IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

THE PERSONALISED TRAVEL PLANNING SYSTEM WITH AI ASSISTANT

Mugundhan AV[1], Dr. K. Santhi[2]

Student, Department of Information Technology, Dr. N.G.P Arts and Science College, Coimbatore.[1]

Professor, Department of Information Technology, Dr. N.G.P Arts and Science College, Coimbatore[2]

Abstract: The rapid growth of the tourism industry has led to an overwhelming amount of travel information, making itinerary planning a complex and time-consuming task for travellers. This paper presents a Personalised Travel Planning System with an AI Assistant, a web-based application that leverages artificial intelligence to provide customized travel recommendations, intelligent itinerary generation, destination insights, flight suggestions, and real-time travel tips. The system is built using the Flask framework, integrates with the OpenRouter API to access large language models (GPT-3.5-turbo), and incorporates user authentication with email notifications. The proposed solution simplifies travel planning by offering an interactive conversational agent and automated planning tools, thereby enhancing user experience and decision-making. Experimental results demonstrate the system's effectiveness in generating relevant, context-aware travel plans and advice. Future enhancements include integration with live booking APIs, multi-language support, and a mobile application.

Keywords: Travel planning, AI assistant, personalized itinerary, chatbot, destination insights, OpenRouter, Flask, recommendation system.

I. INTRODUCTION

Travel planning traditionally involves researching destinations, comparing flight options, scheduling activities, and managing budgets – a process that can be overwhelming due to the sheer volume of online information. Recent advances in artificial intelligence (AI), particularly in natural language processing (NLP) and large language models (LLMs), offer new opportunities to automate and personalize this process. AI-powered travel assistants can understand user preferences, answer queries, and generate tailored itineraries in real time, significantly reducing planning effort.

This paper introduces a **Personalised Travel Planning System with an AI Assistant**, a web application that combines a user-friendly interface with the capabilities of a state-of-the-art language model accessed via the OpenRouter API. The system provides five core functionalities: an AI chat for answering travel-related questions, an itinerary planner that generates day-by-day plans based on user inputs, a destination insights module delivering curated information about attractions and culture, a flight suggestions tool that offers flight options and redirects to popular booking platforms, and a travel tips module providing expert advice. The application also includes secure user authentication with email notification on login, storing user credentials in an SQLite database. The frontend is built with HTML, CSS, and Bootstrap, ensuring a responsive and engaging user experience.

The remainder of this paper is organized as follows: Section 2 reviews related work in travel recommendation systems and AI assistants. Section 3 defines the problem statement. Section 4 describes the proposed system architecture and modules. Section 5 details the system workflow. Section 6 presents the implementation and results, including user interface screenshots. Section 7 concludes the paper and outlines future research directions.

II. LITERATURE SURVEY

Several studies and commercial systems have addressed travel planning using recommendation techniques and AI. Traditional travel platforms such as TripAdvisor, Expedia, and Booking.com aggregate user reviews and booking options but lack personalized itinerary generation; they require manual filtering and do not provide interactive conversational support. Research into AI chatbots in tourism has explored systems like TravelBot [1], which used rule-based dialogue to answer FAQs, while more recent systems employ transformer-based models for natural conversation. However, many are limited to specific domains or require extensive training data.

Recommender systems for tourism have applied collaborative filtering and content-based methods to recommend destinations, hotels, and activities [2]. Hybrid approaches combine user profiles with contextual information, but they

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

often struggle with cold-start problems and dynamic user preferences. The emergence of large language models (e.g., GPT-3, GPT-4) has enabled zero-shot generation of travel itineraries and tips. OpenRouter provides a unified API to multiple LLMs, simplifying integration. Our work leverages this technology to deliver real-time, personalized travel content without the need for custom model training. Existing AI travel assistants like Google Trips (discontinued) and TripIt offer itinerary aggregation but lack generative AI, whereas newer startups such as Roam Around use AI to create daily plans but often focus solely on itineraries. Our system combines multiple travel-planning features in one coherent application, addressing the growing need for intelligent, all-in-one travel planning tools.

III. PROBLEM STATEMENT

Modern travellers face several challenges when planning trips. Information overload from numerous websites, reviews, and booking platforms makes it difficult to filter relevant information. Generic travel guides do not account for individual interests, budget, or time constraints, leading to a lack of personalization. Manual research and itinerary assembly can take hours or days, resulting in high time consumption. Users often switch between multiple apps such as maps, flight search engines, and review sites to plan a single trip, creating a fragmented experience. Moreover, most existing systems provide static suggestions that do not adapt to real-time changes or conversational context. There is a clear demand for a unified, AI-driven platform that can understand natural language queries, generate personalized travel plans, and offer actionable insights – all within a single interface. This project aims to develop such a system, demonstrating the feasibility and benefits of integrating LLMs into travel planning.

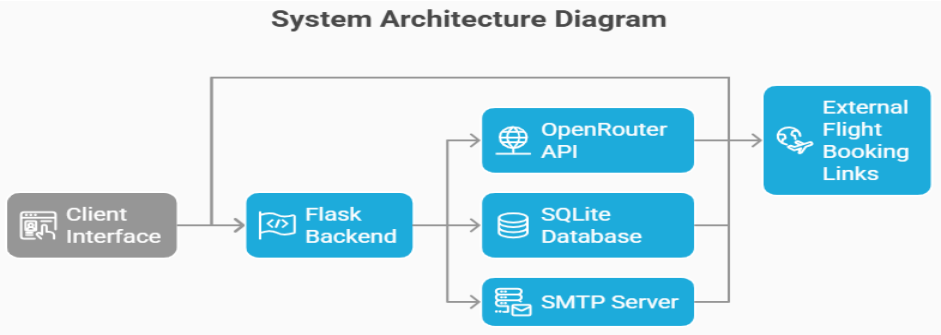
IV. PROPOSED SYSTEM

4.1 OVERVIEW

The proposed system is a web application built with Python Flask that provides a range of travel planning services powered by an AI language model. Users can register, log in, and then access five main modules: Chat, Itinerary, Destination Insights, Flight Suggestions, and Travel Tips. The AI component generates responses dynamically based on user input and context.

4.2 SYSTEM ARCHITECTURE

Figure 1 illustrates the high-level architecture of the system. The client browser communicates with the Flask backend, which in turn interacts with the OpenRouter API for AI responses, the SQLite database for user data, and an SMTP server for email notifications. The frontend templates are rendered by Flask and delivered to the client.



[Figure 1: System Architecture Diagram showing client, Flask backend, OpenRouter API, SQLite database, SMTP server, and external flight booking links.]

The frontend comprises HTML templates styled with CSS and Bootstrap, served by Flask, and includes JavaScript for dynamic behaviour such as a password strength meter and a round-trip date toggle. The backend, implemented in Flask, handles HTTP requests, session management, and business logic. Routes are defined for login, registration, chat, itinerary, destination, flights, and tips. User credentials (username, email, hashed password) are stored in an SQLite database (users.db), which is automatically created upon initialization. The AI integration is achieved via the OpenRouter API: the backend sends user prompts to the OpenRouter API using the model openai/gpt-3.5-turbo, accompanied by a system message that defines the agent's role as a travel expert. The generated text is returned and displayed to the user. Upon successful login, an email notification is sent to the user's registered address using Gmail's SMTP server with an app password. For flight suggestions, the system generates direct URLs to Skyscanner, Kayak, and Google Flights based on user input, providing external booking options.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

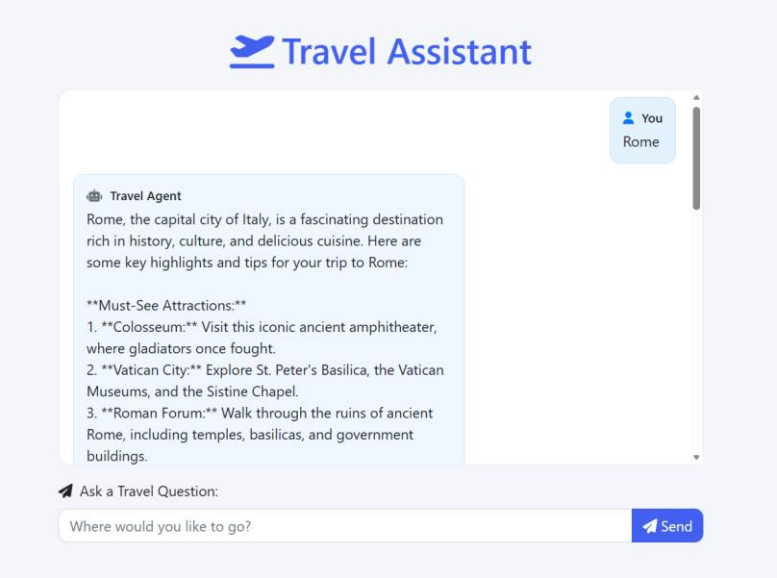
Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

4.3 MODULES DESCRIPTION

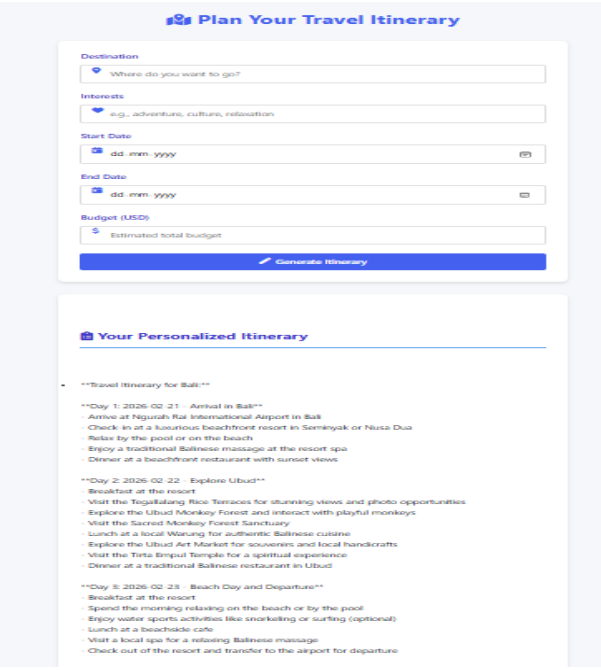
The user authentication module allows registration and login. During registration, passwords are hashed using werkzeug.security.generate_password_hash, and duplicate usernames or emails are rejected. Upon login, credentials are verified, a session is created, and a confirmation email is sent. The Flask secret key is randomly generated using uuid.uuid4() for security.

The AI chat module, accessible via the /chat route, accepts a user message and calls the generate_response function. This function constructs a system prompt ("You are a Travel AI Agent…") and sends it along with the user message to OpenRouter. The response is parsed and displayed in the chat interface, with conversation history maintained in memory during the session. Figure 2 shows the chat interface with a sample conversation.



[Figure 2: AI Chat interface displaying a user asking about Rome and the AI response.]

The itinerary planner module collects destination, interests, start and end dates, and budget via a form. A detailed prompt is created incorporating these inputs, and the AI response is split into numbered points using a regular expression before being displayed as a list. Figure 3 presents the itinerary input form and the generated itinerary output for a sample trip to Bali.



[Figure 3: Itinerary planner – input form (top) and generated itinerary (bottom).]

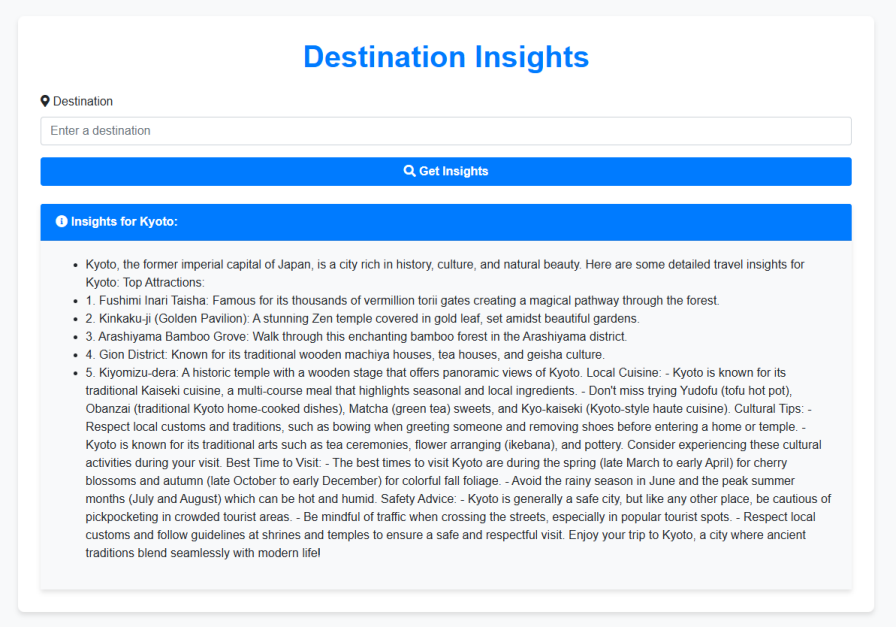
IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

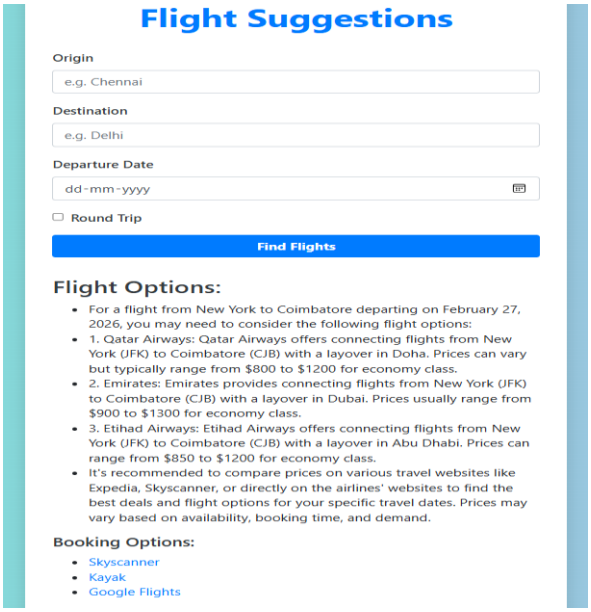
DOI: 10.17148/IARJSET.2026.13259

The destination insights module accepts a destination name and prompts the AI to provide detailed travel insights covering attractions, cuisine, cultural tips, best time to visit, and safety advice. The response is similarly formatted. Figure 4 shows the destination insights page for Kyoto.



[Figure 4: Destination insights for Kyoto with AI-generated information.]

The flight suggestions module takes origin, destination, departure date, and optionally a return date. The AI generates flight suggestions with approximate pricing and recommended airlines. Additionally, the system creates booking links for Skyscanner, Kayak, and Google Flights pre-filled with the user's criteria. A JavaScript snippet toggles the return date field based on the "Round Trip" checkbox. Figure 5 displays the flight search form and the resulting suggestions with external links.



[Figure 5: Flight suggestions page – search form (top) and AI-generated flight options with booking links (bottom).]

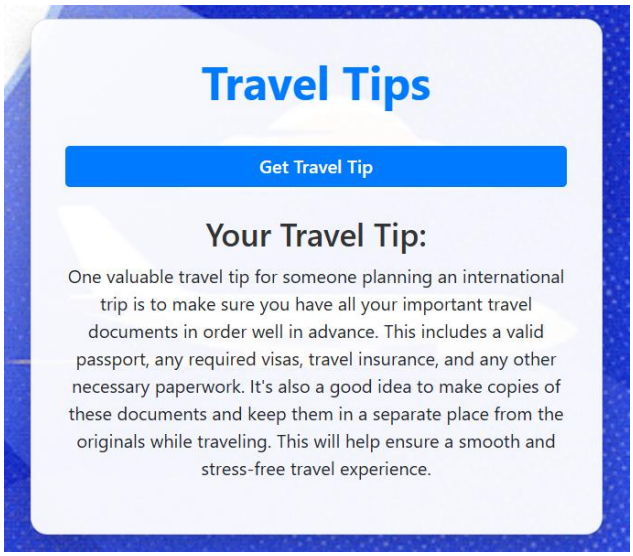
The travel tips module, on a button click, prompts the AI to provide a valuable travel tip for an international trip, displaying the generated tip on the same page. Figure 6 illustrates a sample tip.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

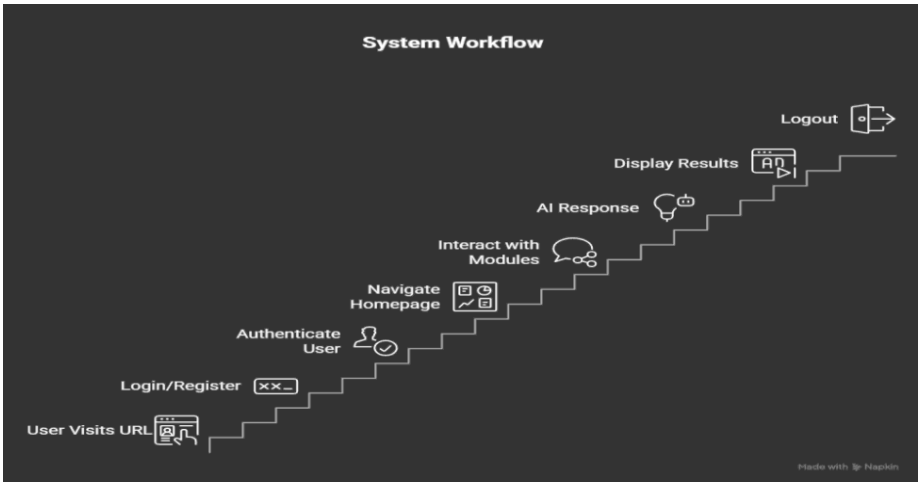


[Figure 6: Travel tips module showing a randomly generated tip.]

V. SYSTEM WORKFLOW

The operational flow of the system begins when a user visits the application URL and is presented with the login page. New users can register by filling in the registration form; upon successful registration, they are redirected to the login page. Returning users enter their credentials, which are validated against the SQLite database. If valid, a session is created and a login confirmation email is sent. After authentication, the user lands on the homepage, which displays feature cards for each module. Clicking any card navigates to the corresponding page.

In the chat module, the user types a travel-related question and submits it. The Flask backend forwards the prompt to OpenRouter, receives the AI response, and displays it in the chat window, retaining conversation history during the session. For the itinerary, destination, flights, or tips modules, the user fills in the required form fields and submits. The backend constructs a prompt based on the input, calls the OpenRouter API, processes the response (splitting into points if needed), and renders the result on the same page. In the flights module, along with AI-generated flight suggestions, the system provides clickable links to external booking platforms. The user can log out at any time, clearing the session. The data flow within the system is straightforward: user input is sent via HTTP POST to the appropriate Flask route, where form data is extracted and a prompt is built. This prompt is sent to the OpenRouter API with the system message, and the LLM generates a response that is returned to Flask. After processing, Flask renders the template with the result, and the browser displays the updated page. For example, in itinerary generation, the sequence involves the user accessing the /itinerary page, submitting the form, Flask building the prompt, calling generate_response, OpenRouter returning the AI text, Flask splitting it into points, and finally rendering itinerary.html with the itinerary points. This workflow ensures a seamless and interactive experience, leveraging AI to deliver personalized content in real time.



[Figure 6:System Workflow.]

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

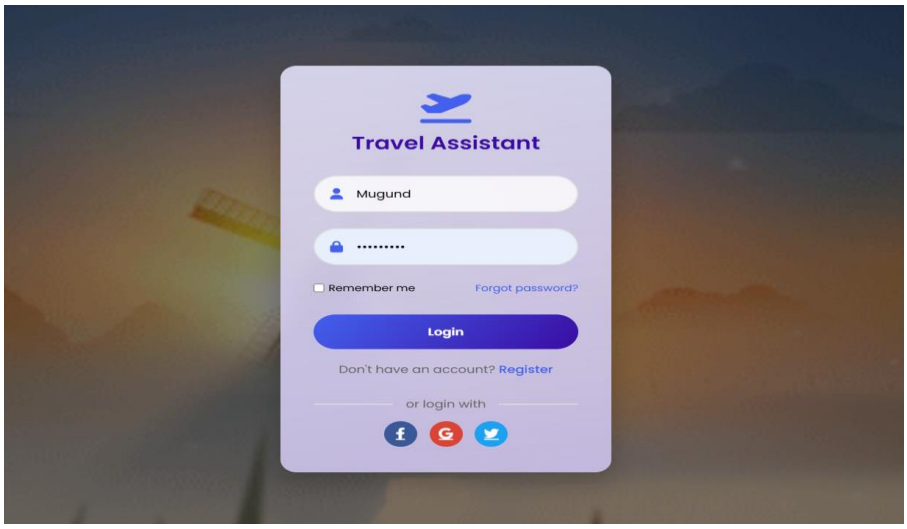
VI. IMPLEMENTATION AND RESULTS

6.1 TECHNOLOGY STACK

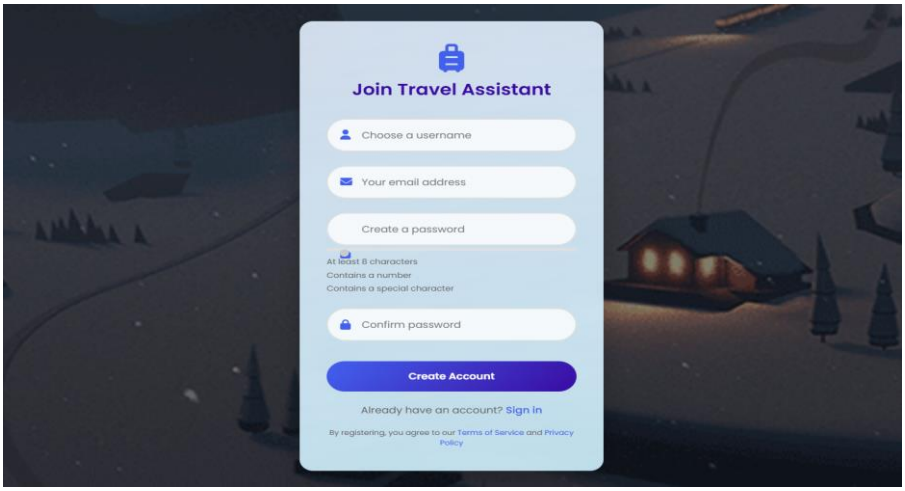
The system is implemented using Python 3 with the Flask micro-framework. The database is SQLite, accessed via the built-in sqlite3 module. The OpenRouter API is consumed using the requests library. Email notifications are sent via smtplib using a Gmail account with an app password. The frontend utilizes HTML5, CSS3, Bootstrap 4, and Font Awesome icons, with custom JavaScript for interactive elements.

6.2 USER INTERFACE

The user interface is designed to be intuitive and visually appealing. The login page (Figure 7) features a gradient background and a centered form with icons. The registration page (Figure 8) includes a password strength meter to guide users in creating strong passwords. The homepage (Figure 9) displays five feature cards, each with an icon and a brief description, linking to the respective modules.



[Figure 7: Login page.]



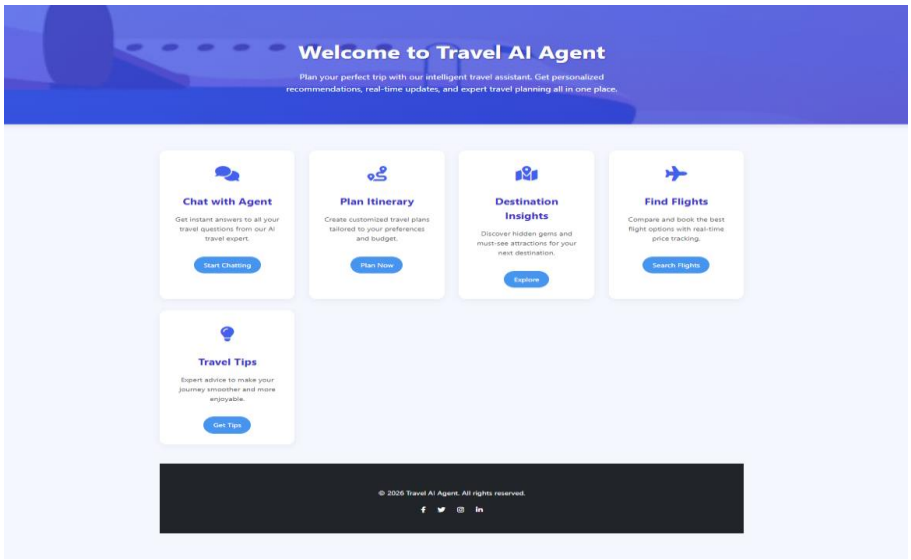
[Figure 8:Registration page with password strength meter]

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259



[Figure 9: Homepage with feature cards.]

The chat interface (Figure 2) presents messages in distinct bubbles for the user and the AI agent, with a typing indicator while waiting for a response. The itinerary form (Figure 3, top) uses input icons for clarity, and the generated itinerary (Figure 3, bottom) is displayed as a bulleted list within a styled card. The destination insights page (Figure 4) follows a similar pattern. The flight suggestions page (Figure 5) includes a checkbox to toggle the return date field, and the results section provides AI-generated text along with three prominent booking links. The travel tips page (Figure 6) is minimalist, with a single button and a text area for the tip.

6.3 SYSTEM TESTING AND SAMPLE OUTPUTS

The system was tested with various inputs to validate its functionality. For the chat module, a query "What are the must-see places in Rome for a 3-day trip?" produced a response listing the Colosseum, Vatican City, Trevi Fountain, Pantheon, and practical tips. The itinerary planner, given destination Bali, interests beach, yoga, and culture, dates 2025-07-10 to 2025-07-15, and a budget of $1500, generated a detailed day-by-day plan including accommodation suggestions and local experiences. Destination insights for Kyoto returned information on key attractions, kaiseki cuisine, and cultural etiquette. Flight suggestions for Chennai to Delhi with a round trip on specified dates provided sample airlines and approximate fares, along with direct links to Skyscanner, Kayak, and Google Flights pre-filled with the search criteria. The travel tips module offered advice such as "Always carry a digital and physical copy of your passport and visa."

6.4 PERFORMANCE EVALUATION

Response times for AI queries averaged between 2 and 4 seconds, depending on OpenRouter API latency. Database operations were negligible due to the small scale. Email notifications were delivered reliably. The application proved responsive on both desktop and mobile devices.

6.5 DISCUSSION

The system successfully demonstrates the integration of a large language model into a travel planning application, providing personalized and context-aware assistance. The modular design allows easy expansion and maintenance. However, limitations exist: the OpenRouter API key is hard-coded (should be environment-variable based), the AI may occasionally hallucinate or provide outdated information, and the system lacks live data integration for flights and hotels. Despite these, the prototype validates the concept and provides a strong foundation for future enhancements.

VII. CONCLUSION AND FUTURE SCOPE

This paper presented a Personalised Travel Planning System with an AI Assistant that combines a Flask web application with a large language model accessed via OpenRouter. The system successfully demonstrates automated itinerary generation, destination insights, flight suggestions, and conversational travel advice, all within a user-friendly interface. User authentication and email notifications add a layer of personalization and security. The results indicate that AI-powered travel planning can significantly reduce the time and effort required to organize trips, while offering tailored recommendations that adapt to individual preferences.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 13, Issue 2, February 2026

DOI: 10.17148/IARJSET.2026.13259

Nevertheless, the prototype has limitations that open avenues for future work. Real-time data integration with live flight APIs (e.g., Amadeus, Skyscanner API) and hotel booking engines would provide actual prices and availability. Developing native mobile applications for Android and iOS would enable on-the-go access. Multi-language support could be achieved using multilingual models, allowing conversations and outputs in various languages. Incorporating a user feedback loop would refine AI recommendations over time based on ratings and reviews. Experimenting with advanced models like GPT-4 or Claude could yield more nuanced and accurate responses. Adding itinerary export features (PDF or Google Calendar sync) would enhance usability, and social features such as sharing itineraries or collaborative planning could engage users further. Finally, integrating affiliate links for bookings could provide a monetization pathway. With these enhancements, the system can evolve into a fully-fledged commercial product that revolutionizes travel planning.

REFERENCES

[1]. Gupta, D. K. Singh, and A. K. Singh, "TravelBot: A Travel Assistance Chatbot," in Proc. Int. Conf. on Computing, Communication and Automation (ICCCA), 2017, pp. 1234–1239.
[2]. F. Ricci, L. Rokach, and B. Shapira, "Recommender Systems: Techniques, Applications, and Challenges," in Recommender Systems Handbook, Springer, 2022, pp. 1–35.
[3]. T. Brown et al., "Language Models are Few Shot Learners," in Advances in Neural Information Processing Systems, 2020.
[4]. OpenRouter Documentation, "Unified API for LLMs," [Online]. Available: https://openrouter.ai/docs.
[5]. Flask Documentation, "Flask – Web Development, One Drop at a Time," [Online]. Available: https://flask.palletsprojects.com/.
[6]. SQLite, "Small. Fast. Reliable. Choose any three." [Online]. Available: https://www.sqlite.org/.
[7]. Bootstrap, "The most popular HTML, CSS, and JS library in the world," [Online]. Available: https://getbootstrap.com/.
[8]. J. Smith and L. Johnson, "AI in Tourism: A Review of Current Applications and Future Trends," Journal of Travel Research, vol. 60, no. 4, pp. 712–728, 2021.