Personal Travel Solver: A Preference-Driven LLM-Solver System for Travel Planning

Zijian Shao[1], Jiancan Wu[1,2*], Weijian Chen[1,2*], Xiang Wang[1]
[1]University of Science and Technology of China
[2]Institute of Dataspace, Hefei Comprehensive National Science Center
{shaozijian, naure}@mail.ustc.edu.cn,
{wujcan, xiangwang1223}@gmail.com

Abstract

Personal travel planning is a challenging task that aims to find a feasible plan that not only satisfies diverse constraints but also meets the demands of the user's explicit and implicit preferences. In this paper, we study how to integrate the user's implicit preference into the progress of travel planning. We introduce RealTravel, an augmented version of the TravelPlanner by incorporating real user reviews and point-of-interest metadata from Google Local. Based on RealTravel, we propose Personal Travel Solver (PTS), an integrated system that combines LLMs with numerical solvers to generate travel plans that satisfy both explicit constraints and implicit user preferences. PTS employs a novel architecture that seamlessly connects explicit constraint validation with implicit preference modeling through five specialized modules. The experimental results demonstrate the system's effectiveness, achieving better performance than baseline methods, and improvement in the level of personalization. Our data and code are available at PersonalTravelSolver.

1 Introduction

Recent advances in Large Language Models (LLMs) have demonstrated remarkable capabilities in understanding complex instructions and reasoning across diverse domains (Huang et al., 2022a,b). While LLMs show promise in planning tasks (Song et al., 2023; Huang et al., 2022a), their application to real-world planning scenarios remains challenging, particularly when dealing with multiple interacting constraints and personalization requirements. Travel planning serves as an example of such challenges (Xie et al., 2024): given a natural language query like "I'm planning a 3-day trip from Grand Rapids to Atlanta from March 18-20, 2022. I'd like a private room for accommodation, and I'm interested in trying both Latin American and European restaurants. Could you help me plan this within a $1500 budget?", a travel planner must not only satisfy constraints (*e.g.,* time, budget) but also understand user preferences (*e.g.,* dining preferences) while coordinating numerous interdependent decisions (*e.g.,* flights, accommodations, activities). Even for humans, this task requires significant effort, involving iterative information seeking, constraint checking, and preference-based decision-making.

Recent efforts have attempted to address travel planning through various approaches. Xie et al. (2024) introduce TravelPlanner, a benchmark for U. S. domestic travel planning. However, results show that even state-of-the-art LLMs like GPT-4 (Achiam et al., 2023) achieve only a 4.4% success rate in generating valid plans (Xie et al., 2024). While some approaches have formalized travel planning as a constraint satisfaction problem using numerical solver (Hao et al., 2024), achieving better constraint satisfaction rates, they often overlook the crucial aspect of personalization, thereby producing technically feasible but potentially unsatisfying plans. Conversely, attempts to incorporate personalization through synthetic user profiles (*e.g.,* TravelPlanner+ (Singh et al., 2024), TravelAgent (Chen et al., 2024a)) or multi-turn dialogues (Jiang et al., 2024) fail to reflect the complexity of real-world user preferences (Yan et al., 2023).

To address the challenge of modeling real-world users' implicit travel preferences, we present **RealTravel**, a novel dataset that extends the TravelPlanner benchmark by incorporating authentic user reviews and metadata sourced from Google Local (Li et al., 2022). Specifically, we extract point-of-interest (POI) data and user reviews for cities covered in the TravelPlanner benchmark. RealTravel provides a comprehensive resource grounded in real user behaviors, thereby fostering research in personalized travel planning.

To generate travel plans that rigorously satisfy

*Corresponding authors.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 27622–27642
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

user constraints while reflecting individual preferences, we propose **Personal Travel Solver (PTS)**, a personalized travel planning system that integrates large language models (LLMs), the SCIP solver (Bestuzheva et al., 2021), and a robust data processing framework to address the aforementioned challenges. PTS comprises five distinct modules: the Translator Module, Search Module, Preference Encode Module, Re-rank Module, and Planning Module. All modules are carefully designed and can operate seamlessly together. The Translator Module converts natural language queries into structured JSON representations, ensuring precise interpretation of user requirements. The Search Module then retrieves contextually relevant candidate information from integrated databases. To further refine personalization, the Preference Encode Module extracts implicit user constraints from review histories. The Re-rank Module performs secondary filtering on these candidates, prioritizing high-quality options aligned with user expectations. Finally, the Planning Module leverages the SCIP solver to generate feasible travel plans that holistically satisfy constraints.

We evaluate our framework on the RealTravel dataset, demonstrating its effectiveness in generating high-quality travel plans. Our approach demonstrates superior constraint satisfaction compared to baseline methods. For personalization evaluation, we leverage the "LLM as judge" paradigm, where a large language model assesses travel plans based on user preference alignment, providing an objective measure of personalization quality. The experiments demonstrate that our framework improves personalization of travel plan.

Our contributions include:

- We introduce **RealTravel**, a novel dataset that augments existing travel planning benchmarks with real user reviews and POI metadata, enabling research on preference-driven travel planning.
- We develop **Personal Travel Solver (PTS)**, an integrated system that effectively bridges the gap between explicit constraint satisfaction and implicit preference understanding through a five-module architecture.
- The experiments show that our framework can generate travel plans that satisfy explicit constraints and capture implicit user preference.

2 Related Works

2.1 Travel Planning

The global tourism boom fuels a strong demand for personalized travel assistance, as today's travelers seek personal experiences beyond generic sightseeing. While major travel platforms like Expedia (Expedia Group, 2024) and Booking (Booking Holdings Inc., 2024) introduced booking integrations and categorical recommendations, and others like Roadtrippers (Roadtrippers, LLC, 2024) provide foundational route planning, they still face considerable limitations in satisfying the deep personalization needs of travelers. These platforms often fail to effectively mining users' underlying preferences, and still heavily relies on manual selection of points of interest (POI) by users, thus making it difficult to deliver automated and personalized travel plans.

Large language models (LLMs) (Achiam et al., 2023) have recently shown significant potential in few-shot learning (Huang et al., 2022a), tool using (Schick et al., 2023) and reasoning (Huang et al., 2022b), prompting researchers to explore their application in planning tasks (Song et al., 2023; Huang et al., 2022a). Initial efforts focused on benchmarking LLMs' planning capabilities (Bohnet et al., 2024) and developing reasoning frameworks for multi-phase planning tasks (Xie and Zou, 2024).

As for travel planning, it is a complex task involving information gathering, point-of-interest (POI) selection, route mapping, and personalization to meet diverse user needs (Halder et al., 2024). Benchmarks such as NaturalPlan (Zheng et al., 2024) and TravelPlanner (Xie et al., 2024) aim to rigorously evaluate logical constraint satisfaction in cross-city planning tasks. However, due to the complexity of constraints and diverse user requirements, directly employing LLMs for travel planning often yields suboptimal results (Xie et al., 2024). To address these limitations, recent studies propose architectural and methodological enhancements. (Chen et al., 2024c) introduces feedback-aware fine-tuning (FAFT) to improve planning performance through supervised fine-tuning, while (Gundawar et al., 2024) augments LLMs with external verifiers for iterative constraint validation. (Lee et al., 2025) integrated iterative travel plan sampling with optimization algorithms, achieving appealing performance on the TravelPlanner benchmark.

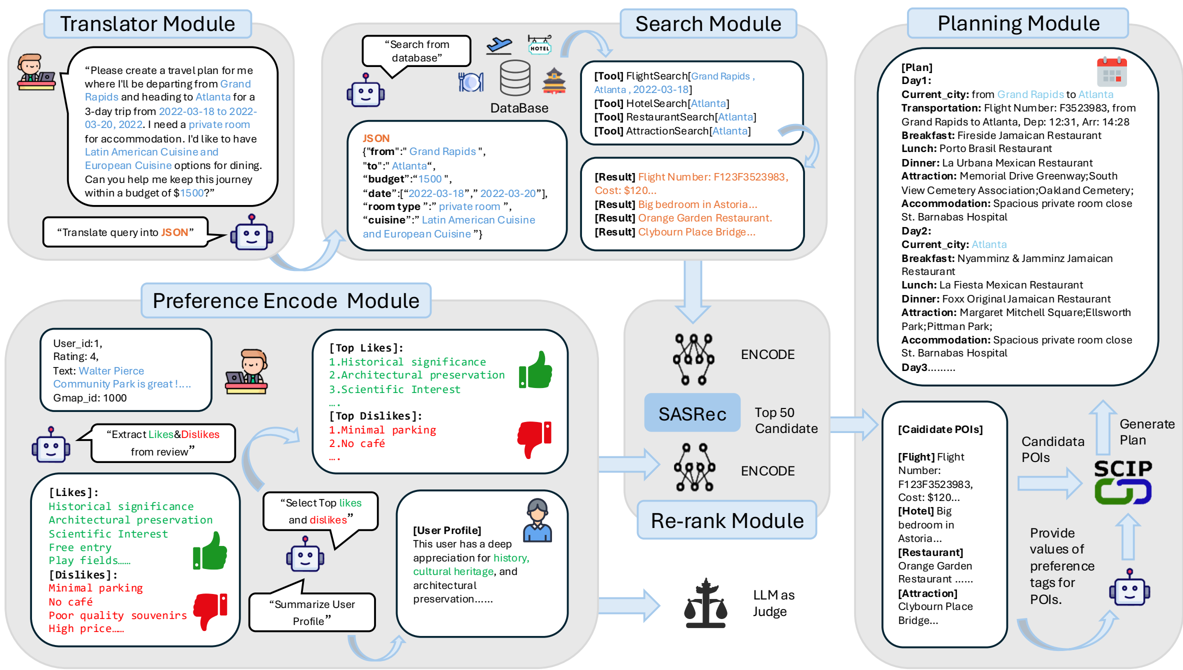

Figure 1: Overview of Personal Travel Solver. Given a natural language query, the system employs five interconnected modules to generate personalized travel plans. The Translator Module converts user queries into structured JSON format, while the Search Module retrieves candidate information from databases. The Preference Encode Module analyzes user review histories to identify key preferences, and the Re-rank Module employs SASRec and text embeddings to select top candidate POIs. Finally, the Planning Module utilizes the SCIP solver to generate feasible travel plans that satisfy both explicit constraints and implicit user preferences.

2.2 Hybrid System of LLM and Solvers

Recent research has explored the integration of symbolic solvers with large language models (LLMs) in both abstract planning tasks (*e.g.,* (Pan et al., 2023; Deng et al., 2024; Silver et al., 2024; de la Rosa et al., 2024)) and real-world planning applications (*e.g.,* (Tang et al., 2024; Hao et al., 2024; Ju et al., 2024)).

These hybrid systems combine the natural language understanding capability of LLMs with the precise constraint satisfaction mechanisms of formal solvers. (Hao et al., 2024) focus on formal constraint verification, prompting LLMs to generate code executable by SMT solvers to ensure generated plans strictly adhere to explicit constraints. In contrast, (Ju et al., 2024) tackle guaranteed itinerary scheduling by employing mixed-integer linear programming (MILP) solvers to enforce spatial and temporal feasibility, relying on synthesized virtual data derived from real-world statistics. (Tang et al., 2024) integrate LLMs with spatial optimization for urban itinerary planning. While these works demonstrate the power of hybrid systems for constraint satisfaction and feasibility guarantees, they primarily focus on explicitly stated requirements or synthesized data.

2.3 User Modeling and Personalization

User modeling constructs comprehensive user profiles by aggregating multi-dimensional user data (e.g., behavioral patterns, interests, and preferences). The integration of large language models (LLMs) with these profiles enables more personalized user experiences. For instance, recommendation engines utilize behavioral histories and content analysis to generate highly targeted suggestions (Liao et al., 2024), while chatbot systems employ adaptive mechanisms to refine linguistic patterns dynamically during interactions (Li and Zhao, 2021). Furthermore, multi-modal systems enhance the relevance of generated text and images by leveraging user profiles (Xu et al., 2024).

In the domain of travel planning, privacy concerns often limit the availability of real-world user behavior data, thereby posing a substantial challenge to effective user profiling. As a result, existing approaches mainly rely on synthetically generated virtual user profiles to improve personalization. For example, (Jiang et al., 2024) propose a dialogue-driven preference extraction agent, while

(Singh et al., 2024; Chen et al., 2024a) adopt LLM-synthesized virtual personas for personalized travel planning. Nevertheless, these methods are inherently limited by their reliance on artificially constructed persona data, which may not accurately reflect authentic user preferences in real-world scenarios.

3 Personal Travel Solver

3.1 RealTravel Dataset

To address the gap in capturing real-world users' implicit travel preferences, we constructed a novel dataset derived from authentic user reviews of points of interest (POIs) obtained from Google Local.

Specifically, we extracted POI metadata and user reviews from the Google Local dataset (Yan et al., 2023), focusing on cities included in the TravelPlanner benchmark. The Google Local dataset comprises business listings and review data from Google Maps, with coverage extending through September 2021 in the United States. From this dataset, we selected 1,155 users with the highest review activity, while excluding POIs with insufficient reviews and cities with inadequate POI coverage. We adopted the evaluation framework from TravelPlanner (Xie et al., 2024), along with associated flight and accommodation data.

Since the original TravelPlanner benchmark lacks symbolic grounding for user requests, we developed a travel request generator to produce structured symbolic queries. Each request was generated by sampling multiple variables (summarized in table 3), with constraint values randomly selected and validated via a verification program to ensure feasible solutions. These symbolic requests were subsequently converted into natural language descriptions using a large language model (LLM) via prompt-based generation. The final dataset consists of 1,000 test samples and 155 validation samples. For additional details on dataset, see Appendix A.

3.2 Overview of Personnel Travel Solver

Based on the RealTravel dataset, we propose Personal Travel Solver (PTS), a personalized travel planning system that integrates large language models (LLMs), the SCIP solver (Bestuzheva et al., 2021), and a structured data processing pipeline. It leverages explicit user requests and review history to extract implicit travel preferences, generating feasible, personalized travel plans.

PTS consists of five modular components working in concert:

(1) Translation Module: Converts natural language travel queries into a structured JSON representation, ensuring machine-readable input for downstream processing.

(2) Search Module: Retrieves candidate travel-related entries (e.g., attractions, accommodations) from the database based on user-specified query parameters.

(3) Preference Encode Module: Extracts latent user constraints and preferences by analyzing review histories, augmenting explicit input with inferred behavioral insights.

(4) Re-rank Module: Applies secondary filtering to refine candidate selections, producing a high-quality subset for optimization.

(5) Planning Module: Formulates travel planning as a constraint satisfaction problem, mapping structured user inputs into decision variables and constraints. This module employs a numerical solver to generate optimal personalized plans that adhere to user-defined and inferred preferences.

3.3 Translator Module

The Translator module leverages a large language model (LLM) to convert natural language user requests into a structured symbolic JSON representation, facilitating seamless interoperability with the numerical solver.

During parsing, the LLM-generated JSON output is cross-referenced against predefined JSON tags derived from reference queries to ensure semantic fidelity between the extracted structured data and the original natural language input. Further details on the natural language-to-JSON methodology are provided in Appendix D.1.

3.4 Search Module

The Search Module queries the database to retrieve entries that satisfy the criteria specified in the structured JSON output generated by the Translator Module.

The module integrates several specialized search tools, including flight, restaurant, attraction, and hotel search functionalities. The corresponding database entry counts for each search category are provided in table 4.

By reducing the solution space during the preliminary search phase, the Search Module improves computational efficiency. The candidate set is subsequently forwarded to the Re-rank Module for further refinement before planning.

3.5 Preference Encode Module

The Preference Encode Module extracts implicit user constraints from review history and formats them into a structured preference representation. Since user-generated reviews are often unstructured and noisy, directly inferring preferences presents a significant challenge for LLMs. To address this, we adopt a dual-perspective approach that models user interests by identifying both positively and negatively evaluated features of POIs.

We first curate the input data by selecting each user's 25 most substantial reviews, excluding those shorter than 20 words to ensure adequate contextual information. This filtered corpus enables the LLM to identify specific POI features that generated explicit positive or negative reactions. However, initial extractions often include ephemeral factors like weather conditions or crowd levels—elements unrelated to a POI's inherent qualities. We therefore implement a refinement stage where the LLM filters out these transient attributes through targeted prompting.

The remaining preference tags reveal different individual variation across users. To create manageable yet expressive profiles, we apply two normalization procedures: prioritizing the five most frequently mentioned preferences in each category, and merging semantically equivalent tags (e.g., "overpriced" and "expensive") through LLM-assisted clustering. The resulting standardized preferences feed into a final profiling step, where the LLM synthesizes a summary of each user's travel priorities. These comprehensive profiles enable assessment of preference alignment in generated travel plans. The complete prompting strategy, including all filtering and normalization instructions, appears in Appendix D.

3.6 Re-rank Module

The Re-rank Module applies a secondary filtering step to produce a refined set of high-quality candidate points of interest (POIs). Specifically, we leverage a pre-trained sequential recommender model (e.g., SASRec (Kang and McAuley, 2018)) to identify the top POIs that best align with user preferences.

We adapted SASRec by removing the embedding and positional encoding layers, using it as an encoder to update the initialized user and POI embeddings. User embeddings were initialized by encoding "likes" and "dislikes" tags into 1024-dimensional vectors using the BGE text embedding model (Chen et al., 2024b), then reduced to 50 dimensions via PCA (Abdi and Williams, 2010). POI embeddings followed a similar process, based on the POI's name, description, and category. The modified SASRec architecture further refines these embeddings to capture nuanced user-POI interaction patterns. Finally, we compute cosine similarity between user and POI embeddings and retain the top 50 POIs as the candidate set for downstream planning.

3.7 Planning Module

The Planning Module generates travel plans based on candidate POIs filtered by other modules. It models the travel planning task as a constrained satisfiability problem and derives a solution that meets the constraints specified in the user's travel query.

The planning module considers the Explicit Request Constraint and Implicit Preference Constraint:

Explicit Request Constraint. The Explicit Requirement Constraint refers to constraints explicitly specified by users in their travel requests, such as the destination, departure time, and dietary requirements. In our framework, it includes the cuisine type, room type, and budget. These constraints are formalized as a system of equations:

$$\begin{cases} r_i \le \mathbb{I}[\text{cuisine}_i \in \mathcal{C}_u], & \forall i \in \mathcal{R} \\ h_i \le \mathbb{I}[\text{room}_i = \text{room}_u], & \forall i \in \mathcal{H} \\ C_{flight} = \sum_{i \in \mathcal{F}_1 \cup \mathcal{F}_2} p_i f_i \\ C_{hotel} = d \sum_{j \in \mathcal{H}} p_j h_j \\ C_{dining} = n \sum_{k \in \mathcal{R}} c_k r_k \\ C_{flight} + C_{hotel} + C_{dining} \le B \end{cases} \tag{1}$$

The first two constraints utilize binary variables r_i (restaurants) and h_i (hotels) to enforce selection rules that align with user preferences. The remaining equations decompose the total cost into three components: transportation (C_{flight}), accommodation (C_{hotel}) over d nights, and dining (C_{dining}) for n travelers, with their sum constrained by the budget B.

To ensure the generated travel plans adhere to commonsense requirements and practical considerations, we introduce the following fundamental constraints:

$$\begin{cases} \sum_{i\in\mathcal{F}_1} f_i = 1, \quad \sum_{i\in\mathcal{F}_2} f_i = 1, \quad \forall i \in \mathcal{F}_1 \cup \mathcal{F}_2 \\ \sum_{i\in\mathcal{H}} h_i = 1, \quad \forall i \in \mathcal{H} \\ h_i \leq \mathbb{I}[\text{occ}_i \geq n], \quad \forall i \in \mathcal{H} \\ \sum_{i\in\mathcal{R}} r_i = 3d, \quad \forall i \in \mathcal{R} \\ 2d \leq \sum_{j\in\mathcal{A}} a_j \leq 3d, \quad \forall i \in \mathcal{A} \end{cases} \tag{2}$$

The constraints ensure the travel plan meets fundamental practical requirements through several mechanisms. For transportation, the binary variables $f_i \in \{0,1\}$ (where $i \in \mathcal{F}_1$ for outbound flights and $i \in \mathcal{F}_2$ for return flights) guarantee complete round-trip travel by requiring exactly one selection from each flight set. Accommodation requirements are handled through hotel selection variables $h_i \in \{0,1\}$ ($i \in \mathcal{H}$), ensuring the chosen hotel's occupancy capacity occ_i matches the group size n. Dining needs are addressed via restaurant selection variables $r_i \in \{0,1\}$ ($i \in \mathcal{R}$), with the constraint $\sum r_i = 3d$ scheduling three meals per day throughout the d-day trip. For activities, attraction selection variables $a_j \in \{0,1\}$ ($j \in \mathcal{A}$) maintain balanced itineraries by limiting daily visits to 2-3 points of interest. Together, these binary selection variables (f_i for flights, h_i for hotels, r_i for restaurants, and a_j for attractions) completely define the travel itinerary while satisfying all commonsense constraints.

Implicit Preference Constraint. The Implicit User Preference Constraint captures users' latent preferences that are not explicitly stated in their travel requests. In our framework, these preferences are inferred from users' expressed positive or negative responses toward points of interest (POIs), derived from associated characteristic tags. For each candidate attraction $j \in \mathcal{A}$, we introduce an integer-valued score variable s_j to quantify its preference alignment.

The scoring metric incorporates a dual-component weighting scheme: liked attributes ($\mathcal{L}$) contribute 10 points each, while avoiding disliked attributes ($\mathcal{D}$) contributes 20 points each. This weighting strategy reflects that avoiding negative experiences is often more critical for user satisfaction than simply including preferred features, as negative experiences can significantly impact the overall quality of the trip.

The preference score s for each attraction can be formulated as:

$$\text{score}_j = 10 \sum_{k\in\mathcal{L}} \text{attr}_{j,k} + 20 \sum_{k\in\mathcal{D}} (1 - \text{attr}_{j,k}) \tag{3}$$

where $\text{attr}_{j,k}$ denotes a binary indicator indicating whether attraction j possesses attribute k. The value of $\text{attr}_{j,k}$ is determined by the LLM through a comprehensive analysis of the attraction's metadata and the pros and cons extracted by the preference extraction module. Details of the prompts used are presented in the appendix.

To ensure preference scores are only considered for selected POIs, we apply the Big M method (Cococcioni and Fiaschi, 2021), a common optimization technique for enforcing logical conditions through linear constraints. In this case, the preference score of an attraction is counted only if it is included in the travel plan:

$$\begin{cases} s_j \leq \text{score}_j + M(1 - a_j), \forall j \in \mathcal{A} \\ s_j \geq \text{score}_j - M(1 - a_j), \forall j \in \mathcal{A} \\ s_j \leq M a_j, \forall j \in \mathcal{A} \\ s_j \geq 0, \forall j \in \mathcal{A} \end{cases} \tag{4}$$

where M represents a sufficiently large constant (set to 10000 in implementation), a_j is a binary decision variable indicating the selection of attraction j, and s_j represents the actual score assigned to attraction j. This constraint set ensures that when an attraction is selected ($a_j = 1$), its score equals the theoretical preference score, and when not selected ($a_j = 0$), its score is zero.

Optimization objective The optimization objective combines cost management with preference satisfaction by minimizing the deviation from budget while maximizing preference score:

$$\min \underbrace{C_{\text{total}} - B}_{\text{Cost Deviation}} - \underbrace{\sum_{j\in\mathcal{A}} s_j}_{\text{Preference Score}} \tag{5}$$

where C_{total} represents the cost of the plan, B is the budget in travel request, s_j denotes the preference score of attraction j. This mixed optimization objective enables our system to generate travel plans that not only satisfy explicit constraints but also align with users' implicit preferences, enabling the generation of travel plans that maintain financial feasibility while optimizing for personal user satisfaction.

3.8 LLM as a Judge

We employ the LLM as a judge framework (Singh et al., 2024) to assess the degree of personalization in different travel plans. Prior research has demonstrated that this approach effectively approximates human evaluative preference (Dong et al., 2024; Senel et al., 2024; Zheng et al., 2023). In our evaluation, the LLM judge is presented with pairs of travel plans alongside user profiles, which are aggregated from liked and disliked characteristics derived from the user's review history.

The LLM judge then determines which plan aligns more closely with the user's preference and need. This evaluation serves to assess the efficacy of the LLM in generating personalized travel plans. Our findings indicate that travel plans generated by our proposed method achieve a higher level of personalization compared to those produced directly by the large language model. Further details regarding the evaluation methodology are provided in Appendix B.

4 Evaluation

4.1 Evaluation Strategies

In this study, we evaluate the quality of the generated travel plans from two perspectives:

Explicit Constraint Satisfaction We assess how well the generated plans adhere to constraints explicitly stated in the user's query (**Hard Constraints**) and those derived from real-world common sense (**Commonsense Constraints**). Our evaluation methodology is adapted from the TravelPlanner benchmark (Xie et al., 2024). Key metrics include: Delivery Rate, Commonsense Constraint Pass Rate, Hard Constraint Pass Rate and Final Pass Rate. Detailed descriptions of each metric are provided in Appendix B.

Implicit Preference Alignment. To evaluate how well generated plans conform to users' underlying preferences, we assess their alignment with preferences inferred from user review histories. This evaluation employs the "LLM as a Judge" methodology (Singh et al., 2024). In this approach, an Large Language Model (LLM) acts as an evaluator, performing pairwise comparisons of plans based on a summarized user profile constructed from their past reviews. The primary metric is the **Preference Rate**, which indicates the frequency with which plans generated by our model are judged as more aligned with the user's inferred preferences compared to alternatives. A detailed description of this evaluation methodology can be found in Appendix B.5.

4.2 Baselines

Greedy Search To evaluate the performance of rule-based strategy, we implement a greedy search algorithm. This approach identifies flights and points of interest (POIs) with the minimal costs. Additional details are provided in Appendix E.

Planning Strategies To evaluate the effectiveness of LLM planning strategies, we assess four prominent approaches: **Direct**, **CoT** (Wei et al., 2022), **ReAct** (Yao et al., 2022), and **Reflexion** (Shinn et al., 2024). For more details, please refer to Appendix E.

4.3 Explicit Preference Evaluation Results

In this section, we evaluate the effectiveness of different planning strategies on RealTravel. In our experiments, we used the GPT-4o model with the temperature parameter set to 0, ensuring deterministic outputs. The experiments were conducted on both the validation and test sets. For numerical solver, we utilized the SCIP solver (Bestuzheva et al., 2021). The table 1 provides a detailed comparison of performance in terms of the constraint pass rate between direct planning using the LLM and our proposed method.

The results reveal fundamental limitations in directly using LLM for travel planning. While achieving near-perfect delivery rates (98.9-99.35%), these baselines fail to deliver the feasible plan with only 0.65% in the final pass rate. It shows that current agents still struggle with complex planning in real-world cases. Even given all the necessary information, these planning strategies still fail to generate a feasible plan. The agent failed to achieve a high macro-level pass rate. Although perform well in micro commonsense pass rate, their macro-level performance remains low. This discrepancy indicates that LLM-generated plans, though syntactically correct, frequently overlook explicit and implicit constraints inherent in user queries and candidate information.

Methods	Delivery Rate	Commonsense Pass Rate		Hard Constraint Pass Rate		Final Pass Rate
		Micro	Macro	Micro	Macro	
Valid Split (#155)						
Greedy	100.00	75.00	0.00	28.57	0.00	0.00
Direct	100.00	85.08	10.32	23.96	10.32	0.65
CoT	100.00	72.42	3.23	30.97	32.90	0.00
ReAct	99.35	74.27	0.65	34.65	46.45	0.00
Reflexion	92.26	53.63	0.00	26.36	0.00	0.00
Ours	100.00	100.00	100.00	99.61	98.06	98.06
Test Split (#1000)						
Greedy	100.00	75.00	0.00	28.71	0.00	0.00
Direct	100.00	84.18	13.00	23.37	8.40	0.40
CoT	100.00	72.20	1.40	30.70	31.40	0.10
ReAct	98.90	74.28	0.50	35.76	54.80	0.30
Reflexion	92.40	52.55	0.00	26.46	0.50	0.00
Ours	100.00	100.00	100.00	99.32	96.60	96.60

Table 1: Performance comparison of different planning strategies on the RealTravel dataset, detailing constraint satisfaction rates across validation and test splits.

Constraint Type	Greedy	Direct	COT	React	Reflexion	Ours
Commonsense Constraint						
Within Sandbox	100	69.6	26.2	6.5	2.7	100
Complete Information	100	35.7	21.2	44.4	6.2	100
Within Current City	100	99.3	91.2	93.4	79.3	100
Diverse Restaurants	0.0	72.0	55.6	80.0	60.7	100
Diverse Attractions	100	98.6	96.3	97.7	92.3	100
Hard Constraint						
Budget	100	23.8	67.7	75.3	92.3	100
Cuisine	1.0	39.8	47.2	76.1	0.5	96.6
Room Type	100	100.0	100.0	98.9	92.4	100

Table 2: Detailed pass rates for specific commonsense and hard constraint types, comparing the performance of different planning methods on the RealTravel test split.

In contrast, our method demonstrates substantial improvements across all metrics, achieving a near-optimal final pass rate. These results indicate that when large language models successfully formalize natural language requirements into logical constraints, their integration with numerical solvers can effectively address real-world constraint satisfaction problems. To further dissect these results, Table 2 provides a more granular breakdown of the pass rates for specific sub-categories of Commonsense and Hard constraints, offering a clearer view of how each method performs against individual constraint types.

4.4 Implicit Preference Evaluation Result

The evaluation results of the preference rate in the personalized setting are presented in the table, with DeepSeek-V3 serving as the judge. Experiments were conducted on both the validation and test sets. The results indicate that both the re-rank module and preference encode module in our method contribute to improving the personalization level in the generated plans.

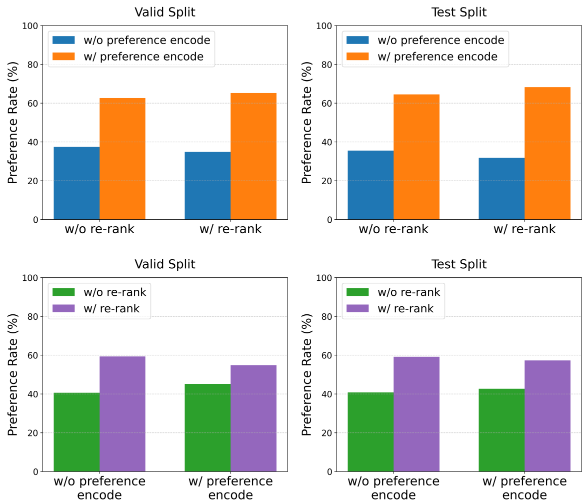

Figure 2: Impact of re-rank module and preference encode module on preference rate for both validation and test split. The results demonstrate that incorporating preference encode module and the re-rank module improve the preference rate, indicating better alignment with user preference.

4.5 Case Study

To illustrate how our system effectively incorporates user preference into travel planning, we present a detailed case study comparing two candidate plans generated for a user traveling from Nashville to San Antonio. For more details, please refer to Appendix F.

5 Conclusion

In this paper, we introduced RealTravel, a novel dataset augmented with real user reviews and POI metadata from Google Local, enabling more realistic preference-driven travel planning research.

Building upon RealTravel, we proposed the Personal Travel Solver (PTS), an innovative integrated system that synergizes the capabilities of Large Language Models (LLMs) with numerical solvers. The PTS system employs a novel five-module architecture—Translator, Search, Preference Encode, Re-rank, and Planning—that seamlessly connects explicit constraint validation with implicit preference modeling.

The experimental results demonstrate the effectiveness of PTS, showcasing superior performance in constraint satisfaction compared to baseline methods and improvement in the level of personalization.

6 Limitations

The Personal Travel Solver (PTS) effectively generates personalized travel plans by integrating LLMs with numerical solvers. However, several key limitations persist.

First, while our system infers implicit user preferences from reviews, this approach is contingent upon the quality and representativeness of the data. Sparse or unrepresentative reviews may result in incomplete or inaccurate preference modeling, potentially undermining personalization efficacy.

Second, the current PTS framework generates plans based on static datasets, which fails to account for the highly dynamic nature of real-world travel scenarios. Critical factors—such as real-time fluctuations in flight and hotel availability, abrupt weather disruptions, or local traffic conditions—can significantly impact the feasibility of generated plans. At present, our system lacks mechanisms for real-time updates or dynamic replanning in response to such unpredictable variables.

Finally, although LLMs facilitate preliminary personalization assessments, human evaluations remain essential for comprehensively evaluate how well the generated plans align with user expectations. Future work should investigate the relationship between human feedback and automated evaluations to further refine personalization performance.

Acknowledgments

This research was supported by the National Science and Technology Major Project (Grant No. 92270114) and the National Natural Science Foundation of China (Grant Nos. 62476116 and 62302321). This research was also supported by the advanced computing resources provided by the Supercomputing Center of the USTC.

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, et al. 2021. The scip optimization suite 8.0. *arXiv preprint arXiv:2112.08872*.

Bernd Bohnet, Azade Nova, Aaron T Parisi, Kevin Swersky, Katayoon Goshvadi, Hanjun Dai, Dale Schuurmans, Noah Fiedel, and Hanie Sedghi. 2024. Exploring and benchmarking the planning capabilities of large language models. *arXiv preprint arXiv:2406.13094*.

Booking Holdings Inc. 2024. Booking.com | official site | the best hotels, flights, car rentals & accommodations. https://www.booking.com/. Accessed on August 5, 2024.

Aili Chen, Xuyang Ge, Ziquan Fu, Yanghua Xiao, and Jiangjie Chen. 2024a. Travelagent: An ai assistant for personalized travel planning. *arXiv preprint arXiv:2409.08069*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024b. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Yanan Chen, Ali Pesaranghader, Tanmana Sadhu, and Dong Hoon Yi. 2024c. Can we rely on llm agents to draft long-horizon plans? let's take travelplanner as an example. *arXiv preprint arXiv:2408.06318*.

Marco Cococcioni and Lorenzo Fiaschi. 2021. The big-m method with the numerical infinite m. *Optimization Letters*, 15(7):2455–2468.

Tomas de la Rosa, Sriram Gopalakrishnan, Alberto Pozanco, Zhen Zeng, and Daniel Borrajo. 2024. Trippal: Travel planning with guarantees by combining large language models and automated planners. *arXiv preprint arXiv:2406.10196*.

Shujie Deng, Honghua Dong, and Xujie Si. 2024. Enhancing and evaluating logical reasoning abilities of large language models. In *Proceedings of the ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

Yijiang River Dong, Tiancheng Hu, and Nigel Collier. 2024. Can llm be a personalized judge? *arXiv preprint arXiv:2406.11657*.

Expedia Group. 2024. Expedia travel: Vacation homes, hotels, car rentals, flights & more. https://www.expedia.com/. Accessed on August 5, 2024.

Atharva Gundawar, Mudit Verma, Lin Guan, Karthik Valmeekam, Siddhant Bhambri, and Subbarao Kambhampati. 2024. Robust planning with llm-modulo framework: Case study in travel planning. *arXiv preprint arXiv:2405.20625*.

Sajal Halder, Kwan Hui Lim, Jeffrey Chan, and Xiuzhen Zhang. 2024. A survey on personalized itinerary recommendation: From optimisation to deep learning. *Applied Soft Computing*, 152:111200.

Yilun Hao, Yongchao Chen, Yang Zhang, and Chuchu Fan. 2024. Large language models can plan your travels rigorously with formal verification tools. *arXiv preprint arXiv:2404.11891*.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2022b. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Song Jiang, Da JU, Andrew Cohen, Sasha Mitts, Aaron Foss, Justine T Kao, Xian Li, and Yuandong Tian. 2024. Towards full delegation: Designing ideal agentic behaviors for travel planning. *arXiv preprint arXiv:2411.13904*.

Da Ju, Song Jiang, Andrew Cohen, Aaron Foss, Sasha Mitts, Arman Zharmagambetov, Brandon Amos, Xian Li, Justine T Kao, Maryam Fazel-Zarandi, and Yuandong Tian. 2024. To the globe (TTG): Towards language-driven guaranteed travel planning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 240–249, Miami, Florida, USA. Association for Computational Linguistics.

Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, pages 197–206. IEEE.

Kuang-Huei Lee, Ian Fischer, Yueh-Hua Wu, Dave Marwood, Shumeet Baluja, Dale Schuurmans, and Xinyun Chen. 2025. Evolving deeper llm thinking. *arXiv preprint arXiv:2501.09891*.

Jiacheng Li, Jingbo Shang, and Julian McAuley. 2022. Uctopic: Unsupervised contrastive learning for phrase representations and topic mining. *arXiv preprint arXiv:2202.13469*.

Sheng Li and Handong Zhao. 2021. A survey on representation learning for user modeling. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 4997–5003.

Jiayi Liao, Sihang Li, Zhengyi Yang, Jiancan Wu, Yancheng Yuan, Xiang Wang, and Xiangnan He. 2024. Llara: Large language-recommendation assistant. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1785–1795.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. 2023. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 3806–3824.

Roadtrippers, LLC. 2024. Road trip planner – build your itinerary and find the best stops. https://roadtrippers.com/. Accessed on August 5, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551.

Lütfi Kerem Senel, Besnik Fetahu, Davis Yoshida, Zhiyu Chen, Giuseppe Castellucci, Nikhita Vedula, Jason Choi, and Shervin Malmasi. 2024. Generative explore-exploit: Training-free optimization of generative recommender systems using llm optimizers. *arXiv preprint arXiv:2406.05255*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36.

Tom Silver, Soham Dan, Kavitha Srinivas, Joshua B. Tenenbaum, Leslie Kaelbling, and Michael Katz. 2024. Generalized planning in pddl domains with pretrained large language models. In *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'24/IAAI'24/EAAI'24. AAAI Press.

Harmanpreet Singh, Nikhil Verma, Yixiao Wang, Manasa Bharadwaj, Homa Fashandi, Kevin Ferreira, and Chul Lee. 2024. Personal large language model agents: A case study on tailored travel planning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 486–514.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Yihong Tang, Zhaokai Wang, Ao Qu, Yihao Yan, Kebing Hou, Dingyi Zhuang, Xiaotong Guo, Jinhua Zhao, Zhan Zhao, and Wei Ma. 2024. Synergizing spatial optimization with large language models for open-domain urban itinerary planning. *arXiv preprint arXiv:2402.07204*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Chengxing Xie and Difan Zou. 2024. A human-like reasoning framework for multi-phases planning task with large language models. *arXiv preprint arXiv:2405.18208*.

Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and Yu Su. 2024. Travelplanner: A benchmark for real-world planning with language agents. In *Forty-first International Conference on Machine Learning*.

Yiyan Xu, Wenjie Wang, Fuli Feng, Yunshan Ma, Jizhi Zhang, and Xiangnan He. 2024. Difashion: Towards personalized outfit generation. *arXiv preprint arXiv:2402.17279*.

An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. 2023. Personalized showcases: Generating multi-modal explanations for recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2251–2255.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V Le, Ed H Chi, et al. 2024. Natural plan: Benchmarking llms on natural language planning. *arXiv preprint arXiv:2406.04520*.

Zhi Zheng, Zhaopeng Qiu, Xiao Hu, Likang Wu, Hengshu Zhu, and Hui Xiong. 2023. Generative job recommendations with large language model. *arXiv preprint arXiv:2307.02157*.

A RealTravel

This appendix provides supplementary information regarding the composition of the RealTravel dataset. Table 3 outlines the various factors and constraints considered during the generation of travel requests within the dataset. Table 4 details the scale of the database used in this study, enumerating the number of entries available for different categories such as cities, flights, restaurants, attractions, and accommodations.

Table 3: Factors considered in travel request generation.

Item	Description
Airline Constraints	location, date, price.
Hotel Constraints	room type, price.
Restaurant Constraints	cuisines, price.
Budget Constraints	total budget.

Table 4: The number of entries in the database.

Tool	Data Entries (#)
City	77
Flight	3, 827, 361
Restaurant	15, 738
Attraction	12, 869
Accommodation	5, 064

B Evaluation Metrics

In this section, we present the definition of the evaluation metrics used in our experiments.

B.1 Delivery Rate

It measures the percentage of queries for which the agent successfully generates a final plan within predefined operational limits (e.g., 30 interaction steps or avoiding fatal error loops). If the agent fails to generate a plan for a given query, it receives a score of 0 for delivery rate and all subsequent evaluation metrics associated with that query.

B.2 Commonsense Constraint Pass Rate

This metric evaluates adherence to implicit real-world travel logic. Validation of the generated plan against these commonsense constraints considers the following aspects:

- **Within Sandbox**: All information incorporated into the plan must originate from the predefined, closed sandbox environment. Any information external to this sandbox is classified as a hallucination.

- **Complete Information**: The plan must not omit critical information. For instance, a failure to include accommodation arrangements for the duration of the travel would constitute an incompleteness.
- **Within Current City**: All scheduled activities for a given day must be located within the designated city for that particular day.
- **Diverse Restaurants**: Selections for dining establishments should exhibit variety and avoid repetition throughout the entirety of the trip.
- **Diverse Attractions**: Choices of attractions should not be duplicated across the travel itinerary.

B.3 Hard Constraint Pass Rate

This metric measures the fidelity of the generated plan to all constraints explicitly stipulated within the user's query. The checks include:

- **Budget**: The total budget of the trip.
- **Room Type**: Room types include "Entire Room", "Private Room".
- **Cuisine**: Cuisines include "American Cuisine", "Latin American Cuisine", "Eastern Asian Cuisine", "Eastern Asian Cuisine", "Chinese Cuisine", "African & Middle Eastern Cuisine", "Seafood & Organic Cuisine" and "European Cuisine".

B.4 Final Pass Rate

The Final Pass Rate denotes the proportion of generated plans that concurrently satisfy *all* Commonsense and Hard constraint checks.

B.5 Micro and Macro Pass Rates

To provide a comprehensive understanding of constraint satisfaction, both Commonsense and Hard Constraint Pass Rates are analyzed from two perspectives:

Micro Pass Rate: This metric calculates the aggregate percentage of *individual constraint checks* successfully passed across the corpus of *all* generated plans, considering *all* constraints within each plan. It offers a detailed assessment of performance with respect to specific constraint types.

$$\text{Micro Pass Rate} = \frac{\sum_{p \in P} \sum_{c \in C_p} \text{passed}(c, p)}{\sum_{p \in P} |C_p|}$$

where P represents the set of all plans under evaluation, C_p denotes the set of constraints pertinent to a specific plan p in P, and $\text{passed}(X, Y)$ is a binary function indicating whether plan Y satisfies constraints X.

Macro Pass Rate: This metric computes the percentage of *plans* wherein *all* applicable constraints of a specific type (i.e., Commonsense or Hard) were satisfied. It assesses the agent's holistic capability to meet all relevant constraints for an individual planning request.

$$\text{Macro Pass Rate} = \frac{\sum_{p \in P} \text{passed}(C_p, p)}{|P|}$$

where P represents the set of all plans being evaluated, C_p denotes the set of constraints pertinent to a specific plan p in P, and $\text{passed}(X, Y)$ is a binary function determining whether Y satisfies all constraints within the set X.

Preference Rate

The Preference Rate assesses the alignment of generated plans with implicit user preferences, employing the "LLM-as-Judge" paradigm (Singh et al., 2024). The evaluation process is as follows:

1. A user profile captures preferences (e.g., likes and dislikes) is generated from review history.
2. For a given user query, two distinct plans, generated by different methods, are presented to an LLM judge (e.g., DeepSeek-V3).
3. The LLM judge is instructed to assessing which plan more closely aligns with the user's preferences.
4. The Preference Rate is the percentage of comparisons where a plan was judged as better than the plan generated by other method.

C Example input queries and output plans

In this section, we provide an example of an input query together with its associated output plans.

Input query:
User ID: 101555027339334930973. Please create a travel plan for me where I'll be departing from Grand Rapids and heading to Atlanta for a 3-day trip from 2022-03-18 to 2022-03-20, 2022. I need a Private room for accommodation. I'd like to have Latin American Cuisine and European Cuisine options for dining. Can you help me keep this journey within a budget of 1500?
Corresponding output plan:
{
"days": 1,
"current_city": "from Grand Rapids to Atlanta",
"transportation": "Flight Number: F3523983, from Grand Rapids to Atlanta, Dep: 12:31, Arr: 14:28",
"breakfast": "Fireside Jamaican Restaurant, Atlanta",
"attraction": "Lionel Hampton-Beecher Hills Park, Atlanta; Six Springs Wetlands, Atlanta ;Cathedral of Christ the King, Atlanta;",
"lunch": "Porto Brasil Restaurant, Atlanta",
"dinner": "La Urbana Mexican Restaurant, Atlanta",
"accommodation": "Spacious private room close St. Barnabas Hospital, Atlanta"
},
{
"days": 2,
"current_city": "Atlanta",
"transportation": "-",
"breakfast": "Nyamminz & Jamminz Jamaican Restaurant, Atlanta",
"attraction": "The Physical Post, Atlanta;CNN Atlanta, Atlanta;Peachtree Hills Park, Atlanta;",
"lunch": "La Fiesta Mexican Restaurant, Atlanta",
"dinner": "Foxx Original Jamaican Restaurant, Atlanta",
"accommodation": "Spacious private room close St. Barnabas Hospital, Atlanta"
},
{
"days": 3,
"current_city": "from Atlanta to Grand Rapids",
"transportation": "Flight Number: F3528969, from Atlanta to Grand Rapids, Dep: 17:15, Arr: 18:58",
"breakfast": "El Mexicano Restaurant, Atlanta",
"attraction": "MODA, Atlanta;Sidewalk Scooter, Atlanta;Binders Disco Mural, Atlanta;",
"lunch": "Do Restaurant, Atlanta",
"dinner": "La Tapatia Mexican Restaurant, Atlanta",
"accommodation": "-"
}

D Prompts

D.1 Prompt for query to JSON

```
Please generate structured JSON data
    ↪ based on the following user
    ↪ requirements:
{query}

The generated JSON data should contain
    ↪ the following fields:
- budget: Budget (e.g., 1000)
- date: Date range (e.g., ["2023-01-01",
    ↪ "2023-01-05"])
- days: Number of days (e.g., 5)
- dest: Destination (e.g., "New York")
- local_constraint: Local constraints
    ↪ (e.g., {{'cuisine': ["American
    ↪ Cuisine", "Latin American
    ↪ Cuisine", "Eastern Asian
    ↪ Cuisine", "Chinese Cuisine",
    ↪ "African & Middle Eastern
    ↪ Cuisine", "European Cuisine",
    ↪ "Seafood & Organic Cuisine"],
    ↪ 'house rule': None, 'room type':
    ↪ 'Private room', 'transportation':
    ↪ None}})
- org: Departure location (e.g., "Los
    ↪ Angeles")

Here are some examples of how to convert
    ↪ natural language to JSON:

 Example 1:
Input: "I want to travel from Los Angeles
    ↪ to New York for 5 days from
    ↪ January 1st to January 5th with a
    ↪ budget of 1000. I prefer American
    ↪ Cuisine and Latin American
    ↪ Cuisine. "
Output:
{{
    "budget": 1000,
    "date": ["2023-01-01", "2023-01-05"],
    "days": 5,
    "dest": "New York",
    "local_constraint": {{'house rule':
    ↪ None, 'room type': None,
```

```
'cuisine': ['American Cuisine', 'Latin American Cuisine'], 'transportation': None}},
    "org": "Los Angeles"
}}

Example 2:
Input: "Plan a 3-day trip from Chicago to Miami starting on March 15th with a budget of 800. I need a private room and prefer American cuisine. "
Output:
{{
    "budget": 800,
    "date": ["2023-03-15", "2023-03-17"],
    "days": 3,
    "dest": "Miami",
    "local_constraint": {{'house rule': None, 'cuisine': ['American Cuisine'], 'room type': 'Private room', 'transportation': None}},
    "org": "Chicago"
}}

Now, please generate the JSON data for the following input:
{query}
```

D.2 Prompt for extracting likes and dislikes from review

```
"Given a review written by a user, list the preferences the user liked and disliked about the attraction under [Like] and [Dislike] in bullet points, respectively. "
"If there is nothing to mention about like/dislike, simply write 'None' under the corresponding tag. "
"DO NOT write any content that is not revealed in the review. Please do not repeat the expressions in the original text, but use one or more words to describe the characteristics of the attractions that the user is interested in. \n"
"Analyze user reviews of attractions through these lenses:\n"
"Attraction type, Cultural Value, Facilities & Services, Activities\n"

"List preferences under [Like]/[Dislike] using these strict criteria:\n"
"1. Focus on: Attraction Type, Cultural Value, Facilities & Services, On-site Activities\n"
"2. EXCLUDE: Transportation, weather, personal scheduling, or off-site locations\n"
"3. Require direct textual evidence in the review\n"
"4. Express characteristics as concise descriptors (1-3 words)\n\n"
```

```
"For EACH bullet point, validate:\n"
"- Directly concerns the attraction's core features/services\n"
"- Not affected by external/temporary factors\n"
"- Not about adjacent locations/activities outside attraction boundaries\n\n"

"If no valid aspects exist for a section, output 'None'. \n"

"Now, analyze the following review and extract meaningful likes and dislikes:\n"
"### Output Format:\n"
"[Like]\n"
"- Encapsulate the preferences the user liked in bullet points. \n"
"If no relevant likes found: None\n\n"
"[Dislike]\n"
"- Encapsulate the preferences the user disliked in bullet points. \n"
"If no relevant dislikes found: None\n\n"
f"Review: {review_text}\n"
```

D.3 Prompt for filter out tags

```
Given these user preference tags: {', '.join(batch)}

Please identify and filter out tags that are NOT relevant to describing features of a tourist attraction. Follow these updated rules:

1. **Keep tags that describe specific features, characteristics, or experiences of a tourist attraction. **
2. **Remove tags that are irrelevant to tourist attractions. **
3. **Return a JSON object with two keys:**
   - "relevant_tags": A list of tags that are relevant to tourist attractions.
   - "irrelevant_tags": A list of tags that are irrelevant and should be removed.
```

D.4 Prompt for generating user profile

```
f"User Preferences:\n"
f"Likes: {likes_text}\n"
f"Dislikes: {dislikes_text}\n\n"
"Based on these preferences, please generate a concise user profile. \n"
"Guidelines:\n"
"1. Focus on high-level characteristics. \n"
"2. Avoid directly repeating the preference lists; infer
```

```
    ↪ underlying traits. \n"
"3. Output in this format:\n"
"Profile: [user profile]\n"
```

D.5 Prompt for generating POI pro and con

```
You are a travel review expert. Please
    ↪ read the following user reviews
    ↪ for the attraction
    ↪ "{attraction_name}" .
Based on the reviews, please summarize
    ↪ the advantages and disadvantages
    ↪ of the attraction by considering
    ↪ the following aspects:
1. Environment (landscapes, cleanliness,
    ↪ etc. );
2. Facilities and Services (guided tours,
    ↪ rest areas, restrooms, staff
    ↪ service, etc. );
3. Accessibility and Convenience
    ↪ (location, parking, public
    ↪ transportation, etc. );
4. Activities and Experience (interactive
    ↪ features, historical/cultural
    ↪ aspects, entertainment, etc. );
5. Value for Money (ticket prices,
    ↪ overall experience, etc. ).
Please output your summary using the
    ↪ following format with special
    ↪ markers:
<PRO>: [Your summary of advantages]
<CON>: [Your summary of disadvantages]
Only output exactly as above (with the
    ↪ markers) and nothing else.
Here are some user reviews:
{reviews_text}
```

D.6 Prompt for selecting top likes and dislikes features

```
Given the following user preference tags
    ↪ after filtering irrelevant ones:
Likes: {', '. join(filtered_likes)}
Dislikes: {', '. join(filtered_dislikes)}

Please select the top 5 tags for likes
    ↪ and top 5 tags for dislikes that
    ↪ best represent the user's most
    ↪ important preferences regarding
    ↪ tourist attractions.
If there are fewer than 5 tags in a
    ↪ category, return all of them.
    Return a JSON object with two keys:
    ↪ "likes_top5" and "dislikes_top5".
```

D.7 Prompt for merging same label

```
Given these user preference tags: {', '.
    ↪ join(tags)}
```

```
Please standardize similar tags into
    ↪ consistent categories. Return a
    ↪ JSON dictionary mapping original
    ↪ tags to standardized tags.
```

D.8 Prompt for generating plans

```
You are a proficient planner. Based on
    ↪ the provided information and
    ↪ query, please give me a detailed
    ↪ plan, including specifics such as
    ↪ flight numbers (e.g., F0123456),
    ↪ restaurant names, and
    ↪ accommodation names. Note that
    ↪ all the information in your plan
    ↪ should be derived from the
    ↪ provided data. You must adhere to
    ↪ the format given in the example.
    ↪ Additionally, all details should
    ↪ align with commonsense. The
    ↪ symbol '-' indicates that
    ↪ information is unnecessary. For
    ↪ example, in the provided sample,
    ↪ you do not need to plan after
    ↪ returning to the departure city.
    ↪ When you travel to two cities in
    ↪ one day, you should note it in
    ↪ the 'Current City' section as in
    ↪ the example (i.e., from A to B).

***** Example *****
Query: Could you create a travel plan for
    ↪ 7 people from Ithaca to Charlotte
    ↪ spanning 3 days, from March 8th
    ↪ to March 14th, 2022, with a
    ↪ budget of 30, 200?
Travel Plan:
Day 1:
Current City: from Ithaca to Charlotte
Transportation: Flight Number: F3633413,
    ↪ from Ithaca to Charlotte
Breakfast: Nagaland's Kitchen, Charlotte
Attraction: The Charlotte Museum of
    ↪ History, Charlotte; Marion L. Fox
    ↪ Memorial Park, Charlotte; The Big
    ↪ Zoo, Charlotte.
Lunch: Cafe Maple Street, Charlotte
Dinner: Bombay Vada Pav, Charlotte
Accommodation: Affordable Spacious
    ↪ Refurbished Room in Bushwick!,
    ↪ Charlotte

Day 2:
Current City: Charlotte
Transportation: -
Breakfast: Olive Tree Cafe, Charlotte
Attraction: The Mint Museum, Charlotte;
    ↪ Romare Bearden Park, Charlotte;
    ↪ USS Bullhead Memorial Park,
    ↪ Charlotte.
Lunch: Birbal Ji Dhaba, Charlotte
Dinner: Pind Balluchi, Charlotte
Accommodation: Affordable Spacious
    ↪ Refurbished Room in Bushwick!,
    ↪ Charlotte
```

```
Day 3:
Current City: from Charlotte to Ithaca
Transportation: Flight Number: F3786167, from Charlotte to Ithaca
Breakfast: Subway, Charlotte
Attraction: Books Monument, Charlotte; The National WWII Museum, Charlotte; Tifft Nature Preserve, Charlotte.
Lunch: Olive Tree Cafe, Charlotte
Dinner: Kylin Skybar, Charlotte
Accommodation: -

***** Example Ends *****

Given information: {text}
Query: {query}
Travel Plan:
```

D.9 Prompt for LLM as judge

```
You are a travel itinerary evaluation assistant. Your task is to compare two travel plans and choose the one that better matches the user's preferences.

**User Profile:**
{user_profile if user_profile else "No specific profile provided. "}

----------------------------------------
**Plan 1 Attractions:**
{json. dumps(plan1, ensure_ascii=False, indent=2)}

**Plan 1 - Attraction List:**
{json. dumps(part_data1, ensure_ascii=False, indent=2)}

**Plan 1 - Detailed Attraction Info (from attractions.csv):**
{json. dumps(attraction_details_1, ensure_ascii=False, indent=2)}

**Plan 1 - Attraction Pros and Cons (from summary.json via gmap_id):**
{json. dumps(attraction_summaries_1, ensure_ascii=False, indent=2)}

----------------------------------------
**Plan 2 Attractions:**
{json. dumps(plan2, ensure_ascii=False, indent=2)}

**Plan 2 - Attraction List:**
{json. dumps(part_data2, ensure_ascii=False, indent=2)}

**Plan 2 - Detailed Attraction Info (from attractions.csv):**
{json. dumps(attraction_details_2, ensure_ascii=False, indent=2)}

**Plan 2 - Attraction Pros and Cons (from summary.json via gmap_id):**
{json. dumps(attraction_summaries_2, ensure_ascii=False, indent=2)}

----------------------------------------
Based on the above information, please evaluate the two plans and choose the one that better matches the user's preferences. Your answer should be one of the following:
- <choose plan 1> if Plan 1 is better.
- <choose plan 2> if Plan 2 is better.
```

D.10 Prompt for getting attribute

```
Based on the following comprehensive information about an attraction:

Name: {attraction_info.get('name')}
Description: {description}
Amenities: {', '.join(amenities)}
Accessibility: {', '.join(accessibility)}
Categories: {', '.join(categories)}

Additional Analysis:
Positive Aspects: {pros}
Negative Aspects: {cons}

Please analyze if this attraction satisfies the attribute "{attribute}". Consider both the general information and the detailed positive/negative aspects provided.

Required Response Format: Respond with exactly one word - either "True", "False", or "Uncertain", where:
  "True" means the attribute is definitely present
  "False" means the attribute is definitely not present
  "Uncertain" means there isn't enough information to make a determination

Evaluation Guidelines:
1. Consider both explicit mentions and implicit indicators in all provided information
2. Look for both supporting and contradicting evidence
3. If there's significant uncertainty, prefer "Uncertain" over making an assumption
```

E Baseline

We evaluate different planning strategies on Personal Travel Planning task:

- Greedy Search: This method selects the POIs and flights with the lowest cost.
- Direct: This strategy directly provides the

user's query and relevant POI information to the large model, instructing it to generate a travel plan that meets the user's requirements.

- CoT (Wei et al., 2022): Based on the Direct method, this approach prompts the LLM to "think step by step" to guides the model to articulate its deductive process, often leading to more robust results on complex tasks.
- ReAct (Yao et al., 2022): It integrates reasoning with actionable steps within an environment. It enables the LLM to interleave verbal reasoning traces ("Reason") with actions ("Act") such as querying an external tool, and then incorporating the resulting observations back into its reasoning loop, allowing dynamic interaction.
- Reflexion (Shinn et al., 2024): It leverages self-critique and refinement over multiple attempts. After an initial attempt, the model reflects on its own generated trajectory and output, identifies flaws or areas for improvement, and uses this self-generated feedback to guide subsequent, more refined attempts at solving the problem.

The specific prompts used for each planning strategy are provided in the appendix below.

E.1 Prompt for Direct

```
You are a proficient planner. Based on
    ↪ the provided information and
    ↪ query, please give me a detailed
    ↪ plan, including specifics such as
    ↪ flight numbers (e.g., F0123456),
    ↪ restaurant names, and
    ↪ accommodation names. Note that
    ↪ all the information in your plan
    ↪ should be derived from the
    ↪ provided data. You must adhere to
    ↪ the format given in the example.
    ↪ Additionally, all details should
    ↪ align with commonsense. The
    ↪ symbol '-' indicates that
    ↪ information is unnecessary. For
    ↪ example, in the provided sample,
    ↪ you do not need to plan after
    ↪ returning to the departure city.
    ↪ When you travel to two cities in
    ↪ one day, you should note it in
    ↪ the 'Current City' section as in
    ↪ the example (i.e., from A to B).

***** Example *****
Query: Could you create a travel plan for
    ↪ 7 people from Ithaca to Charlotte
    ↪ spanning 3 days, from March 8th
    ↪ to March 14th, 2022, with a
    ↪ budget of $30, 200?
Travel Plan:
Day 1:
Current City: from Ithaca to Charlotte
Transportation: Flight Number: F3633413,
    ↪ from Ithaca to Charlotte
Breakfast: Nagaland's Kitchen, Charlotte
Attraction: The Charlotte Museum of
    ↪ History, Charlotte; Marion L. Fox
    ↪ Memorial Park, Charlotte; The Big
    ↪ Zoo, Charlotte;
Lunch: Cafe Maple Street, Charlotte
Dinner: Bombay Vada Pav, Charlotte
Accommodation: Affordable Spacious
    ↪ Refurbished Room in Bushwick!,
    ↪ Charlotte

Day 2:
Current City: Charlotte
Transportation: -
Breakfast: Olive Tree Cafe, Charlotte
Attraction: The Mint Museum, Charlotte;
    ↪ Romare Bearden Park, Charlotte;
    ↪ USS Bullhead Memorial Park,
    ↪ Charlotte;
Lunch: Birbal Ji Dhaba, Charlotte
Dinner: Pind Balluchi, Charlotte
Accommodation: Affordable Spacious
    ↪ Refurbished Room in Bushwick!,
    ↪ Charlotte

Day 3:
Current City: from Charlotte to Ithaca
Transportation: Flight Number: F3786167,
    ↪ from Charlotte to Ithaca
Breakfast: Subway, Charlotte
Attraction: Books Monument, Charlotte;The
    ↪ National WWII Museum,
    ↪ Charlotte;Tifft Nature Preserve,
    ↪ Charlotte;
Lunch: Olive Tree Cafe, Charlotte
Dinner: Kylin Skybar, Charlotte
Accommodation: -
***** Example Ends *****

Given information: {text}
Query: {query}
Travel Plan:
```

E.2 Prompt for CoT

```
You are a proficient planner. Based on
    ↪ the provided information and
    ↪ query, please give me a detailed
    ↪ plan, including specifics such as
    ↪ flight numbers (e.g., F0123456),
    ↪ restaurant names, and hotel
    ↪ names. Note that all the
    ↪ information in your plan should
    ↪ be derived from the provided
    ↪ data. You must adhere to the
    ↪ format given in the example.
    ↪ Additionally, all details should
    ↪ align with common sense.
    ↪ Attraction visits and meals are
```

```
expected to be diverse. The symbol '-' indicates that information is unnecessary. For example, in the provided sample, you do not need to plan after returning to the departure city. When you travel to two cities in one day, you should note it in the 'Current City' section as in the example (i.e., from A to B).
***** Example *****
Query: Could you create a travel plan for 7 people from Ithaca to Charlotte spanning 3 days, from March 8th to March 14th, 2022, with a budget of $30, 200?
Travel Plan:
Day 1:
Current City: from Ithaca to Charlotte
Transportation: Flight Number: F3633413, from Ithaca to Charlotte
Breakfast: Nagaland's Kitchen, Charlotte
Attraction: The Charlotte Museum of History, Charlotte; Marion L. Fox Memorial Park, Charlotte; The Big Zoo, Charlotte;
Lunch: Cafe Maple Street, Charlotte
Dinner: Bombay Vada Pav, Charlotte
Accommodation: Affordable Spacious Refurbished Room in Bushwick!, Charlotte

Day 2:
Current City: Charlotte
Transportation: -
Breakfast: Olive Tree Cafe, Charlotte
Attraction: The Mint Museum, Charlotte; Romare Bearden Park, Charlotte; USS Bullhead Memorial Park, Charlotte;
Lunch: Birbal Ji Dhaba, Charlotte
Dinner: Pind Balluchi, Charlotte
Accommodation: Affordable Spacious Refurbished Room in Bushwick!, Charlotte

Day 3:
Current City: from Charlotte to Ithaca
Transportation: Flight Number: F3786167, from Charlotte to Ithaca
Breakfast: Subway, Charlotte
Attraction: Books Monument, Charlotte;The National WWII Museum, Charlotte;Tifft Nature Preserve, Charlotte;
Lunch: Olive Tree Cafe, Charlotte
Dinner: Kylin Skybar, Charlotte
Accommodation: -
***** Example Ends *****
Given information: {text}
Query: {query}
Travel Plan: Let's think step by step. First,
```

E.3 Prompt for ReAct

```
You are a proficient planner. Based on the provided information and query, please give me a detailed plan, including specifics such as flight numbers (e.g., F0123456), restaurant names, and hotel names. Note that all the information in your plan should be derived from the provided data. You must adhere to the format given in the example. Additionally, all details should align with common sense. Attraction visits and meals are expected to be diverse. The symbol '-' indicates that information is unnecessary. For example, in the provided sample, you do not need to plan after returning to the departure city. When you travel to two cities in one day, you should note it in the 'Current City' section as in the example (i.e., from A to B). Solve this task by alternating between Thought, Action, and Observation steps. The 'Thought' phase involves reasoning about the current situation. The 'Action' phase can be of two types:
(1) CostEnquiry[Sub Plan]: This function calculates the cost of a detailed sub plan, which you need to input the people number and plan in JSON format. The sub-plan should encompass a complete one-day plan. An example will be provided for reference.
(2) Finish[Final Plan]: Use this function to indicate the completion of the task. You must submit a final, complete plan as an argument.
***** Example *****
Query: Could you create a travel plan for 7 people from Ithaca to Charlotte spanning 3 days, from March 8th to March 14th, 2022, with a budget of $30, 200?
You can call CostEnquiry like CostEnquiry[{{"people_number": 7, "day": 1, "current_city": "from Ithaca to Charlotte", "transportation": "Flight Number: F3633413, from Ithaca to Charlotte", "breakfast": "Nagaland's Kitchen, Charlotte", "attraction": "The Charlotte Museum of History, Charlotte", "lunch": "Cafe Maple Street, Charlotte", "dinner": "Bombay Vada Pav, Charlotte", "accommodation": "Affordable Spacious Refurbished Room in Bushwick!, Charlotte"}}]
You can call Finish like Finish[Day: 1
```

```
Current City: from Ithaca to Charlotte
Transportation: Flight Number: F3633413,
     ↪ from Ithaca to Charlotte
Breakfast: Nagaland's Kitchen, Charlotte
Attraction: The Charlotte Museum of
     ↪ History, Charlotte; Marion L. Fox
     ↪ Memorial Park, Charlotte; The Big
     ↪ Zoo, Charlotte;
Lunch: Cafe Maple Street, Charlotte
Dinner: Bombay Vada Pav, Charlotte
Accommodation: Affordable Spacious
     ↪ Refurbished Room in Bushwick!,
     ↪ Charlotte
Day 2:
Current City: Charlotte
Transportation: -
Breakfast: Olive Tree Cafe, Charlotte
Attraction: The Mint Museum, Charlotte;
     ↪ Romare Bearden Park, Charlotte;
     ↪ USS Bullhead Memorial Park,
     ↪ Charlotte;
Lunch: Birbal Ji Dhaba, Charlotte
Dinner: Pind Balluchi, Charlotte
Accommodation: Affordable Spacious
     ↪ Refurbished Room in Bushwick!,
     ↪ Charlotte
Day 3:
Current City: from Charlotte to Ithaca
Transportation: Flight Number: F3786167,
     ↪ from Charlotte to Ithaca
Breakfast: Subway, Charlotte
Attraction: Books Monument, Charlotte;
     ↪ The National WWII Museum,
     ↪ Charlotte; Tifft Nature Preserve,
     ↪ Charlotte;
Lunch: Olive Tree Cafe, Charlotte
Dinner: Kylin Skybar, Charlotte
Accommodation: -]
***** Example Ends *****
{reflections}
You must use Finish to indicate you have
     ↪ finished the task. And each
     ↪ action only calls one function
     ↪ once.
Given information: {text}
Query: {query}{scratchpad}
```

E.4 Prompt for Reflexion

```
You are an advanced reasoning agent that
     ↪ can improve based on self
     ↪ reflection. You will be given a
     ↪ previous reasoning trial in which
     ↪ you were given access to an
     ↪ automatic cost calculation
     ↪ environment, a travel query to
     ↪ give a plan and relevant
     ↪ information. Only the selection
     ↪ whose name and city match the
     ↪ given information will be
     ↪ calculated correctly. You were
     ↪ unsuccessful in creating a plan
     ↪ because you used up your set
     ↪ number of reasoning steps. In a
     ↪ few sentences, Diagnose a
     ↪ possible reason for failure and
     ↪ devise a new, concise, high-level
     ↪ plan that aims to mitigate the
     ↪ same failure. Use complete
     ↪ sentences.
Given information: {text}
Previous trial:
Query: {query}{scratchpad}
Reflection:
```

F Case Study

To illustrate how our system effectively incorporates user preferences into travel planning, we present a detailed case study comparing two candidate plans generated for a user traveling from Nashville to San Antonio. "query": "Please create a travel plan for me where I'll be departing from Nashville and heading to San Antonio for a 3-day trip from 2022-03-01 to 2022-03-03, 2022. I need a Private room for accommodation. I'd like to have American Cuisine and Chinese Cuisine options for dining. Can you help me keep this journey within a budget of 1400?"

User Profile: This user values experiences that are rich in historical and cultural significance, with a strong preference for locations that offer educational value and family-friendly activities. They appreciate well-maintained, clean, and accessible facilities, particularly those that provide unique observation opportunities, such as 360-degree views or the ability to see the curvature of the Earth. The user enjoys attractions that blend seamlessly with their surroundings, whether it's an iconic skyline presence, natural landscapes, or culturally authentic settings. They favor destinations that offer diverse and engaging exhibits, from historical statues and memorials to interactive play areas and wildlife sightings. Additionally, they prefer sites that are less commercialized, providing a more genuine and immersive experience, and they have a particular fondness for venues that offer free admission, good food options, and relaxing environments. Their ideal location would be one that combines these elements with modern amenities, like air conditioning and on-site dining, while also offering memorable photo opportunities and a touch of architectural uniqueness.

The user's profile, constructed from their review history, revealed a strong appreciation for several key characteristics:

- Historical and cultural significance in attractions

- Educational value and learning opportunities
- Family-friendly environments
- Unique observational experiences
- Less commercialized, more authentic settings

This profile was automatically extracted by our Preference Encode Module through analysis of the user's previous reviews and ratings.

Plan 1 (Generated without preference integration):

```
days": 1,
current_city": "-",
transportation": "Flight Number:
    ↪ F3935015, from Nashville to San
    ↪ Antonio",
breakfast": "Los Reyes Mexican
    ↪ Restaurant, San Antonio",
lunch": "Wahkee Chinese Seafood
    ↪ Restaurant, San Antonio",
dinner": "Royal Cuisine - International
    ↪ Restaurant, San Antonio",
attraction": "St. Joseph Parish, San
    ↪ Antonio;Extreme Escape -
    ↪ Colonnade, San Antonio;Texas
    ↪ Transportation Museum, San
    ↪ Antonio;",
accommodation": "Accommodation Number
    ↪ 480, San Antonio"

days": 2,
current_city": "-",
transportation": "-",
breakfast": "Jimador Restaurant & Bar,
    ↪ San Antonio",
lunch": "Oaks Crossing Restaurant & Bar,
    ↪ San Antonio",
dinner": "Koi Kawa Japanese Restaurant,
    ↪ San Antonio",
attraction": "Galeria E.V.A., San
    ↪ Antonio;Morgan's Wonderland, San
    ↪ Antonio;San Fernando Cathedral,
    ↪ San Antonio;",
accommodation": "Accommodation Number
    ↪ 480, San Antonio"

days": 3,
current_city": "-",
transportation": "Flight Number:
    ↪ F3966925, from San Antonio to
    ↪ Nashville",
breakfast": "Mama Margie's Mexican
    ↪ Restaurant, San Antonio",
lunch": "Blanquita Mexican Restaurant,
    ↪ San Antonio",
dinner": "Thai Lucky Sushi Bar &
    ↪ Restaurant, San Antonio",
attraction": "Yanaguana Garden at
    ↪ Hemisfair, San Antonio;Fort Sam
    ↪ Houston, San Antonio;Terror On
    ↪ The Plaza Haunted House, San
    ↪ Antonio;",
```

```
accommodation": "-"
```

The first plan included attractions such as Terror On The Plaza Haunted House (rating: 3.9) and Extreme Escape - Colonnade. While these venues are popular tourist destinations, they represent more commercialized, entertainment-focused experiences that don't align with the user's documented preferences for historical and educational content.

Plan 2 (Generated with PTS):

```
"days": 1,
current_city": "from Nashville to San
    ↪ Antonio",
transportation": "Flight Number:
    ↪ F3935015, from Nashville to San
    ↪ Antonio, Dep: 07:44, Arr: 10:00",
breakfast": "Samurai Sushi Restaurant,
    ↪ San Antonio",
lunch": "Oaks Crossing Restaurant & Bar,
    ↪ San Antonio",
dinner": "Las Palapas Mexican Restaurant,
    ↪ San Antonio",
attraction": "Fort Craig National
    ↪ Historic Site, San
    ↪ Antonio;Mission San Jose Church,
    ↪ San Antonio;Natural Bridge
    ↪ Caverns, San Antonio;",
accommodation": "Williamsburg Gardens
    ↪ Flat, Large Room Private Bath,
    ↪ San Antonio"

days": 2,
current_city": "San Antonio",
transportation": "-",
breakfast": "Valentina's Mexican
    ↪ Restaurant, San Antonio",
lunch": "54th Street Restaurant &
    ↪ Drafthouse- The Rim, San Antonio",
dinner": "Hsiu Yu Chinese Restaurant, San
    ↪ Antonio",
attraction": "The Deco District, San
    ↪ Antonio;The Stray Grape Urban
    ↪ Winery, San Antonio;GO RIO San
    ↪ Antonio River Cruises, San
    ↪ Antonio;",
accommodation": "Williamsburg Gardens
    ↪ Flat, Large Room Private Bath,
    ↪ San Antonio"

days": 3,
current_city": "from San Antonio to
    ↪ Nashville",
transportation": "Flight Number:
    ↪ F3966926, from San Antonio to
    ↪ Nashville, Dep: 06:59, Arr:
    ↪ 09:03",
breakfast": "Restaurant Le Reve, San
    ↪ Antonio",
lunch": "Pearl Inn Chinese Restaurant,
    ↪ San Antonio",
```

```
dinner": "House of Joy Chinese
    ↪ Restaurant, San Antonio",
attraction": "Camden Street Riverwalk
    ↪ Bridge, San Antonio;Military
    ↪ Working Dog Teams National
    ↪ Monument, San Antonio;San Antonio
    ↪ Explorer Pass??, San Antonio;",
accommodation": "-"
```

In contrast, the second plan, generated by our preference-aware system, includes historically significant sites such as Mission San Jose Church (rating: 4.8) and Fort Craig National Historic Site (rating: 4.3). The plan also incorporates Natural Bridge Caverns, which offers unique geological observations and educational value, directly matching the user's interest in distinctive observational experiences.

G Hardware Setup

The hardware setup includes the following specifications:

- **Processor**: Intel(R) Xeon(R) Gold 6348
- **Memory**: 128GB DDR4 RAM
- **Graphics Card**: NVIDIA RTX3090
- **Operating System**: Ubuntu 20.04 LTS