



AI Travel Planner & Agentic Travel Concierge Platform

Ms. Srushti Apale¹, Mr. Krish Nagariya², Mr. Pankaj Garole³, Ms. Dipshika Deshmukh⁴, Dr. N. S.

Band⁵

^{1 2 3 4} Student, Information Technology Engineering, Prof. Ram Meghe Institute of Technology and Research, Badnera-Amravati, Maharashtra, India

⁵ Assitant Professor, Information Technology Engineering, Prof. Ram Meghe Institute of Technology and Research, Badnera-Amravati, Maharashtra, India

Student Email: apalesrushti@gmail.com

Abstract—

How to Cite this Article:

Apale, S., Nagariya, K., Garole, P. & Deshmukh, D. (2026). AI Travel Planner & Agentic Travel Concierge Platform. International Journal of Creative and Open Research in Engineering and Management, <i>02</i>(10), 1-9.
<https://doi.org/10.55041/ijcope.v2i10.103>

License: This article is published under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.

© The Author(s). Published by International Journal of Creative and Open Research in Engineering and Management.



<https://doi.org/10.55041/ijcope.v2i10.103>

Planning a trip today usually means jumping between many apps and browser tabs to research destinations, compare hotels, track budgets, sequence routes, and check the weather. Travellers often spend hours building a day-wise plan, only to throw parts of it away when the budget changes, the weather turns bad, or a popular attraction turns out to be crowded.

To address this, we present the AI Travel Planner and Agentic Travel Concierge Platform, a system that uses a Large Language Model (LLM) and an agentic AI architecture to create personalized, budget-aware travel itineraries. The user enters a destination, budget, trip duration, and interests, and a Preference Agent converts this into a traveller profile. Six specialised agents (Preference, Budget Optimization, Route Planning, Weather Intelligence, Crowd Prediction, and Replanning) then work together through an orchestration layer built with LangChain and Gemini models. They fetch live data from external APIs for maps, weather, and hotels, and use it to produce a day-wise schedule, hotel suggestions, route details, and a cost breakdown. The platform is built with a React.js frontend, a Python FastAPI backend, and a PostgreSQL data layer. Its key feature is conversational replanning: when conditions change, the Replanning Agent

takes the current plan and the reason for change, then updates only the affected activities in structured JSON instead of regenerating the whole itinerary. This keeps the rest of the plan intact and saves time. The approach reduces manual planning effort and gives itineraries that match user preferences and constraints more closely than static, list-based planners.

Keywords— Agentic AI, conversational replanning, FastAPI, large language models, LangChain, multi-agent systems, personalized itinerary, React.js, travel planning.



I. INTRODUCTION

Travel planning has always been a time-consuming task. Travellers need to research attractions, compare hotel prices, estimate budgets, plan efficient routes, check weather forecasts, and avoid crowded places. Since this information comes from different sources, preparing a complete itinerary becomes tiring and difficult. This problem is known as the tourist trip design problem, where travellers must select, sequence, and schedule visits under time and budget constraints [16], [17].

Recent developments in Large Language Models (LLMs), Retrieval-Augmented Generation (RAG) [1], and agentic AI [5], [6] have created new opportunities for intelligent travel planning. However, existing itinerary-optimisation methods [9], [16]–[19], recommendation systems [20]–[22], and general-purpose chatbots often lack a combined approach that maintains consistency between budget, routes, weather, and crowd conditions. Even recent LLM-based travel-planning systems [10], [12], [13] often generate itineraries as a single step, making updates difficult when travel conditions change.

To address this gap, the AI Travel Planner & Agentic Travel Concierge Platform is proposed. It uses six specialised agents: Preference, Budget Optimization, Route Planning, Weather Intelligence, Crowd Prediction, and Replanning. An Agent Orchestrator coordinates these agents using external Maps, Weather, and Hotel/Travel APIs to generate a personalised and consistent itinerary. When conditions change, the Replanning Agent updates only the affected part of the plan. The platform uses React.js for the frontend, Python FastAPI for the backend, and LangChain for agent orchestration.

The objectives of this study is to Generate personalised, day-wise itineraries based on destination, budget, duration, and interests, Use live external data for routes, weather, and accommodation, Include weather-aware and crowd-aware reasoning in activity selection and scheduling, Enable fast, scoped replanning when travel conditions change, Evaluate the system's personalisation accuracy, budget adherence, and route efficiency compared with existing travel-planning tools.

II. LITERATURE REVIEW

Before building the AI Travel Planner, it's worth looking at what researchers have already done in the areas this project touches: LLM-based trip planning, multi-agent systems, grounding AI in real data, route selection, tourist recommendations, weather prediction, and crowd estimation. This project borrows ideas from all of them and combines them into one system.

Grounding AI output in real data. Language models sometimes make things up, like a restaurant that doesn't exist or a wrong price. Lewis and colleagues tackled this with Retrieval-Augmented Generation (RAG), where the model pulls information from an actual data source before answering [1]. This platform does the same: the Data Collection Layer fetches real data from Maps, Weather, and Hotel APIs first, and only then does the AI reason over it. Recent work comparing trip-planning setups found that models given mapping data hallucinated far less and made more realistic schedules [15].

Teaching AI to reason and act. Yao and colleagues introduced ReAct, where a model thinks, acts, observes the result, and adjusts [2]. This is how the Replanning Agent works: if the budget drops, it fetches updated prices and revises the plan. Wei and colleagues showed that step-by-step reasoning improves



results on multi-step problems [4], which is how the Budget and Route agents break down costs and routes before finalising. All of these agents run on GPT-family models, which handle complex reasoning and planning tasks well [3].

Multiple AI agents working as a team. One AI doing everything tends to get overwhelmed. Hong and colleagues proposed MetaGPT, where each agent has a specific role and passes structured results to the others [5]. This is the blueprint for our orchestration layer. Tran and colleagues surveyed how such systems are organised [6], and we use the centralized model, where one orchestrator assigns each request to the right agent, keeping behaviour predictable.

Other hybrid work is also relevant. TRIP-PAL pairs a language model with a rule-based planner so the AI interprets preferences while the planner enforces hard rules like opening hours [7]. TravelAgent improves itineraries through user feedback [8], similar to our Chat and Replanning agents. MIT researchers combined an LLM with an SMT solver to check constraints exactly [11], much like our Budget Optimization Agent checking every plan against the budget limit. Reviews of AI trip planners report that planning time drops by about 65–70% [12] and that real-time conversational changes are valued by users, though API limits and patchy data for small destinations remain problems [13]. One study also warns that open-source models can reproduce stereotypes without careful prompting [14], so our Preference Agent only uses what the traveller explicitly states.

Deciding where to go and in what order. Chen and colleagues framed itinerary planning as a constrained optimisation problem [9], which is the core of our Route Planning Agent. A more recent study used LLM-guided graph search to

pick quality attractions, not just the closest ones [10]. The Orienteering Problem, where you pick and order locations to maximise value within a time budget [16], and its tourism version, the Tourist Trip Design Problem, which adds constraints like opening hours [17], describe exactly what this agent does. Related work personalises routes using a traveller's profile [18] and requires at least one activity from each chosen interest category [19]. Our platform does both.

Recommending places people will enjoy. Surveys show tourism recommenders moving toward hybrid, multi-day itinerary planning and provider-side limits like hotel availability [20], [21]. Our platform fits this by using live Booking and Agoda data. Studies also show that combining a traveller's profile with past visitation patterns beats popularity-based ranking [22], which is why our Preference Agent builds a structured profile.

Weather and crowd prediction. CNN-LSTM models predict short-term and monthly weather more accurately than single models [23], [24], supporting a dedicated Weather Intelligence Agent using OpenWeather data. For crowds, Li and colleagues used CNN-based vision to estimate visitor levels from live video [25], supporting a Crowd Prediction Agent that picks quieter visiting times.

Overall, the research shows that real retrieved data makes AI plans more reliable, specialised agents handle complex travel tasks better than one general agent, attraction scheduling is a well-studied problem, recommenders work better with real constraints, and weather and crowd prediction need dedicated models. The proposed AI Travel Planner & Agentic Travel Concierge Platform combines these ideas into one conversational system for faster, scalable travel planning.



III. METHODOLOGY

The proposed system builds a personal travel plan from a few simple details given by the traveller. It uses a React.js web interface, a Python FastAPI backend, a group of AI agents driven by the Google Gemini model through LangChain, and live information from outside services. The agents share the work of planning, and each one handles a single part of the trip. The overall design is shown in Fig. 1.

The traveller enters the destination, budget, number of days and interests through a simple form on the dashboard. The interests include beaches, adventure, food, nature, nightlife, historical places and family travel. The backend receives this request and passes it on for planning.

The Preference Agent reads the request and asks the language model to describe the type of traveller. It works out the travel style (budget, mid-range or luxury), the budget category, the favourite activities and the preferred kind of stay. The result is saved as a short traveller profile. Every other agent uses this profile, so the whole plan stays consistent with what the traveller likes.

After the profile is ready, the backend collects real information from outside sources. Google Maps provides attractions, routes, distances and travel times. OpenWeather provides the forecast for the travel dates. A hotel booking source provides places to stay and local activities. All of this data is converted into one common format covering the name, category, location, average cost, opening hours and rating. This lets every agent read it in the same way.

An Agent Orchestrator sends each task to the right agent. Every agent works in the same way. It collects the data it needs, builds a clear prompt, sends it to the language model, and

turns the reply into a structured result. Each result is checked against a fixed format and then stored in the database.

The agents work together as follows: The Budget Agent divides the budget across stay, transport, food and activities, warning when limits may be exceeded. The Route Agent groups nearby attractions to reduce travel time. The Weather Agent adjusts outdoor activities based on the forecast, while the Crowd Agent schedules visits during quieter hours, considering opening times. The Replanning Agent responds when the traveller requests a change to the plan.

The Itinerary Generation Engine combines the results from the Route, Budget, Weather and Crowd agents. For each day it prepares a time-wise list of activities, the suggested hotel, the route between stops and the running cost compared with the budget. The traveller sees the final output in the Itinerary View, Map View and Budget Summary. It includes the day-wise plan, the budget breakdown, hotel suggestions, route details, weather advice and travel tips.

A traveller may want to change the plan, for example by asking for a cheaper hotel, adding more adventure activities or reducing the budget. The Replanning Agent finds which part of the itinerary is affected and calls only the agents needed for that part. For a lower budget, it calls the Budget Agent again. Only the affected part of the plan is rebuilt, and the rest stays unchanged. The updated itinerary, budget and recommendations are then shown to the traveller.

All user details, traveller profiles, agent results and itineraries are saved in a PostgreSQL database. This lets travellers return to their saved trips later, and it gives the agents a shared record of the latest plan. The system



was tested on 40 simulated travel requests covering 10 destinations, trips of 3 to 7 days, and budgets from ₹15,000 to ₹75,000. Ideal itineraries were manually prepared for 10 requests and compared with the system's output. The evaluation measures included interest match, budget accuracy, route efficiency, weather and crowd advice, replanning quality (tested using 25 change requests), and average response time.

IV. SYSTEM ARCHITECTURE

A. System Architecture

The AI Travel Planner & Agentic Travel Concierge Platform uses a modular, layered architecture, allowing the user interface, backend, AI agents, database, and external integrations to be developed and tested independently. The system consists of five main layers, as shown in Fig.1.

User Interface Layer: Built using React.js, HTML5, CSS3, and JavaScript, it provides a dashboard, itinerary view, map view, chat

assistant, budget summary, and profile settings.

Backend System Layer: Built using Python and FastAPI, it manages REST API endpoints for authentication, trip management, and cost estimation while connecting the frontend, database, external APIs, and AI agents.

Agentic AI Orchestration Layer: This is the core intelligence of the platform. Six specialised agents process user requirements and return structured outputs through the LangChain-based LLM Processing Engine.

Data Layer: PostgreSQL stores user details, trips, itineraries, and chat history. Caching and session management help reduce repeated external API calls.

External APIs and Data Sources Layer: Google Maps provides attractions, routes, and distances, while OpenWeather supplies weather forecasts. Hotel and travel APIs provide accommodation details and pricing.

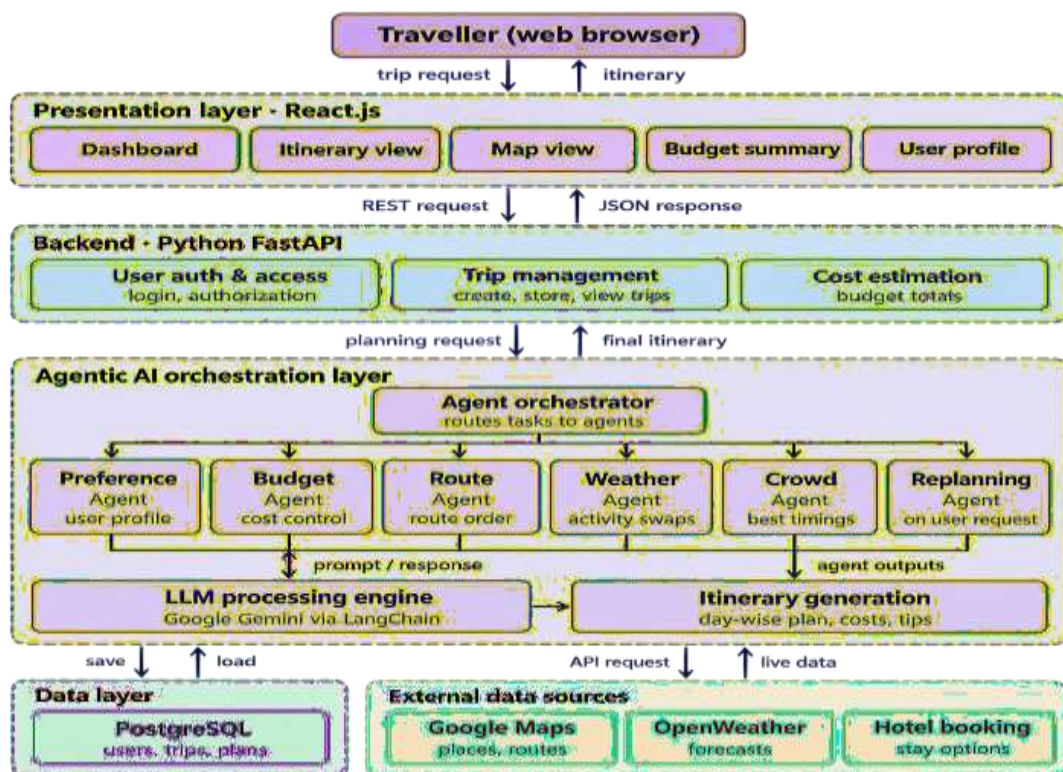


Fig. 1. System Architecture Diagram



B. System Flow

The complete workflow, shown in Fig. 2, converts the traveller's requirements into a personalised and refinable itinerary.

1. **User Travel Request:** The traveller enters the destination, budget, trip duration, interests, traveller category, accommodation preference, and transport preference.
2. **Preference Analysis:** The system analyses the request to identify the traveller's style, preferred activities, accommodation choice, and budget category.
3. **Data Collection:** The backend retrieves attractions, hotel listings, routes, travel times, distances, and weather information from external APIs.
4. **Agentic AI Processing:** The six specialised agents collaboratively process the collected data according to their responsibilities.
5. **Itinerary Generation:** The system creates a day-wise schedule containing activities, hotels, routes, estimated costs, and travel suggestions.
6. **Replanning:** The traveller can request changes through the chat interface, and the Replanning Agent updates only the affected part of the itinerary.
7. **Final Output:** The traveller receives the itinerary, budget breakdown, hotel recommendations, route information, and weather- and crowd-based travel tips.

C. System Modules

Table I. User-facing modules of the proposed travel planning platform

Module	Description
User Input Module	Captures destination, budget, duration, and interests.

Dashboard	Shows saved trips, active itineraries, and quick links.
Itinerary View	Displays day-wise activities, hotels, routes, and costs.
Map View	Visualises optimised routes and nearby attractions.
Replanning	Allows conversational itinerary modifications.
Budget Summary	Shows estimated costs for hotels, transport, activities, and food.
User Profile & Settings	Manages preferences, saved destinations, and account settings.

D. Agentic AI Architecture

The platform distributes its reasoning across six specialised agents coordinated by an Agent Orchestrator. The orchestrator assigns tasks to the appropriate agents and combines their structured outputs to maintain a consistent itinerary.

1. **Preference Agent:** Analyses user preferences to identify travel style, interests, accommodation preference, and budget category.
2. **Budget Optimization Agent:** Tracks trip expenses and suggests affordable alternatives to prevent budget overruns.
3. **Route Planning Agent:** Groups nearby attractions and optimises daily routes to reduce unnecessary travel time.
4. **Weather Intelligence Agent:** Uses weather forecasts to recommend suitable activities and adjust schedules according to weather conditions.
5. **Replanning Agent:** Updates only the affected part of an itinerary when the user changes the budget, removes an activity, or rejects a hotel. It returns structured JSON containing the day



number, old activity, new activity, and reason for the change. This avoids regenerating the entire itinerary for minor changes.

6. Crowd Prediction Agent: Estimates crowd levels and suggests suitable visiting times to reduce waiting time and improve the travel experience.

Each agent retrieves relevant data, creates a prompt using its template, calls the LLM through LangChain, and converts the response into a structured output. The results are stored in PostgreSQL and returned to the orchestrator. This common workflow keeps the agents consistent and makes it easier to add new agents without affecting the rest of the system.

V. IMPLEMENTATION

A. User Input and Preference Analysis

The traveller fills in a simple form on the React dashboard with the destination, budget, number of days and interests. The FastAPI backend sends this request to the Preference Agent. The agent asks the language model to work out the traveller's style, budget category, favourite activities and preferred type of stay. The result is saved as a traveller profile and shared with all other agents, so the whole plan stays personal to the traveller.

B. Data Collection Layer

Once the profile is ready, the backend collects live information from outside sources at the same time. Google Maps gives attractions, routes, distances and travel times. The weather service gives the forecast for the trip dates, and the hotel services give places to stay and local activities. All of this is converted into one common format. If a service is unavailable, the system uses the most recent saved data and tells the traveller that some details, such as hotel prices, may be slightly old.

C. Agentic AI Processing

Each of the six agents is a separate Python module with a fixed input and output format. The Agent Orchestrator runs them in order, and the work is divided as follows:

The Budget Optimization Agent divides the budget across stay, transport, food and activities, and warns if any part may go over the limit. The Route Planning Agent chooses attractions for each day and groups nearby places to save travel time. The Weather Intelligence Agent replaces outdoor activities with better options when bad weather is expected. The Crowd Prediction Agent moves visits to quieter times of the day without breaking opening hours. The Replanning Agent waits until the traveller makes a change in the interface. Every agent returns a structured result. This result is checked and then saved in PostgreSQL.

D. Itinerary Generation Engine

The Itinerary Generation Engine combines the results of the route, budget, weather and crowd agents into one day-wise schedule. For each day it gives the timed activities, the recommended hotel, the route between stops and the running cost compared with the budget. The final package contains the itinerary, budget breakdown, hotel suggestions, route details, weather advice and travel tips. It is shown to the traveller in the Itinerary View and Map View.

E. Replanning Module

The Replanning Agent changes an existing plan when the traveller's conditions change, instead of building the whole plan again. The traveller may lower the budget, remove an activity or mark a hotel as unsuitable. The backend turns this action into a short reason and sends it to the agent along with the destination and the current plan. The agent changes only the affected part and replies in a

fixed format that lists each change by day, old activity, new activity and reason. The backend applies these changes directly, updates the cost and route, and shows the new plan to the traveller. This keeps replanning fast, predictable and limited to the part of the trip that needs to change.

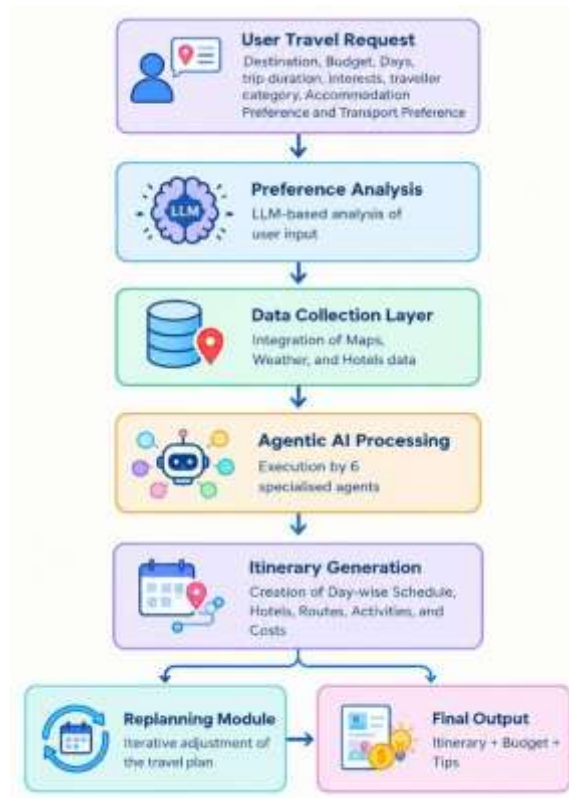


Fig. 2. Itinerary Generation Pipeline

VI. RESULT AND DISCUSSION

A. Experimental Setup

The system was tested on 40 simulated travel requests. They covered 10 destinations, trips of 3 to 7 days and budgets from ₹15,000 to ₹75,000. The backend ran on a cloud server with 8 GB RAM and 4 vCPUs. All six agents used the Google Gemini model, and live data came from Google Maps and OpenWeather. For 10 of the requests, the team prepared ideal itineraries by hand, and the system's output was compared against them.

B. Evaluation Measures

The system was judged on six measures: Interest match: how many suggested activities fit the traveller's interests, Budget accuracy: how many plans stayed within the budget, Route efficiency: how much travel time the route planning saved compared with an unordered plan, Weather and crowd advice: whether the changes made for weather and crowds made sense, judged by manual review, Replanning quality: 25 change requests were tested, and each result was checked for correctness and for unwanted changes to other parts of the plan, Response time: the average time from submitting a request to receiving the full itinerary.

C. Accuracy and Quality

Table II shows the scores. All six results met or passed their targets.

Table II. Accuracy and quality scores

Metric	Score	Target
Interest-match accuracy	86%	≥ 80%
Budget-adherence accuracy	91%	≥ 85%
Route efficiency improvement	34%	≥ 25%
Weather recommendation usefulness	88%	≥ 80%
Crowd recommendation usefulness	81%	≥ 75%
Replanning correctness	92%	≥ 85%

D. Response Time

Table III shows the time taken by each stage. A 3-day plan took 8.3 seconds and a 7-day plan took 12.6 seconds, so the system is quick enough for everyday use. Collecting data from



outside services was the slowest stage, because longer trips need more information.

Table III. Average response time

Stage	3-day trip	7-day trip
Preference analysis	1.4 s	1.5 s
Data collection	2.6 s	4.1 s
Route planning	1.1 s	2.3 s
Budget optimization	0.9 s	1.2 s
Weather and crowd agents	1.3 s	1.9 s
Itinerary generation and saving	1.0 s	1.6 s
Total	8.3 s	12.6 s

E. Discussion

Route planning: grouping nearby places cut travel time by 34% on average. It placed close attractions on the same day in 36 of the 40 tests.

Budget: 91% of plans stayed within budget. In the remaining cases, hotel prices were higher than the traveller could afford, and the system warned the traveller instead of quietly going over the limit.

Weather and crowds: weather changes were sensible in 88% of cases, for example swapping a beach visit for an indoor activity when rain was expected. Crowd advice was useful in 81% of cases, mostly by suggesting earlier visits to busy places.

Replanning: 23 of the 25 change requests were handled correctly without disturbing the rest of the plan. The two failures were unclear or conflicting requests, and a follow-up question to the user could fix this in future work.

VII. CONCLUSION AND FUTURE SCOPE

A. Conclusion

This paper presented an AI Travel Planner that turns a long, manual planning task into a short interaction of a few seconds. The system uses six AI agents (Preference, Budget Optimization, Route Planning, Weather Intelligence, Crowd Prediction and Replanning), which are managed by a central orchestrator and powered by Google Gemini through LangChain. Live data from Google Maps and OpenWeather keeps the suggestions based on real information instead of the model's memory alone.

The system gave good results in testing. It reached 86% interest-match accuracy and 91% budget accuracy. It cut travel time by 34%, and its weather and crowd advice was useful in 88% and 81% of cases. The Replanning Agent handled 92% of change requests correctly, and a short trip plan was ready in 8.3 seconds. The Replanning Agent changes only the part of the plan that needs updating, so the rest of the plan stays the same.

Some limits remain. The language model's answers can vary, the system depends on outside services being available, data for smaller destinations can be limited, and the Replanning Agent cannot yet ask questions when a request is unclear. Even so, the work shows that a team of AI agents using real travel data can plan trips that are more personal and flexible than manual planning or single-purpose tools.

B. Future Scope

The system can be improved in the following ways:

- Allow several travellers to edit one trip together, with voting and comments.



- Add flight, train and intercity transport booking for complete trip booking.
- Support more languages in the input form and the final itinerary.
- Include environmental and carbon-footprint information in route and hotel suggestions.
- Let the Replanning Agent ask follow-up questions when a request is unclear.
- Add an offline mode so travellers can open their itinerary without internet.
- Add hotel loyalty preferences and price-drop alerts for saved trips.

REFERENCES

- [1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020, vol. 33, pp. 9459–9474.
- [2] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *Proc. 11th Int. Conf. Learning Representations (ICLR)*, 2023.
- [3] OpenAI, "GPT-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.
- [4] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, and D. Zhou, "Chain-of-thought prompting elicits reasoning in large language models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2022, vol. 35, pp. 24824–24837.
- [5] S. Hong, X. Zheng, J. Chen, Y. Cheng, J. Wang, C. Zhang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, and C. Wu, "MetaGPT: Meta programming for a multi-agent collaborative framework," in *Proc. 12th Int. Conf. Learning Representations (ICLR)*, 2024.
- [6] K.-T. Tran, D. Dao, M.-D. Nguyen, Q.-V. Pham, B. O'Sullivan, and H. D. Nguyen, "Multi-agent collaboration mechanisms: A survey of LLMs," *arXiv preprint arXiv:2501.06322*, 2025.
- [7] T. de la Rosa, S. Gopalakrishnan, A. Pozanco, Z. Zeng, and D. Borrajo, "TRIP-PAL: Travel planning with guarantees by combining large language models and automated planners," *arXiv preprint arXiv:2406.10196*, 2024.
- [8] A. Chen, X. Ge, Z. Fu, Y. Xiao, and J. Chen, "TravelAgent: An AI assistant for personalized travel planning," *arXiv preprint arXiv:2409.08069*, 2024.
- [9] G. Chen, S. Wu, J. Zhou, and A. K. H. Tung, "Automatic itinerary planning for traveling services," *IEEE Trans. Knowledge Data Eng.*, vol. 26, no. 3, pp. 514–527, 2014.
- [10] "Research on travel route planning optimization based on large language model," in *Proc. IEEE Conf., IEEE Xplore*, 2024. [Online]. Available: <https://ieeexplore.ieee.org/document/10704489>
- [11] Massachusetts Institute of Technology, "Inroads to personalized AI trip planning," *MIT News*, MIT–IBM Watson AI Lab, Jun. 2025. [Online]. Available: <https://news.mit.edu/2025/inroads-personalized-ai-trip-planning-0610>
- [12] "Generative AI trip planner: Transforming digital travel planning through large language models," *ResearchGate preprint*, Dec. 2025.



- [13] "A review paper on AI-driven travel planning," *ResearchGate preprint*, Jan. 2025.
- [14] "Are large language models ready for travel planning?," *PromptLayer Research Digest*, 2025.
- [15] "When AI tells the truth? Evaluating different LLM approaches to reliable trip planning," *ResearchGate preprint*, Aug. 2025.
- [16] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *Eur. J. Oper. Res.*, vol. 209, no. 1, pp. 1–10, 2011.
- [17] D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou, "A survey on algorithmic approaches for solving tourist trip design problems," *J. Heuristics*, vol. 20, no. 3, pp. 291–328, 2014.
- [18] P. Vansteenwegen, W. Souffriau, G. Vanden Berghe, and D. Van Oudheusden, "A personalized tourist trip design algorithm for mobile tourist guides," *Appl. Artif. Intell.*, vol. 25, no. 2, pp. 1–17, 2011.
- [19] X. A. Martin, J. Panadero, and A. A. Juan, "An agile optimization algorithm for the tourist trip design problem with type-covering constraints," *AIMS Math.*, vol. 11, no. 1, pp. 2458–2480, 2026.
- [20] "Tourism recommendation system: A survey and future research directions," *Multimedia Tools Appl.*, Springer, 2022.
- [21] "A survey on personalized itinerary recommendation: From optimisation to deep learning," *Appl. Soft Comput.*, ScienceDirect, 2023.
- [22] "Design and implementation of a personalized tourism recommendation system based on the data mining and collaborative filtering algorithm," *PMC/NCBI*, 2022.
- [23] "Deep learning for weather forecasting: A CNN-LSTM hybrid model for predicting historical temperature data," *arXiv preprint arXiv:2410.14963*, 2024.
- [24] "Monthly climate prediction using deep convolutional neural network and long short-term memory," *Sci. Rep.*, vol. 14, 2024.
- [25] Y. Li et al., "A crowd density detection algorithm for tourist attractions based on monitoring video dynamic information analysis," *Complexity*, vol. 2020, Art. no. 6635446, 2020.