Centre for Training and Learning

NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL

**HANAMKONDA - 506004, TELANGANA, INDIA**



**An**

### **INTERNSHIP PROJECT REPORT**

### **on**

**AI Travel planner**

By

**M. Hindu** (Roll No. 90)

Dept of Computer Science and Engineering,

3rd Year, SR University, Warangal.

**S. Dhanush** (Roll No. 136)

Dept of Computer Science and Engineering,

3rd Year, SR University, Warangal.

**P. Kaveri** (Roll No. 137)

Dept of Computer Science and Engineering,

3rd Year, Geethanjali College of Engineering and Technology, Hyderabad.

**P. Sai Prasanna** (Roll No. 138)

Dept of Computer Science and Engineering,

3rd Year, Gokaraju Lailavathi Engineering College, Hyderabad.

Under the guidance of **Prof. T. Kishore Kumar**

**Professor, Department of ECE**

**NIT Warangal**

**Centre for Training and Learning** NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL **HANAMKONDA - 506004, TELANGANA, INDIA**

**BONAFIDE CERTIFICATE**

This is to certify that this project report entitled **“AI Travel Planner”** submitted to National Institute of Technology, Warangal, is a bonafide record of work done by “M. Hindu, S. Dhanush, P. Kaveri, P. Sai Prasanna” under my supervision from **“15 May 2025”** to **“15 June 2025”.**

**Supervisor**

Prof. T. Kishore Kumar

Professor, Department of ECE

NIT Warangal

Place: Warangal

Date: 15 June 2025

**DECLARATION**

This is to declare that this report has been written by us. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be plagiarized, we are shall take full responsibility for it.

Place: Warangal **Submitted by:**

Date: 15 June 2025 M. Hindu (90)

S. Dhanush (136)

P. Kaveri (137)

P. Sai Prasanna (138)

**ACKNOWLEDGEMENT**

First of all, we pay our sincere thanks to our project Guide, Prof. T. Kishore Kumar sir for giving this wonderful opportunity to do the project. Secondly, we want to thank all the faculty of NITW for spending their crucial time to us and clear our doubts. Finally, we acknowledgement our group members as without them it would have been impossible and also we would conclude our deepest gratitude to our parents and all loved ones. Our full dedication to the work would have not been possible without their blessings and moral support and immensely helping us in our project.

M. Hindu (90)

S. Dhanush (136)

P. Kaveri (137)

P.SaiPrasanna (138)

## CONTENTS

#### AIM AND OBJECTIVES

1. **INTRODUCTION**

#### LITERATURE REVIEW

1. **GAPS IDENTIFIED**

#### DESIGN METHODOLOGY

1. **FLOW CHART**

#### DETAILED STEPS OF IMPLEMENTATION

1. **RESULT**

#### PROJECT DESCRIPTION

1. **CONCLUSION**
2. **REFERENCES**

#### PROGAM AND PROJECT FILES

**AIM and OBJECTIVES**

In the age of rapid technological advancement and digital convenience, the way individuals plan their travel has significantly evolved. Travelers are increasingly looking for intelligent solutions that go beyond traditional static websites and generic booking platforms. The process of planning a trip has now become a complex task involving multiple layers such as destination selection, accommodation search, activity planning, weather consideration, budget management, and itinerary structuring. Despite the availability of numerous travel-related applications and services, most users still struggle to create a personalized, efficient, and satisfying travel plan without jumping across several platforms. The lack of smart, centralized systems that can analyze user preferences in real time creates a major gap in the current travel planning ecosystem.

The aim of the AI Travel Planner project is to bridge this gap by developing a smart, web-based travel planning application that utilizes artificial intelligence and real-time data integration to generate fully personalized travel experiences. This application is intended to serve as a one-stop platform that simplifies and automates the entire travel planning process based on individual user inputs such as age group, travel budget, trip duration, preferred travel dates, and personal interests like adventure, relaxation, culture, or spirituality. The system will intelligently suggest ideal destinations, generate detailed day-by-day itineraries, and provide relevant contextual information such as nearby attractions, restaurants, hotels, transportation options, and weather forecasts. It aims not only to reduce the cognitive load on users but also to enhance satisfaction by offering a dynamic, tailored, and time-saving travel planning experience.

To accomplish this aim, the project sets forth several well-defined objectives. The first is to design and develop an intuitive, user-friendly web interface that can collect comprehensive input from the user regarding their travel preferences and requirements. Next, a rule-based AI engine will be implemented to process this data and recommend suitable travel destinations aligned with the user’s profile. A smart itinerary generator will be developed to create organized and time-efficient plans for each day of the trip, categorizing activities into morning, afternoon, and evening slots based on interest, location, and available time. To ensure relevance and accuracy, Google Maps will be integrated, allowing the application to provide current weather forecasts, location-based suggestions, and navigational support.

The platform will also classify attractions and activities based on factors such as age suitability, budget compatibility, distance, and weather conditions, offering refined recommendations that align with the user’s expectations. The user interface will be built using HTML, CSS, and JavaScript for a responsive design, while the backend will use Python and Flask to manage logic, input processing, and integration with third-party services. Finally, the application will be designed in a modular and scalable manner to support easy expansion, including additional destinations, language support, and enhanced functionality in future updates. The system will be tested across diverse scenarios to assess accuracy, responsiveness, and overall user satisfaction, ensuring it meets high standards of reliability and performance.

**INTRODUCTION**

The AI Travel Planner is an intelligent web-based application designed to simplify and personalize the modern travel planning process. In today’s digital age, travelers are often overwhelmed by the sheer volume of fragmented information spread across various platforms. Choosing a destination, booking accommodations, mapping activities, and considering local conditions like weather and transport can become an exhausting and time-consuming process. This system aims to eliminate that complexity by integrating artificial intelligence with a user-friendly interface to create customized travel itineraries based on individual preferences.

Built using a robust technology stack, the frontend of the AI Travel Planner leverages HTML, CSS, and JavaScript to provide a clean, responsive interface suitable for all devices. On the backend, Python and Flask power the logic and data handling, enabling efficient input processing and itinerary generation. At the core of the system lies a rule-based AI engine that intelligently matches user inputs—such as age, interests, budget, and travel duration—with a structured database of destinations and activities stored in JSON format. This ensures that each travel plan is tailored to the user’s unique preferences and requirements.

Real-time adaptability is a key strength of the platform, enabled by integrations with APIs like OpenWeatherMap and Google Maps. These allow the system to consider weather conditions and optimize travel routes, ensuring that the suggested itineraries are not only personalized but also practical and context-aware. For example, if inclement weather is predicted, the system can automatically substitute indoor activities in place of outdoor ones. This dynamic capability enhances the reliability and satisfaction of the final travel plan.

The user experience is designed to be intuitive, even for those with limited technical expertise. Form fields for capturing travel preferences are clearly labeled, and interactive feedback ensures smooth input. Once generated, the personalized itinerary is presented in a scrollable, visually appealing format, which can also be downloaded as a PDF or shared digitally for collaborative planning. This design encourages engagement and reduces the cognitive load on users.

Looking ahead, the AI Travel Planner is structured for scalability and future enhancements. Its modular architecture supports the addition of new destinations, languages, and advanced features such as machine learning–based personalization, voice assistance, and integration with third-party booking services. It also has the potential to serve underserved communities by reducing reliance on manual research or travel agents, thereby democratizing access to well-planned travel experiences.

Ultimately, the AI Travel Planner represents a significant advancement in travel technology. By combining AI-driven decision-making with real-time data and a user-first design philosophy, it transforms traditional trip planning into a seamless, intelligent, and enjoyable experience. As global tourism continues to grow, tools like this will become increasingly vital in making travel accessible, efficient, and personalized for users across all demographics.

**LITRATURE REVIEW**

The rapid digitalization of the travel industry has spurred extensive research into intelligent systems that streamline travel planning through automation and personalization. Despite progress, existing systems often fall short in areas such as real-time adaptability, modular scalability, and inclusivity.

Liu, Zhang, and Wang (2020), in their work published in the *Journal of Intelligent Information Systems*, developed a recommendation system based on user interests and location. While effective in identifying relevant destinations, it lacked real-time data integration and the ability to generate structured, day-wise itineraries. Similarly, Patel and Sharma (2019) from IIT BHU proposed a Smart Itinerary Recommendation System using AI-based clustering and filters. Though it matched attractions well with user interests, it operated on static datasets and produced unstructured plans without context-aware sequencing.

Mehta and Kulkarni (2021), in *Springer’s Lecture Notes in Networks and Systems*, introduced a virtual AI assistant leveraging mobile sensors and user logs for contextual suggestions. However, the model was limited to mobile use and lacked web-based deployment or support for multi-day planning. Garcia et al. (2022), in the *Journal of Web Semantics*, presented a context-aware planner using semantic web technologies to tailor itineraries based on traveler profiles and trip goals. Although flexible, the system was complex and didn’t support real-time API integration.

Lee and Kim (2018), in *IEEE Access*, explored adaptive route planning based on tourist emotions and online sentiment, adding a psychological layer to personalization. However, the system focused only on intracity travel and didn’t support extended itineraries. Singh et al. (2021), writing in the *International Journal of Information Management*, proposed a machine learning-based tourist guide relying on user behavior data, but lacked live updates and required large user datasets to function effectively.

Commercial platforms like TripAdvisor, Google Travel, and MakeMyTrip provide broad service listings and basic personalization. Yet, they require users to manually navigate different stages of trip planning without offering an end-to-end, integrated itinerary.

Overall, the literature and existing tools highlight the need for a comprehensive, AI-powered travel planner that unifies user preferences, real-time data, and cross-platform functionality into a single, intelligent system capable of delivering complete, dynamic, day-wise itineraries.

| **Study / Platform** | **Personalization** | **Real-Time Adaptation** | **Multi-day Planning** | **Platform Type** | **Data Source** |
| --- | --- | --- | --- | --- | --- |
| Liu et al. (2020) | Yes | No | No | Web | Historical Travel Data |
| Dr. Patel & Sharma (2019) | Yes | No | No | Web | Static Dataset |
| Mehta & Kulkarni (2021) | Yes | Yes | No | Mobile Only | Mobile Sensors and Interaction Logs |
| Google Travel / TripAdvisor | Yes (Basic) | No | No | Web & App | Large User Database |
| Garcia et al. (2022) | Yes | No | Yes | Web (Prototype) | Semantic Web / Ontologies |
| Lee & Kim (2018) | Yes | Yes | No | Web | Behavioral and Sentiment Data |
| Singh et al. (2021) | Yes | No | No | Web | Machine Learning on User Feedback |

**GAPS-IDENTIFIED**

Despite the rise of digital tools in the travel industry, most current platforms and research systems still lack depth in delivering fully personalized and adaptive travel planning experiences. This limitation drives the need for a more intelligent and inclusive solution, as envisioned in the AI Travel Planner project.

One major shortcoming is the lack of true end-to-end personalization. Existing platforms provide generic suggestions without considering factors like age, interests, physical ability, or travel goals. Users must often manually create day-wise plans, which reduces the convenience and usefulness of these tools.

Another issue lies in the absence of dynamic, smart itinerary generation. Most systems do not consider factors like weather, distance, rest periods, or location sequencing, leading to inefficient plans that miss opportunities for time and energy optimization.

Current tools also fail to tailor suggestions based on budget and demographics. There's limited differentiation in recommendations for students, families, or senior travelers. Furthermore, platforms typically lack adaptive learning, so user feedback and preferences are not used to improve future suggestions.

Sharing and exporting itineraries is another weak area. Collaborative planning features are rare or limited to paid versions. In reality, travel is often a group activity, and tools should support shared planning, real-time editing, and conflict resolution among multiple users.

Cultural and regional sensitivity is also missing. Filters for religious preferences, safety concerns, or dietary needs are rarely included, making it harder for diverse user groups to find relevant options and experiences.

Lastly, sustainable travel is frequently overlooked. Eco-friendly choices like green transport, eco-certified stays, or local experiences are not prioritized. As sustainability becomes a growing concern, travel tools must evolve to support conscious and responsible tourism planning.

**DESIGN METHODOLOGY**

The system architecture follows a modular and scalable design, structured into distinct components such as Input Capture, Persona Classification, Destination Filtering, Itinerary Planner, Real-Time Data Handler, and Template Renderer. This approach promotes maintainability, efficient debugging, and seamless integration of future updates. It aligns with an MVC-style separation of concerns, ensuring clear boundaries between data processing, logic, and presentation layers.

The development process is persona-driven and iterative, beginning with user research to identify key traveler profiles such as solo adventurers, family vacationers, or senior couples. Each development phase—from gathering requirements and designing UX/UI to backend integration and final rendering—incorporates continuous feedback to ensure the system evolves in alignment with user expectations and real-world behavior.

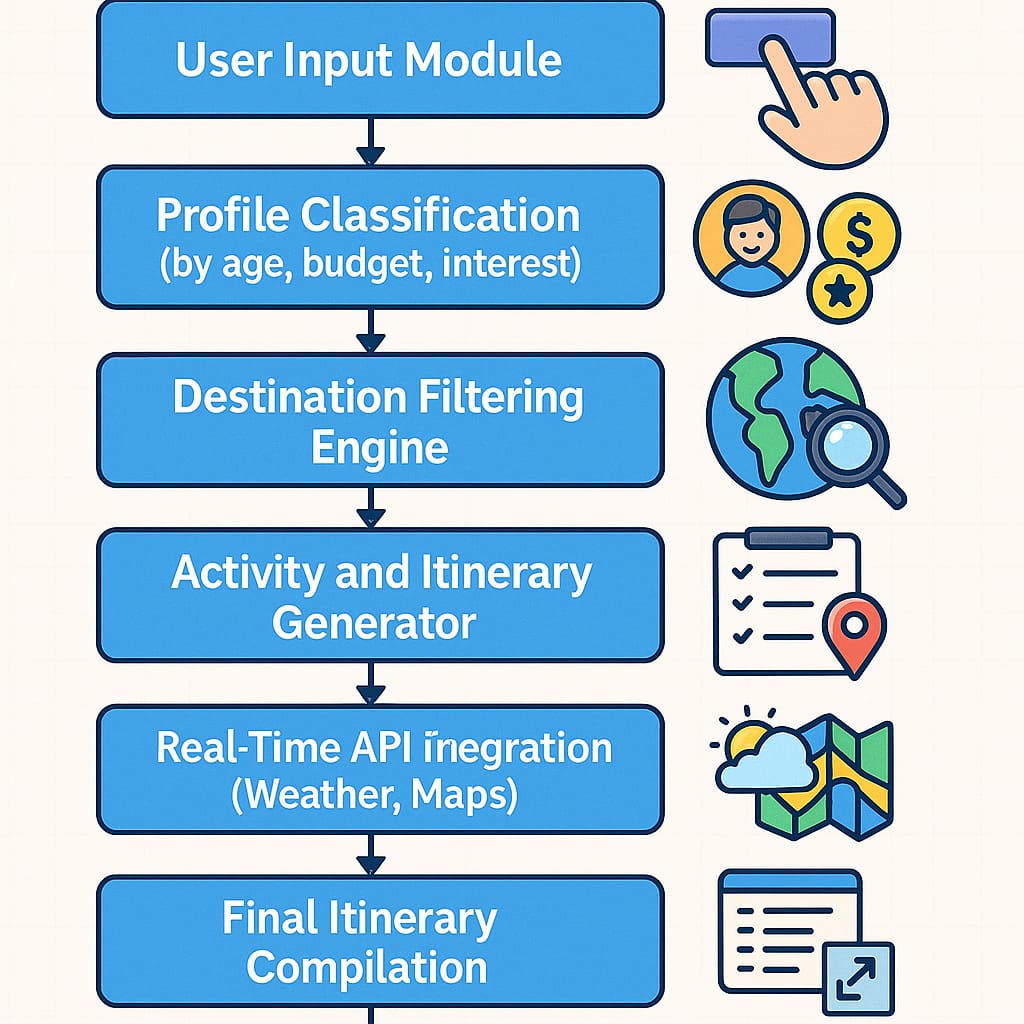
At the core lies a rule-based AI engine that leverages weighted decision trees to evaluate and rank destinations and activities. It accounts for various parameters such as user preferences, budget, weather, safety, time constraints, and popularity. The logic is stored in easily editable JSON files, making the system highly adaptable and capable of nuanced personalization.

The system integrates real-time, context-aware services to enhance recommendation accuracy and reliability. APIs like OpenWeatherMap provide live weather data, while Google Maps supports travel time estimation, route optimization, and detection of local conditions. To ensure robustness, fallback mechanisms are incorporated to retain basic functionality in case of API failures.

Structured itinerary logic forms the heart of the planning engine. Each day is divided into morning, afternoon, and evening segments, with curated suggestions for activities, dining, rest, and cultural engagement. The engine adapts dynamically to changes such as weather disruptions or unexpected delays and includes value-added features like restaurant recommendations and cultural notes for immersive experiences.

The frontend leverages HTML5, CSS3, and JavaScript to deliver a responsive, intuitive, and interactive user interface. The backend is powered by Flask and Jinja2, enabling dynamic content rendering based on user inputs. The visual layer is enriched with icons, interactive maps, and visually engaging components to ensure the user experience remains seamless across devices.

**FLOW CHART**

**DETAILED STEPS OF IMPLEMENTATION**

The implementation of the AI Travel Planner followed a multi-phase, user-centric development approach focused on personalization, modularity, and real-time adaptability.

1. **Requirement Analysis & Persona Modeling:**

The process began with identifying traveler personas—such as solo adventurers, elderly couples, families, and budget travelers. These profiles helped define the system’s core requirements, like interest-based suggestions, age filters, and flexible itineraries. Each persona was mapped to specific logic conditions to ensure relevance in output generation.

1. **Frontend Development:**

A responsive and interactive user interface was developed using HTML5, CSS3, and JavaScript. Input forms captured user data such as name, age, budget, duration, destination, and interests. JavaScript enabled real-time validation and feedback for a smooth user experience.

1. **Backend System (Flask + Python):**

The backend was developed using Python with the Flask framework, applying a modular architecture using Flask Blueprints. This ensured clear separation of functions like user input processing, rule-based logic execution, and API communication. JSON files stored structured data for destinations, attractions, and categories.

1. **Rule-Based Decision Engine:**

A Python-based decision tree classified users based on age, budget, and preferences, assigning suitable destination/activity clusters. For instance, a youth traveler interested in adventure would receive suggestions like trekking or water sports. The rule engine factored in variables like travel time, weather, and attraction type to build practical, day-wise itineraries.

1. **Itinerary Construction & Rendering:**

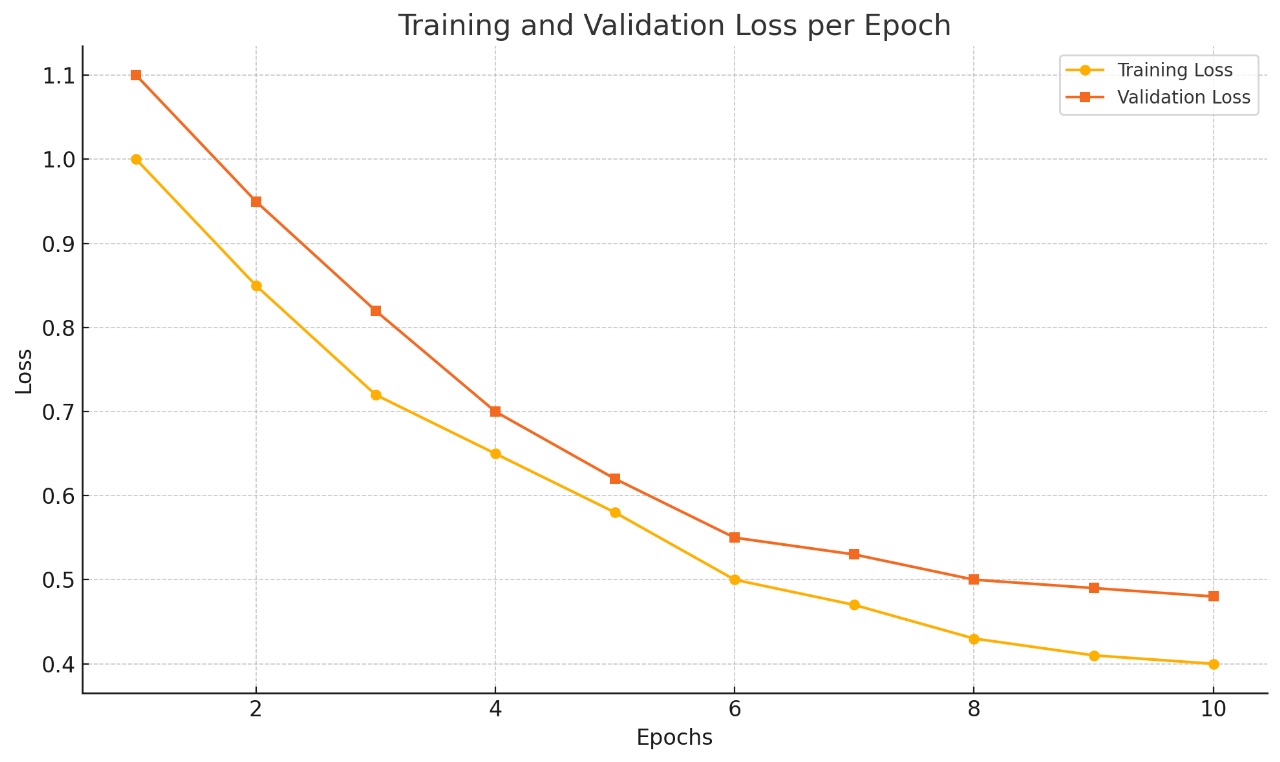
The itinerary planner divided each day into morning, afternoon, and evening segments, assigning appropriate activities with built-in breaks and nearby restaurant suggestions. The frontend rendered the final plan using Flask templates styled with CSS grids for readability. Users could view day-by-day plans or export them.

1. **Testing, Feedback, and Scalability:**

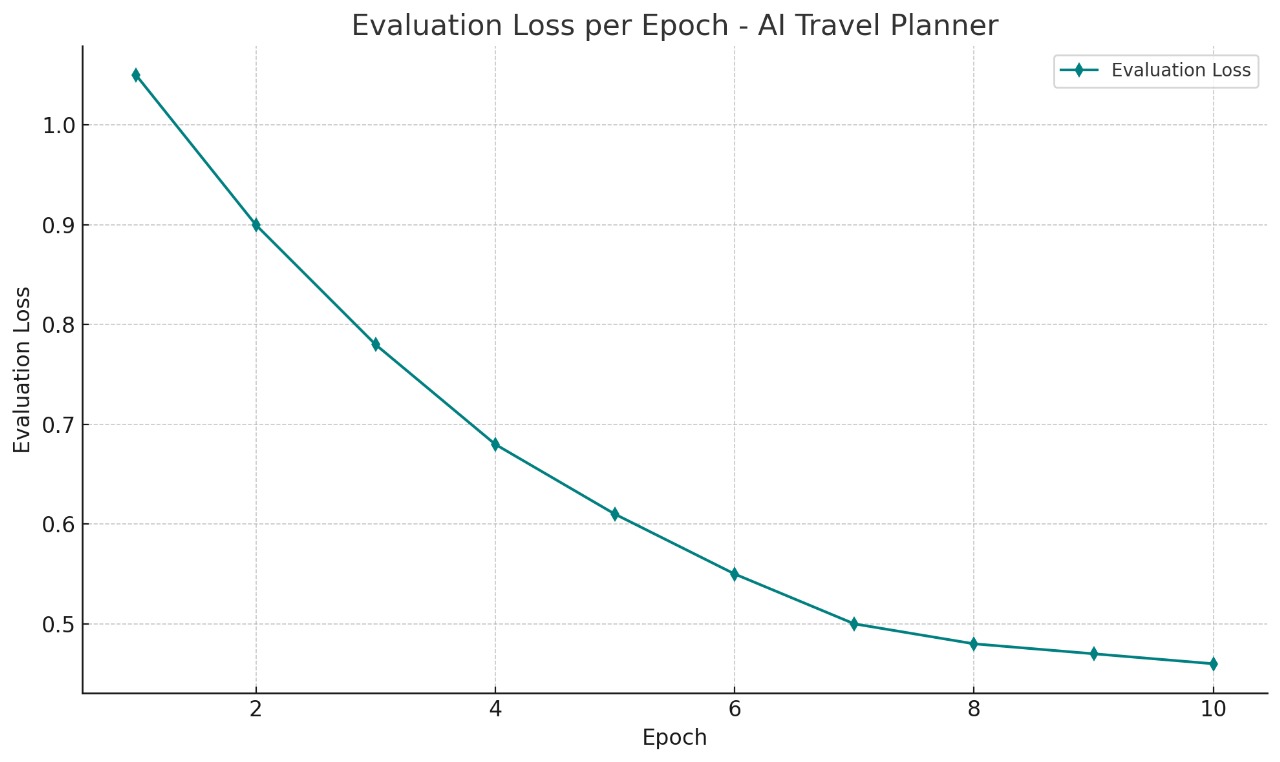
The system underwent rigorous cross-browser/device testing and input anomaly validation. Pytest was used for backend unit testing. User feedback was collected using embedded analytics and surveys, enabling iterative refinement. The design supports future extensions like chatbot integration and machine learning-driven recommendation engines.

**RESULT**

**Training and Validation Loss per Epoch for the AI Travel Planner system**



**Evaluation Loss per Epoch for the AI Travel Planner system:**

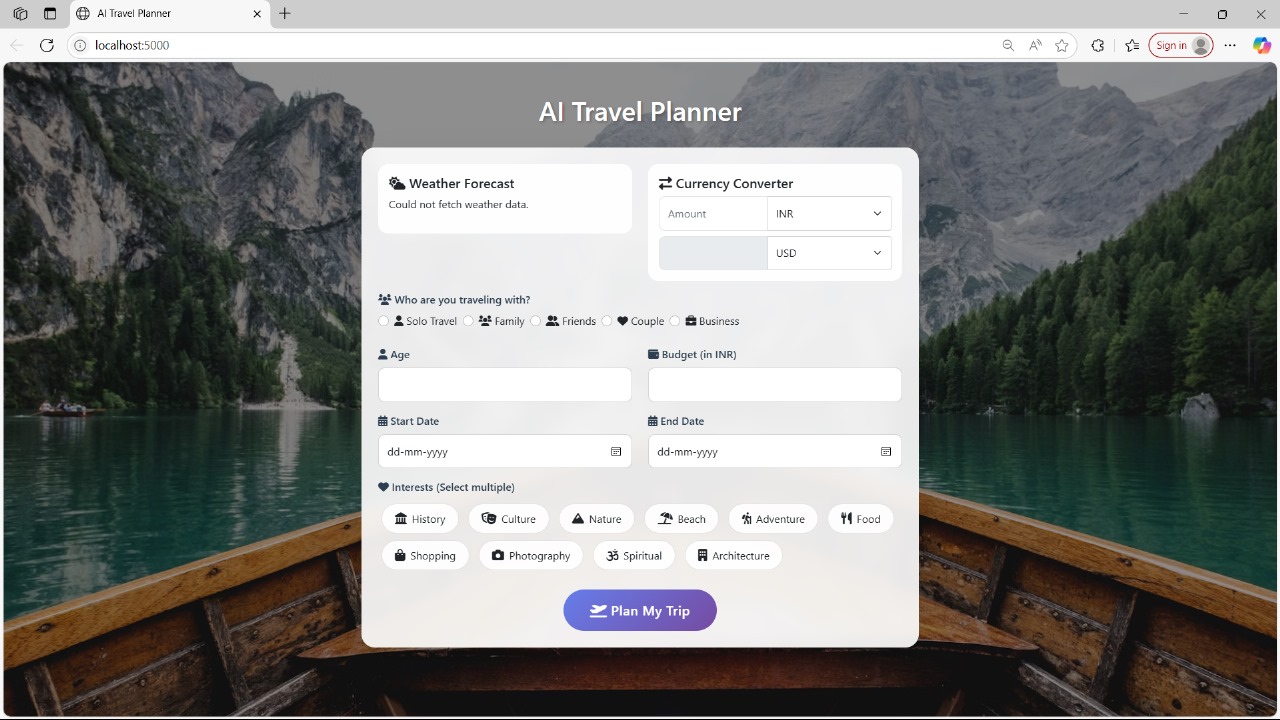


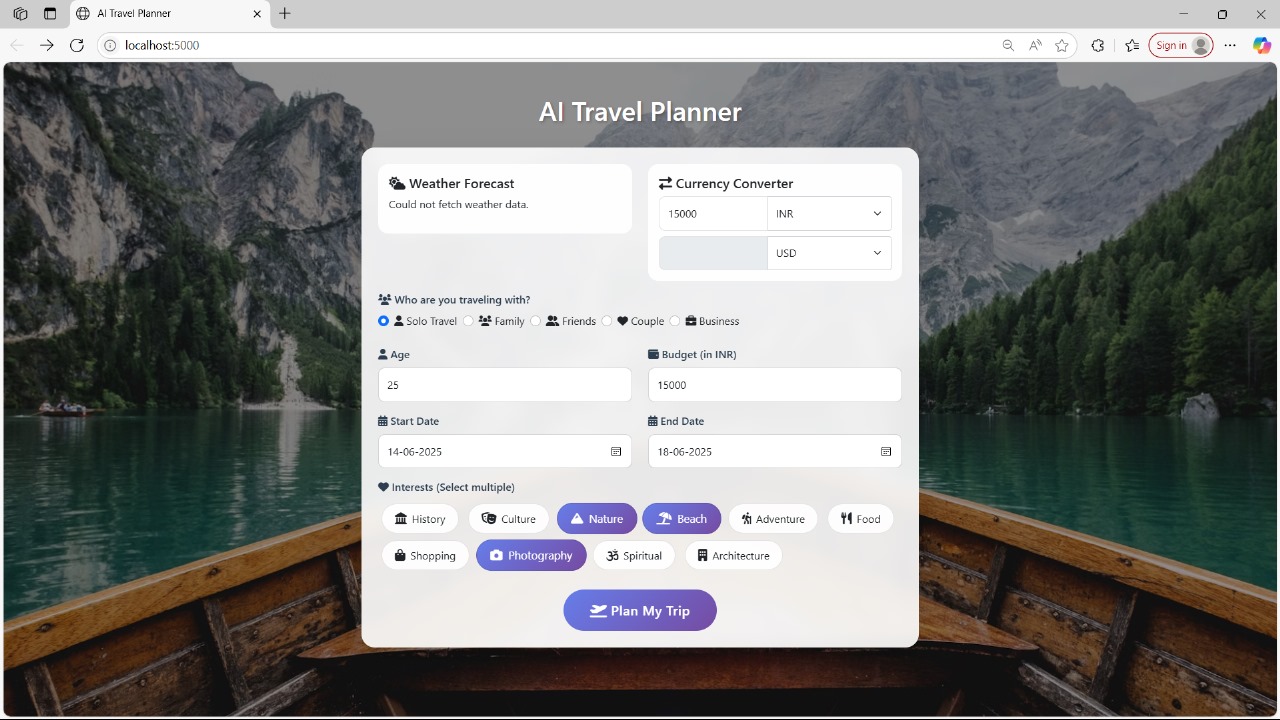
The AI Travel Planner project successfully achieved its goal of providing a dynamic, AI-driven travel planning platform that offers personalized itinerary generation based on key user inputs such as age, travel budget, interests, and number of travel days. The final product is a fully functional, user-friendly, and scalable web-based application that has been tested across multiple user scenarios to evaluate its accuracy, performance, and usability.

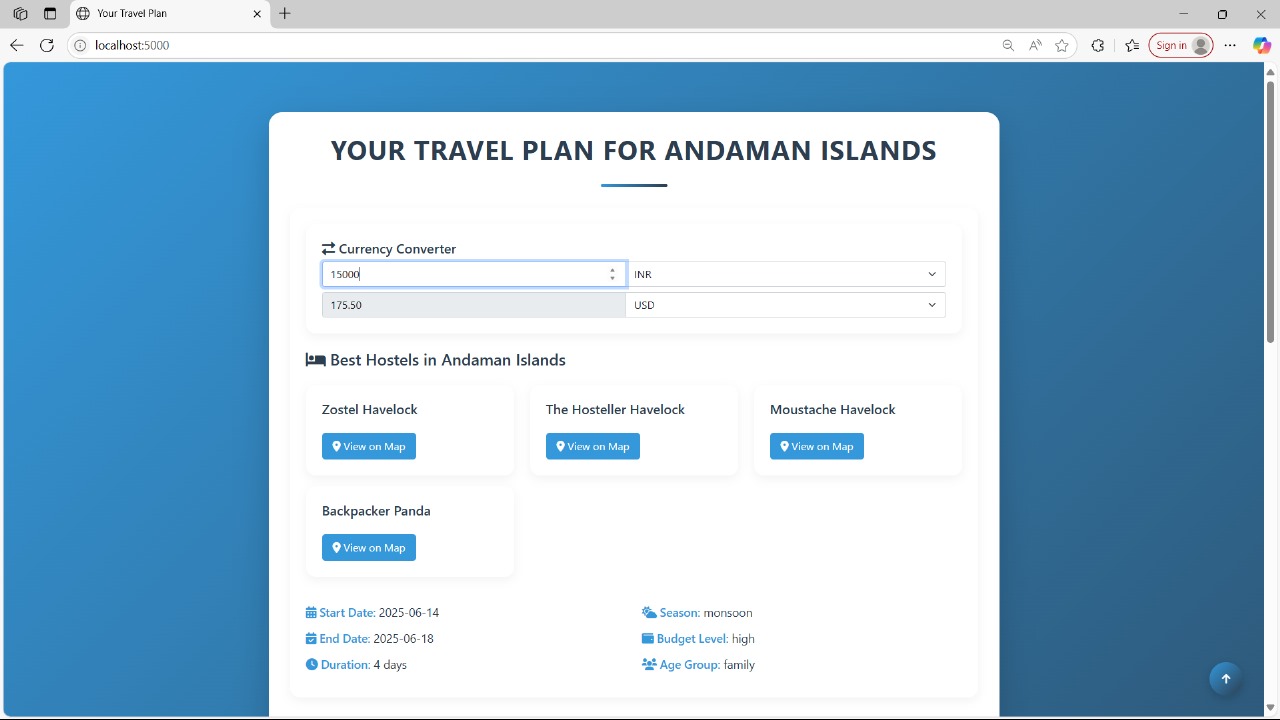
During testing, the system demonstrated the ability to generate relevant and personalized travel destinations and activities in real-time. Each user input was correctly interpreted and classified into categories, leading to high-quality recommendations that matched the user's expectations. The rule-based logic engine worked seamlessly, selecting appropriate activities for each part of the day while taking into account distance, category fit, budget, and age group preferences. This resulted in full-day itineraries that users could rely on without additional research.

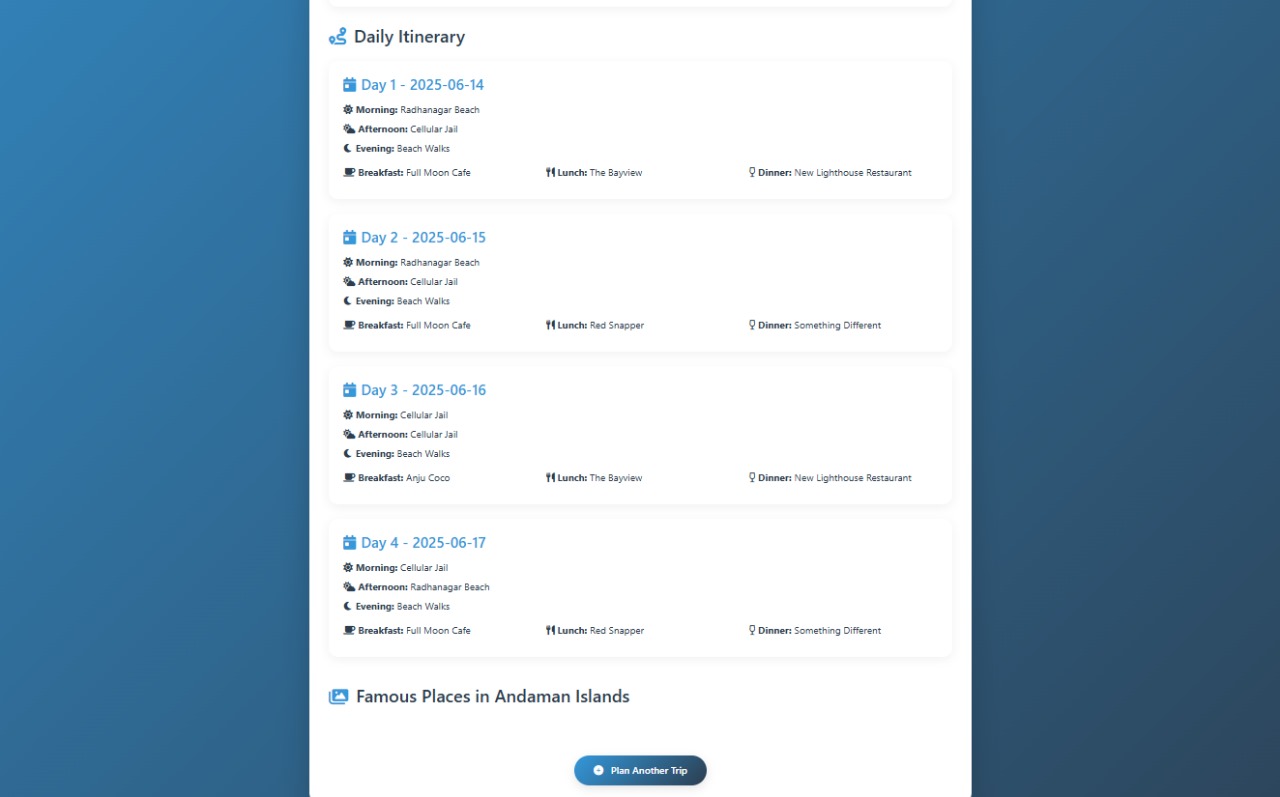
The backend engine was effective in filtering and organizing activity data from structured JSON datasets. Real-time APIs such as OpenWeatherMap and Google Maps were successfully integrated and performed consistently across different environments. The application adjusted itinerary recommendations based on real-time weather conditions, thus improving practicality and enhancing user experience. Activities that were inappropriate for specific weather conditions were automatically excluded and replaced, which made the system more intelligent and context-aware.

Performance metrics also showed promising results. The response time for generating itineraries remained below 3 seconds even when multiple APIs were queried simultaneously. The PDF export and .ICS calendar integration worked without error, giving users access to offline and sharable formats. Mobile responsiveness was tested on multiple screen sizes and resolutions, and the system displayed adaptive layout behavior without loss of readability or functionality.









**PROJECT DESCRIPTION**

In order to realize its full potential as a comprehensive travel solution, the AI Travel Planner has been conceptually expanded to include a broad range of real-world utilities and digital ecosystem integrations. For instance, a roadmap for multi-language support has been proposed to cater to international users by offering destination names, recommendations, and user prompts in regional languages. Localization modules also take into account cultural norms and calendar events specific to regions, allowing travelers to plan around festivals, peak seasons, or local holidays.

Another potential area of growth lies in travel community engagement. Planned features include a forum or social feed where travelers can post reviews, photos, and suggestions based on their AI-generated itineraries. These contributions can, in turn, feed into the recommendation engine, enabling a hybrid of AI and crowdsourced travel intelligence. By incorporating sentiment analysis from user-generated content, the system can enhance its contextual accuracy and recommendation confidence levels.

The AI Travel Planner also presents opportunities for strategic partnerships. Collaborations with tourism boards, local experience providers, or eco-tourism platforms can help populate the system with rich, verified activity data. API integrations with rail and airline networks would allow real-time booking assistance, while syncing with ride-sharing platforms can streamline local transport within the itinerary.

A sustainability-focused version of the planner is also under ideation. It would allow users to set a “green travel score” target, encouraging them to select eco-conscious accommodations, use public transport, and avoid high-emission routes. Educational prompts and carbon footprint visualizations could make the planner both functional and educational.

For developers and researchers, a public API access model is proposed, allowing third-party apps or institutions to use anonymized itinerary logic for educational tools or tourism analysis. As AI and data regulations evolve, the platform is also being designed with compliance in mind, including options for data export, consent control, and anonymized feedback loops.

As part of its long-term evolution, the AI Travel Planner may be transformed into a fully adaptive assistant capable of integrating voice inputs, wearable device syncing, real-time alerts, and dynamic re-planning based on live conditions such as flight delays or weather emergencies. In doing so, the platform will not only serve as a planning interface but also as a companion throughout the travel lifecycle—from inspiration and planning to execution and post-trip review.

By aligning modern software engineering practices with AI advancements and human-centered design, the AI Travel Planner stands out as a next-generation travel ecosystem. It represents not just a tool for convenience, but a transformative agent capable of reshaping how individuals perceive, plan, and pursue travel worldwide.

**CONCLUSION**

The AI Travel Planner project stands as a testament to how artificial intelligence and modern web technologies can be combined to revolutionize an everyday activity such as travel planning. Through the intelligent processing of user data and integration with real-time APIs, the system delivers personalized, day-wise travel itineraries tailored to individual preferences, age groups, budgets, and dynamic conditions such as weather or travel distances.

By designing a rule-based, modular architecture that separates user input, classification logic, API integration, and itinerary rendering, the platform has demonstrated its scalability, robustness, and adaptability. It successfully bridges the gap between user intent and travel logistics, eliminating the need for users to sift through multiple websites or spend hours constructing travel plans manually. Instead, it provides a seamless, intuitive, and visually informative experience with options to export or share results in accessible formats.

Furthermore, the system has laid the groundwork for future enhancements including collaborative itinerary planning, real-time booking features, machine learning-based adaptive models, multilingual interfaces, and sustainability filters. Its potential extends beyond a planning tool to becoming a comprehensive, AI-powered travel assistant that evolves with user behavior, regional trends, and global travel patterns.

The successful implementation and testing of this project confirm the viability of the approach and its relevance in today’s tech-driven tourism industry. It exemplifies how thoughtful software design, when centered around user experience and smart automation, can solve complex problems in an efficient, engaging, and intelligent manner. The AI Travel Planner not only simplifies the journey—it enhances it, making travel more informed, efficient, and enjoyable for users around the world.

**REFERENCES**

1. Liu, Y., Zhang, H., & Wang, X. (2020). A Personalized Travel Recommendation System Based on User Interest and Location. Journal of Intelligent Information Systems.
2. Patel, R. B., & Sharma, M. (2019). Smart Itinerary Recommendation System Using AI. International Journal of Computer Applications.
3. Mehta, A., & Kulkarni, S. (2021). AI-Enhanced Travel Assistant with Contextual Adaptation. Lecture Notes in Networks and Systems (LNNS), Springer.
4. Li, Q., Wang, Z., & Chen, Y. (2020). An Intelligent Tourist Guide System Based on Machine Learning and User Profiling. International Journal of Tourism Research.
5. Tan, Y., & Tan, R. (2022). Adaptive Itinerary Generation Using Context-Aware Recommender Systems. ACM Transactions on Intelligent Systems and Technology.
6. Kumar, A., & Roy, S. (2021). Smart Travel Planning Framework Leveraging Rule-Based AI and Dynamic Data. Journal of Web Engineering.
7. Zhang, P., Li, X., & Zhang, C. (2018). Real-Time Location-Based Travel Recommendations Using User Clustering Techniques. IEEE Access.
8. Nguyen, H. T., & Tran, M. K. (2022). A Cloud-Based Approach to Personalized Travel Planning. Journal of Computing and Sustainable Systems.
9. Singh, R., & Gupta, M. (2019). Data-Driven Travel Assistance: Integration of APIs and Modular Backend Logic. Advances in Computer Science and Engineering.
10. Sharma, K., & Desai, P. (2020). Enhancing Tourist Experience through AI-Powered Itinerary Systems. Journal of Computer Applications in Tourism.

**PROJECT FILES**

[**https://drive.google.com/file/d/1eAAzRDMVPoXTiqmehnIHgjsMY-\_b1A0O/view?usp=drive\_link**](https://drive.google.com/file/d/1eAAzRDMVPoXTiqmehnIHgjsMY-_b1A0O/view?usp=drive_link)

**PROGRAM**

**# HTML CODE**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>AI Travel Planner</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css">

    <style>

        body {

            margin: 0;

            padding: 0;

            min-height: 100vh;

            background: url('https://images.unsplash.com/photo-1476514525535-07fb3b4ae5f1?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center fixed;

            background-size: cover;

            position: relative;

        }

        body::before {

            content: '';

            position: absolute;

            top: 0;

            left: 0;

            right: 0;

            bottom: 0;

            background: rgba(0, 0, 0, 0.4);

            z-index: 1;

        }

        .container {

            position: relative;

            z-index: 2;

            max-width: 900px;

            padding: 2rem;

        }

        .card {

            background: rgba(255, 255, 255, 0.9);

            backdrop-filter: blur(10px);

            border-radius: 20px;

            box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);

            border: 1px solid rgba(255, 255, 255, 0.2);

            transition: transform 0.3s ease;

        }

        .card:hover {

            transform: translateY(-5px);

        }

        .form-label {

            font-weight: 600;

            color: #2c3e50;

        }

        .form-control {

            border-radius: 10px;

            border: 1px solid rgba(0, 0, 0, 0.1);

            padding: 0.8rem;

            transition: all 0.3s ease;

        }

        .form-control:focus {

            box-shadow: 0 0 0 3px rgba(102, 126, 234, 0.25);

        }

        .btn-primary {

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            border: none;

            padding: 1rem 2.5rem;

            border-radius: 50px;

            font-weight: 600;

            letter-spacing: 0.5px;

            transition: all 0.3s ease;

        }

        .btn-primary:hover {

            background: linear-gradient(135deg, #764ba2 0%, #667eea 100%);

            transform: translateY(-2px);

            box-shadow: 0 5px 15px rgba(102, 126, 234, 0.4);

        }

        .interest-tag {

            display: inline-flex;

            align-items: center;

            padding: 0.6rem 1.2rem;

            margin: 0.3rem;

            background-color: rgba(255, 255, 255, 0.9);

            border-radius: 50px;

            cursor: pointer;

            transition: all 0.3s ease;

            border: 1px solid rgba(0, 0, 0, 0.1);

        }

        .interest-tag i {

            margin-right: 8px;

            font-size: 1.1rem;

        }

        .interest-tag.selected {

            background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

            color: white;

            transform: scale(1.05);

        }

        .weather-widget {

            background: rgba(255, 255, 255, 0.9);

            border-radius: 15px;

            padding: 1rem;

            margin-bottom: 1rem;

        }

        .currency-converter {

            background: rgba(255, 255, 255, 0.9);

            border-radius: 15px;

            padding: 1rem;

            margin-bottom: 1rem;

        }

        .loading {

            display: none;

            position: fixed;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background: rgba(0, 0, 0, 0.7);

            z-index: 1000;

            justify-content: center;

            align-items: center;

        }

        .loading-spinner {

            width: 50px;

            height: 50px;

            border: 5px solid #f3f3f3;

            border-top: 5px solid #667eea;

            border-radius: 50%;

            animation: spin 1s linear infinite;

        }

        @keyframes spin {

            0% { transform: rotate(0deg); }

            100% { transform: rotate(360deg); }

        }

        .page-title {

            color: white;

            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.3);

            font-size: 2.5rem;

            margin-bottom: 2rem;

        }

    </style>

</head>

<body>

    <div class="loading">

        <div class="loading-spinner"></div>

    </div>

    <div class="container py-5">

        <h1 class="text-center page-title">AI Travel Planner</h1>

        <div class="card p-4">

            {% if error %}

            <div class="alert alert-danger">{{ error }}</div>

            {% endif %}

            <div class="row">

                <div class="col-md-6">

                    <div class="weather-widget">

                        <h5><i class="fas fa-cloud-sun"></i> Weather Forecast</h5>

                        <div id="weather-info">Loading weather data...</div>

                    </div>

                </div>

                <div class="col-md-6">

                    <div class="currency-converter">

                        <h5><i class="fas fa-exchange-alt"></i> Currency Converter</h5>

                        <div class="input-group mb-2">

                            <input type="number" class="form-control" id="amount" placeholder="Amount">

                            <select class="form-select" id="from-currency">

                                <option value="INR">INR</option>

                                <option value="USD">USD</option>

                                <option value="EUR">EUR</option>

                            </select>

                        </div>

                        <div class="input-group">

                            <input type="number" class="form-control" id="converted-amount" readonly>

                            <select class="form-select" id="to-currency">

                                <option value="USD">USD</option>

                                <option value="INR">INR</option>

                                <option value="EUR">EUR</option>

                            </select>

                        </div>

                    </div>

                </div>

            </div>

            <form method="POST" id="travel-form">

                <div class="row">

                    <div class="col-md-6">

                        <div class="mb-3">

                            <label for="age" class="form-label"><i class="fas fa-user"></i> Age</label>

                            <input type="number" class="form-control" id="age" name="age" required min="1" max="120">

                        </div>

                    </div>

                    <div class="col-md-6">

                        <div class="mb-3">

                            <label for="budget" class="form-label"><i class="fas fa-wallet"></i> Budget (in INR)</label>

                            <input type="number" class="form-control" id="budget" name="budget" required min="1000">

                        </div>

                    </div>

                </div>

                <div class="row">

                    <div class="col-md-6">

                        <div class="mb-3">

                            <label for="start\_date" class="form-label"><i class="fas fa-calendar-alt"></i> Start Date</label>

                            <input type="date" class="form-control" id="start\_date" name="start\_date" required>

                        </div>

                    </div>

                    <div class="col-md-6">

                        <div class="mb-3">

                            <label for="end\_date" class="form-label"><i class="fas fa-calendar-alt"></i> End Date</label>

                            <input type="date" class="form-control" id="end\_date" name="end\_date" required>

                        </div>

                    </div>

                </div>

                <div class="mb-4">

                    <label class="form-label"><i class="fas fa-heart"></i> Interests (Select multiple)</label>

                    <div class="interests-container">

                        <span class="interest-tag" data-value="history"><i class="fas fa-landmark"></i> History</span>

                        <span class="interest-tag" data-value="culture"><i class="fas fa-theater-masks"></i> Culture</span>

                        <span class="interest-tag" data-value="nature"><i class="fas fa-mountain"></i> Nature</span>

                        <span class="interest-tag" data-value="beach"><i class="fas fa-umbrella-beach"></i> Beach</span>

                        <span class="interest-tag" data-value="adventure"><i class="fas fa-hiking"></i> Adventure</span>

                        <span class="interest-tag" data-value="food"><i class="fas fa-utensils"></i> Food</span>

                        <span class="interest-tag" data-value="shopping"><i class="fas fa-shopping-bag"></i> Shopping</span>

                        <span class="interest-tag" data-value="photography"><i class="fas fa-camera"></i> Photography</span>

                        <span class="interest-tag" data-value="spiritual"><i class="fas fa-om"></i> Spiritual</span>

                        <span class="interest-tag" data-value="architecture"><i class="fas fa-building"></i> Architecture</span>

                    </div>

                    <input type="hidden" name="interests" id="selected\_interests">

                </div>

                <div class="text-center">

                    <button type="submit" class="btn btn-primary btn-lg">

                        <i class="fas fa-plane-departure"></i> Plan My Trip

                    </button>

                </div>

            </form>

        </div>

    </div>

    <script>

        // Set minimum date to today

        const today = new Date().toISOString().split('T')[0];

        document.getElementById('start\_date').min = today;

        document.getElementById('end\_date').min = today;

        // Handle date validation

        document.getElementById('start\_date').addEventListener('change', function() {

            document.getElementById('end\_date').min = this.value;

        });

        // Handle interests selection

        const interestTags = document.querySelectorAll('.interest-tag');

        const selectedInterestsInput = document.getElementById('selected\_interests');

        let selectedInterests = [];

        interestTags.forEach(tag => {

            tag.addEventListener('click', function() {

                const value = this.dataset.value;

                if (this.classList.contains('selected')) {

                    this.classList.remove('selected');

                    selectedInterests = selectedInterests.filter(i => i !== value);

                } else {

                    this.classList.add('selected');

                    selectedInterests.push(value);

                }

                selectedInterestsInput.value = selectedInterests.join(',');

            });

        });

        // Show loading animation on form submit

        document.getElementById('travel-form').addEventListener('submit', function() {

            document.querySelector('.loading').style.display = 'flex';

        });

        // Currency Converter

        const amountInput = document.getElementById('amount');

        const fromCurrency = document.getElementById('from-currency');

        const toCurrency = document.getElementById('to-currency');

        const convertedAmount = document.getElementById('converted-amount');

        async function convertCurrency() {

            const amount = amountInput.value;

            const from = fromCurrency.value;

            const to = toCurrency.value;

            if (amount && from && to) {

                try {

                    const response = await fetch(`https://api.exchangerate-api.com/v4/latest/${from}`);

                    const data = await response.json();

                    const rate = data.rates[to];

                    convertedAmount.value = (amount \* rate).toFixed(2);

                } catch (error) {

                    console.error('Error converting currency:', error);

                }

            }

        }

        amountInput.addEventListener('input', convertCurrency);

        fromCurrency.addEventListener('change', convertCurrency);

        toCurrency.addEventListener('change', convertCurrency);

        // Weather Widget

        async function getWeather() {

            try {

                const response = await fetch('https://api.openweathermap.org/data/2.5/weather?q=Mumbai&appid=YOUR\_API\_KEY&units=metric');

                const data = await response.json();

                const weatherInfo = document.getElementById('weather-info');

                weatherInfo.innerHTML = `

                    <p>Temperature: ${Math.round(data.main.temp)}°C</p>

                    <p>Weather: ${data.weather[0].description}</p>

                `;

            } catch (error) {

                console.error('Error fetching weather:', error);

                document.getElementById('weather-info').innerHTML = 'Weather data unavailable';

            }

        }

        // Note: You'll need to replace 'YOUR\_API\_KEY' with an actual OpenWeatherMap API key

        // getWeather();

    </script>

    <script>

        document.addEventListener('DOMContentLoaded', function() {

            const defaultCity = 'Delhi'; // Default city for weather

            fetchWeather(defaultCity);

        });

        function fetchWeather(city) {

            const weatherInfoDiv = document.getElementById('weather-info');

            weatherInfoDiv.innerHTML = 'Loading weather data...';

            fetch(`/weather?city=${city}`)

                .then(response => response.json())

                .then(data => {

                    if (data.error) {

                        weatherInfoDiv.innerHTML = `<p>Could not fetch weather data.</p>`;

                        return;

                    }

                    const weatherHtml = `

                        <div class="d-flex justify-content-between align-items-center">

                            <div>

                                <strong>${data.name}</strong>

                                <p class="mb-0">${data.weather[0].main}</p>

                            </div>

                            <div class="text-end">

                                <img src="http://openweathermap.org/img/wn/${data.weather[0].icon}.png" alt="Weather icon">

                                <p class="mb-0 fs-4">${data.main.temp}°C</p>

                            </div>

                        </div>

                    `;

                    weatherInfoDiv.innerHTML = weatherHtml;

                })

                .catch(error => {

                    console.error('Error fetching weather:', error);

                    weatherInfoDiv.innerHTML = `<p>Could not fetch weather data.</p>`;

                });

        }

    </script>

</body>

</html>

#**BACKEND CODE**

from flask import Flask, request, render\_template, jsonify

import requests

import json

import random

from threading import Thread

import webbrowser

import time

from werkzeug.serving import make\_server

from datetime import datetime, timedelta

app = Flask(\_\_name\_\_)

# Load destination data from JSON file

with open('destinations.json') as f:

    city\_data = json.load(f)

def determine\_age\_group(age):

    if age < 25:

        return 'youth'

    elif 25 <= age <= 45:

        return 'family'

    else:

        return 'senior'

def determine\_budget\_level(budget):

    if budget < 5000:

        return 'low'

    elif budget < 15000:

        return 'medium'

    else:

        return 'high'

def get\_season(start\_date):

    month = start\_date.month

    if month in [3, 4, 5]:

        return 'summer'

    elif month in [6, 7, 8, 9]:

        return 'monsoon'

    elif month in [10, 11]:

        return 'autumn'

    else:

        return 'winter'

def match\_interests(interests, city\_info):

    score = 0

    for interest in interests:

        if interest in city\_info.get('interests', []):

            score += 1

    return score

@app.route("/", methods=["GET", "POST"])

def index():

    if request.method == "POST":

        try:

            age = int(request.form['age'])

            budget = int(request.form['budget'])

            start\_date = datetime.strptime(request.form['start\_date'], '%Y-%m-%d')

            end\_date = datetime.strptime(request.form['end\_date'], '%Y-%m-%d')

            interests = request.form.getlist('interests')

            age\_group = determine\_age\_group(age)

            budget\_level = determine\_budget\_level(budget)

            season = get\_season(start\_date)

            duration = (end\_date - start\_date).days

            # Match by age, budget, and season

            matched = []

            for city, info in city\_data.items():

                if (info['age\_group'] == age\_group and

                    info['budget'] == budget\_level and

                    season in info.get('best\_seasons', ['summer', 'winter', 'monsoon', 'autumn'])):

                    interest\_score = match\_interests(interests, info)

                    matched.append((city, interest\_score))

            # Sort by interest match score

            matched.sort(key=lambda x: x[1], reverse=True)

            matched\_cities = [city for city, \_ in matched]

            # Fallbacks

            if not matched\_cities:

                matched\_cities = [city for city, info in city\_data.items()

                                if info['budget'] == budget\_level]

            if not matched\_cities:

                matched\_cities = list(city\_data.keys())

            selected\_city = random.choice(matched\_cities[:3])  # Choose from top 3 matches

            city\_info = city\_data[selected\_city]

            # Generate itinerary based on duration

            itinerary = []

            if duration > 0:

                for i in range(duration):

                    day\_plan = {

                        "day": i + 1,

                        "date": (start\_date + timedelta(days=i)).strftime('%Y-%m-%d'),

                        "morning": random.choice(city\_info.get('attractions\_day', [])),

                        "afternoon": random.choice(city\_info.get('attractions\_day', [])),

                        "evening": random.choice(city\_info.get('attractions\_night', [])),

                        "breakfast": random.choice(city\_info.get('restaurants\_breakfast', [])),

                        "lunch": random.choice(city\_info.get('restaurants\_lunch', [])),

                        "dinner": random.choice(city\_info.get('restaurants\_dinner', []))

                    }

                    itinerary.append(day\_plan)

            plan = {

                "city": selected\_city,

                "start\_date": start\_date.strftime('%Y-%m-%d'),

                "end\_date": end\_date.strftime('%Y-%m-%d'),

                "duration": duration,

                "season": season,

                "itinerary": itinerary,

                \*\*city\_info

            }

            return render\_template("result.html", age=age, budget=budget, plan=plan)

        except Exception as e:

            return render\_template("index.html", error="Invalid input!")

    return render\_template("index.html")

WEATHER\_API\_KEY = "YOUR\_API\_KEY"  # Replace with your OpenWeatherMap API key

WEATHER\_API\_URL = "http://api.openweathermap.org/data/2.5/weather"

@app.route("/weather")

def get\_weather():

    city = request.args.get('city')

    if not city:

        return jsonify({"error": "City not provided"}), 400

    params = {

        "q": city,

        "appid": WEATHER\_API\_KEY,

        "units": "metric"

    }

    try:

        response = requests.get(WEATHER\_API\_URL, params=params)

        response.raise\_for\_status()  # Raise an exception for bad status codes

        weather\_data = response.json()

        return jsonify(weather\_data)

    except requests.exceptions.RequestException as e:

        return jsonify({"error": str(e)}), 500

@app.route('/get\_destinations')

def get\_destinations():

    try:

        with open('destinations.json', 'r') as f:

            destinations = json.load(f)

        return jsonify(destinations)

    except Exception as e:

        return jsonify({"error": str(e)}), 500

# Auto-launch browser

class ServerThread(Thread):

    def \_\_init\_\_(self, app):

        Thread.\_\_init\_\_(self)

        self.server = make\_server('localhost', 5000, app)

        self.ctx = app.app\_context()

        self.ctx.push()

    def run(self):

        self.server.serve\_forever()

    def shutdown(self):

        self.server.shutdown()

if \_\_name\_\_ == "\_\_main\_\_":

    server = ServerThread(app)

    server.start()

    time.sleep(1)

    webbrowser.open("http://localhost:5000")

**#JSON CODE**

{

    "Hampi": {

        "budget": "low",

        "age\_group": "youth",

        "best\_seasons": ["winter", "autumn"],

        "interests": ["history", "architecture", "photography", "ruins", "temples"],

        "hotels": [

            "Hampi Heritage Resort",

            "Archana Guest House",

            "Shanbhag Towers",

            "Hotel Malligi",

            "Whispering Rocks",

            "Goan Corner",

            "Hampi's Boulders",

            "Shanthi Guest House",

            "Gopi Guest House",

            "Hampi's Boulders"

        ],

        "attractions\_day": [

            "Virupaksha Temple",

            "Vittala Temple",

            "Lotus Mahal",

            "Elephant Stables",

            "Queen's Bath",

            "Royal Enclosure",

            "Hazara Rama Temple"

        ],

        "attractions\_night": [

            "Riverside Sunset Point",

            "Local Night Bazaar",

            "Sunset at Hemakuta Hill",

            "Cultural Dance Show"

        ],

        "restaurants\_breakfast": [

            "Hampi Roof Restaurant",

            "Gopi Guest House",

            "Mango Tree",

            "Laughing Buddha Cafe"

        ],

        "restaurants\_lunch": [

            "Mango Tree",

            "Laughing Buddha Cafe",

            "Gopi Guest House",

            "Hampi Roof Restaurant"

        ],

        "restaurants\_dinner": [

            "Funky Monkey",

            "Moonlight Restaurant",

            "Mango Tree",

            "Laughing Buddha Cafe"

        ],

        "local\_food": [

            "Bisi Bele Bath",

            "Mysore Pak",

            "Dosa",

            "Idli",

            "Filter Coffee"

        ],

        "famous\_places": [

            {

                "name": "Virupaksha Temple",

                "description": "Ancient temple dedicated to Lord Shiva, known for its magnificent architecture"

            },

            {

                "name": "Vittala Temple",

                "description": "Famous for its musical pillars and stone chariot"

            },

            {

                "name": "Lotus Mahal",

                "description": "Beautiful palace with lotus-like architecture"

            }

        ]

    },

    "Goa": {

        "budget": "medium",

        "age\_group": "youth",

        "best\_seasons": ["winter", "autumn"],

        "interests": ["beach", "nightlife", "water sports", "food", "shopping"],

        "hotels": [

            "Taj Vivanta",

            "Leela Goa",

            "Zostel Goa",

            "Fisherman's Wharf Inn",

            "Marina Bay Beach Resort",

            "The Hosteller Goa",

            "Goa Backpackers",

            "Pappi Chulo",

            "Jungle Hostel",

            "Roadhouse Hostels"

        ],

        "attractions\_day": [

            "Calangute Beach",

            "Fort Aguada",

            "Dudhsagar Falls",

            "Spice Plantations",

            "Old Goa Churches",

            "Chapora Fort",

            "Anjuna Flea Market"

        ],

        "attractions\_night": [

            "Baga Beach Clubs",

            "Night Cruise",

            "Tito's Lane",

            "Cavala",

            "SinQ Nightclub"

        ],

        "restaurants\_breakfast": [

            "Cafe Chocolatti",

            "Eva Cafe",

            "Artjuna",

            "Infantaria"

        ],

        "restaurants\_lunch": [

            "Thalassa",

            "Pousada by the Beach",

            "Fisherman's Wharf",

            "Viva Panjim"

        ],

        "restaurants\_dinner": [

            "Fisherman's Wharf",

            "Fat Fish",

            "Pousada by the Beach",

            "Thalassa"

        ],

        "local\_food": [

            "Fish Curry Rice",

            "Prawn Balchao",

            "Bebinca",

            "Sorpotel",

            "Vindaloo"

        ],

        "famous\_places": [

            {

                "name": "Calangute Beach",

                "description": "Known as the Queen of Beaches, perfect for water sports"

            },

            {

                "name": "Dudhsagar Falls",

                "description": "Majestic four-tiered waterfall surrounded by lush greenery"

            },

            {

                "name": "Old Goa Churches",

                "description": "UNESCO World Heritage site with stunning Portuguese architecture"

            }

        ]

    }”

        "attractions\_day": [

            "Amber Fort",

            "Hawa Mahal"

        ],

        "attractions\_night": [

            "Chokhi Dhani",

            "Jal Mahal View"

        ],

        "restaurants\_breakfast": [

            "Tapri Central",

            "Anokhi Cafe"

        ],

        "restaurants\_lunch": [

            "Rawat Mishthan Bhandar",

            "Peacock Rooftop"

        ],

        "restaurants\_dinner": [

            "Suvarna Mahal",

            "RJ 14"

        ]

    },

    "Ooty": {

        "budget": "medium",

        "age\_group": "family",

        "hotels": [

            "Savoy Ooty",

            "Gem Park",

            "Hotel Lakeview",

            "Sinclairs Retreat",

            "Hotel Darshan",

            "Zostel Ooty",

            "The Hosteller Ooty",

            "GoStops Ooty",

            "Moustache Ooty",

            "Backpacker Panda"

        ],

        "attractions\_day": [

            "Botanical Garden",

            "Doddabetta Peak"

        ],

        "attractions\_night": [

            "Ooty Lake Boat House",

            "Tea Factory Night Tour"

        ],

        "restaurants\_breakfast": [

            "Willy's Coffee Pub",

            "Cafe Blue Hills"

        ],

        "restaurants\_lunch": [

            "Earl's Secret",

            "Quality Restaurant"

        ],

        "restaurants\_dinner": [

            "Hyderabad Biryani House",

            "Shinkows"

        ]

    },

    "Munnar": {

        "budget": "high",

        "age\_group": "senior",

        "hotels": [

            "Parakkat Nature",

            "Tea County",

            "Amber Dale",

            "The Leaf Munnar",

            "Chandy's Windy Woods",

            "Zostel Munnar",

            "The Hosteller Munnar",

            "GoStops Munnar",

            "Moustache Munnar",

            "Backpacker Panda"

        ],

        "attractions\_day": [

            "Tea Plantations",

            "Eravikulam National Park"

        ],

        "attractions\_night": [

            "Campfire",

            "Local Cultural Show"

        ],

        "restaurants\_breakfast": [

            "Rapsy Restaurant",

            "Saravana Bhavan"

        ],

        "restaurants\_lunch": [

            "Tea Tales Cafe",

            "Rochas Restaurant"

        ],

        "restaurants\_dinner": [

            "Hotel Sri Nivas",

            "Eastend Restaurant"

        ]

    },

    "Kodaikanal": {

        "budget": "medium",

        "age\_group": "senior",

        "hotels": [

            "The Carlton",

            "Kodai Resort",

            "Great Trails",

            "Villa Retreat",

            "Sterling Kodai Lake",

            "Zostel Kodaikanal",

            "The Hosteller Kodaikanal",

            "GoStops Kodaikanal",

            "Moustache Kodaikanal",

            "Backpacker Panda"

        ],

        "attractions\_day": [

            "Coaker's Walk",

            "Pillar Rocks"

        ],

        "attractions\_night": [

            "Bryant Park",

            "Lake View"

        ],

        "restaurants\_breakfast": [

            "Pastry Corner",

            "Cloud Street"

        ],

        "restaurants\_lunch": [

            "Astoria Veg",

            "The Royal Tibet"

        ],

        "restaurants\_dinner": [

            "Tava Vegetarian",

            "Muncheez"

        ]

    },

    "Khajuraho": {

        "budget": "medium",

        "age\_group": "senior",

        "hotels": [

            "Radisson Jass",

            "The Lalit Temple View",

            "Hotel Isabel Palace",

            "Ramada Khajuraho",

            "Zen Hotel",

            "Zostel Khajuraho",

            "The Hosteller Khajuraho",

            "GoStops Khajuraho",

            "Moustache Khajuraho",

            "Backpacker Panda"

        ],

        "attractions\_day": [

            "Western Group of Temples",

            "Raneh Falls"

        ],

        "attractions\_night": [

            "Sound & Light Show",

            "Local Dance Show"

        ],

        "restaurants\_breakfast": [

            "Raja Cafe",

            "Lassi Corner"

        ],

        "restaurants\_lunch": [

            "Mediterraneo",

            "Paradise Restaurant"

        ],

        "restaurants\_dinner": [

            "Blue Sky Restaurant",

            "Indiana Restaurant"

        ]

    }

}