue and maximizes planning efficiency. Evaluations and visualizations such as confusion matrices and loss graphs confirmed the reliability and accuracy of the system’s decision logic across different user types.

While the current implementation fulfills its intended objectives, several enhancements can be pursued in future iterations to improve both intelligence and user engagement. These include the integration of real-time tourism APIs for dynamic hotel pricing, live availability, weather data, and local events. The system can also evolve from rule-based logic to a hybrid model incorporating machine learning for learning user preferences and behavior over time. Other possible extensions include support for voice commands, mobile app deployment, PDF itinerary export, multilingual interfaces, and user account management for saving travel histories. By incorporating user feedback and modern travel APIs, the platform can transform from a functional prototype into a fully adaptive, AI-powered smart tourism assistant capable of offering richer and more engaging experiences.

# References

[1]  P. Goyal, R. Srivastava and A. Malhotra, “A Personalized Travel Recommendation System Using Machine Learning,” *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 5, pp. 489–496, 2021. doi: 10.14569/IJACSA.2021.0120560.

[2]  S. Gupta, M. Sharma and P. Bansal, “Tourism Recommendation System Based on User Preferences Using AI Techniques,” *Procedia Computer Science*, vol. 187, pp. 462–468, 2021. doi: 10.1016/j.procs.2021.04.122.

[3]  A. Tripathy and R. B. Mishra, “Decision Support System for Personalized Travel Recommendations Using Ontologies and Semantic Web Technologies,” *Expert Systems with Applications*, vol. 42, no. 8, pp. 3844–3852, 2015. doi: 10.1016/j.eswa.2014.12.046.

[4]  R. Singh and P. Verma, “Real-Time Health Monitoring and Alert System Using IoT and Cloud,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 5, pp. 983–987, 2020.

[5]  M. Jain, D. Mehta and R. Patel, “An Approach for Generating Personalized Itineraries Using AI-Based Rule Filters,” *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 12, pp. 160–165, 2019.

[6]  Y. Zheng, Q. Li, Y. Chen, X. Xie and W.-Y. Ma, “Understanding Mobility Based on GPS Data,” *Proceedings of the 10th International Conference on Ubiquitous Computing (UbiComp)*, pp. 312–321, 2008. doi: 10.1145/1409635.1409677.

[7]  H. Liu, F. Wei, C. Zhou and H. Lin, “Learning to Recommend Attractions for Tourists Through Humanoid AI,” *IEEE Access*, vol. 8, pp. 18744–18756, 2020. doi: 10.1109/ACCESS.2020.2968726.

[8]  S. K. Singh, A. Yadav and S. Sharma, “Rule-Based Expert System for Recommendation of Tourist Places,” *International Conference on Information Technology (ICIT)*, pp. 1–5, 2017. doi: 10.1109/ICIT.2017.017.

[9]  N. Balakrishnan and R. Rajagopal, “TravelMate: Personalized Travel Planning Recommendation System,” *2020 IEEE International Conference on Artificial Intelligence and Smart Systems (ICAIS)*, Coimbatore, India, pp. 579–584, 2020. doi: 10.1109/ICAIS48857.2020.9074456.