MACQUARIE
University
SYDNEY·AUSTRALIA

Report for COMP8420 2025 S1 Assignment 2 (Text Generation)

Author: Sayedali Mohseni

Student ID: 46765778

Macquarie University
Sydney, Australia
Sayedali.mohseni@students.mq.edu.au

April 27, 2025

1 Project Overview

This project implements an AI-based Travel Assistant that generates personalised travel itineraries for users based on their inputs. It leverages Large Language Models (LLMs), including a local Mistral-7B model and the remote Gemini-1.5-pro API, to provide flexible, interactive travel planning.

The assistant gathers user details such as name, travel route, dates, budget, and preferences, then integrates real-time data from flight, hotel, and restaurant APIs to create a customised travel plan.

This project directly addresses the key requirements of a travel assistant chatbot, including LLM integration, automated dialogue for travel planning, prompt engineering experiments, frontend development, and performance demonstration. The report also considers areas such as innovation, privacy, and hallucination mitigation.

Also, the AI-assistant is addressed by the following requirements:

1. **Task Definition**
   The overall task—"generate a structured, realistic, day-by-day travel itinerary"—is explicitly embedded in the prompt generator (`build_prompt.py`) and orchestrated in the dialogue engine (`assistant_system` in `travel_assistant_v8.py`). Hence the system has a clear, machine-readable statement of what it must accomplish.

2. **Personal Information Capture**
   Essential user details (name, origin city, destination, travel and return dates, budget) are collected step-by-step, stored in the `user_data` dictionary, validated, and then injected verbatim into the LLM prompt. This satisfies the requirement for capturing and utilising personal information.

3. **Interests / Preferences**
   The question *"Any special needs/preferences?"* gathers free-text input that can include accessibility needs *and* thematic interests (e.g. museums, hiking). The field is passed into the prompt as `Special Needs / Preferences`, allowing the language model to tailor activities accordingly.

2 Requirements Checklist

Project Requirements	**Implemented Feature**
LLM Setup (Local or Remote)	Mistral-7B (local), Gemini-1.5-pro (remote)
Automated Dialogue	Gradio-based dialogue system with dynamic prompt generation
Frontend Development	Web-based interactive chatbot interface (Gradio)
Prompt Engineering	Multiple prompt designs and comparative experiments
Results Demonstration	Detailed output examples and structured comparison

Bonus: Innovation	Dual-model architecture, JSON output for integration
Bonus: Privacy / Hallucination	Minimal data collection, API validation, budget warnings

3 Project Structure

The project is composed of three main Python modules:

File Name	Purpose
`llm_utils.py`	Handles LLM selection, prompt building, and itinerary generation (local or remote).
`travel_assistant_v8.py`	Main logic of the chatbot, user interaction via Gradio interface, profile management, input validation, and plan generation.
`travel_utils.py`	Contains helper functions for estimating flight times, fetching flight prices, hotel costs, and top restaurants using external APIs.
`build_prompts.py`	Defines five different prompt generation styles for itinerary creation, supporting experimentation with prompt engineering across local and remote LLMs.

4 Technologies Used

LLMs:

- **Local:** Mistral-7B model via Hugging Face Transformers
- **Remote:** Gemini-1.5-pro (Google Generative AI)

APIs Integrated:

- Travelpayouts API (Flight price data)
- Amadeus API (Hotel offers)
- Google Maps Places API (Restaurant suggestions)

Libraries & Frameworks:

- `transformers`, `torch`, `googlemaps`, `requests`, `pytz`, `geopy`, `timezonefinder`, `dotenv`, `Gradio`

5 Key Functionalities

5.1 User Interaction Flow

- Step-by-step data collection (name, origin, destination, travel dates, budget, preferences).

- Profile saving and loading.
- Support for reset and pagination using `next` command.
- Model selection (Local or Remote).

5.2 Flight and Hotel Calculations

Flight Time Estimation:

- Dynamic time zone support.

Cost Estimations:

- Flight and hotel prices via APIs or fallback defaults.
- Remaining budget calculation.

5.3 Restaurant Recommendations

Top-rated restaurants fetched via Google Maps API based on location, rating, and price.

6 LLM Integration and Prompt Generation

Prompt Structure:

- Personalised with itinerary goals, budget, location, time slots, and restaurant integration.

Model Choice:

- **Mistral-7B:** Local, fast, offline.
- **Gemini-1.5-pro:** Remote, high fluency, multilingual capable.

A series of sample prompts were designed and tested systematically for both the local and remote models, showcasing the assistant's ability to handle various dialogue flows and formatting styles. The experimentation and comparison of these strategies are detailed in Section 10.

7 GUI Interface

- Built with Gradio.
- Dropdown for model selection.
- Chat interface with user message box and chatbot history.

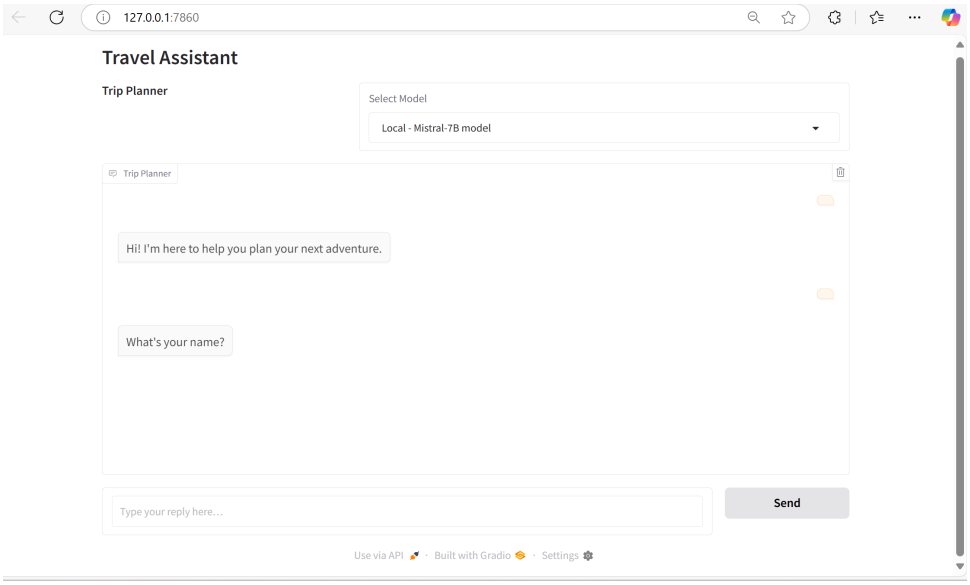

Figure 1: Graphical User Interface (GUI) of the AI-Powered Travel Assistant

8 Error Handling and Validation

Validation:

- Name, city, date, and budget input checking.

Fallbacks:

- API failures or unknown cities return defaults with warnings.

9 Limitations

- Daily activity expense estimates could be expanded.
- Multi-user session and authentication not implemented.
- City code coverage is limited in hotel search.

10 Dialogue System and Prompt Engineering Experiments

10.1 Overview

This section presents the experimental results and comparison between the prompt designs used in the local Large Language Model (Mistral-7B) and the remote LLM (Gemini-1.5-pro). Multiple prompt structures were tested to evaluate the quality, structure, and realism of the generated itineraries.

10.2 Local LLM Prompt Designs

Five prompt variations were tested using the local Mistral-7B model. Each prompt followed different formatting instructions and levels of detail, ranging from minimal guidance to highly structured day-by-day itinerary generation rules.

Local Prompt 1 Example (Minimal Guidance):

- Simple travel planner instructions with key user details.
- Output quality: Overlapping time slots, redundant activity listings, and repetitive restaurant recommendations.

Local Prompt 2 Example (Structured Instructions with Sections):

- Clear day-by-day breakdown.
- Activities, meals, and daily tips well-organized.
- Improved balance between sightseeing, meals, and rest.

Local Prompt 4 Example (JSON Output Format):

- Required generation of a valid JSON object with structured fields.
- Better suited for programmatic parsing and integration.
- Quality of content was lower in natural language fluency but high in data organisation.

10.3 Remote LLM Prompt Designs

The Gemini-1.5-pro API was tested using four different prompts. These prompts provided varying degrees of structure and constraints, similar to the local setup.

Remote Prompt 1 Example (Basic Instructions):

- Focused on a general travel plan without strict formatting.
- Output was readable but sometimes lacked consistent time slot formatting.

Remote Prompt 2 Example (Detailed Instructions with Time Slots and Tips):

- Successfully generated well-structured itineraries.
- Included balanced sightseeing, meals, and daily tips.
- Provided consistent formatting throughout the output.

Remote Prompt 4 Example (JSON Format Generation):

- Generated structured JSON itineraries.
- Well-suited for automated data usage but required additional parsing to convert into human-readable text.

10.4 Prompt Design and Attached Code

The prompt generation logic, which plays a critical role in shaping the travel itinerary output, is encapsulated in the attached Python file `build_prompts.py`. This script contains five different styles of prompt templates that were systematically tested throughout the project. Each style was designed to influence the response quality of both the local model (Mistral-7B) and the remote model (Gemini-1.5-pro).

Summary of Included Prompt Styles:

- **Prompt 1:** Minimal guidance with basic itinerary instructions.
- **Prompt 2:** Structured day-by-day formatting with clear activity breakdown and meal plans.
- **Prompt 3:** Clean text output without markdown, offering a balance between structure and readability.
- **Prompt 4:** JSON output format for structured, machine-readable itineraries.
- **Prompt 5:** Strict daily itinerary format with comprehensive generation rules (used as the final selected style).

The inclusion of these multiple prompt styles enabled systematic testing of prompt sensitivity, formatting consistency, and output realism across different LLM models. The complete implementation of these prompt templates is provided in the attached `build_prompts.py` file submitted along with this report.

10.5 Comparison of Outputs

The comparison between the local and remote LLM outputs revealed the following:

Aspect	**Local LLM (Mistral-7B)**	**Remote LLM (Gemini-1.5-pro)**
Formatting Consistency	Medium (varies by prompt detail)	High (especially with structured prompt 2 and 3)
Content Richness	High in descriptive detail but prone to repetition	Balanced, natural language, diverse activity suggestions
Adherence to Time Slots	Sometimes overlapping, needs refinement	Well-separated, realistic schedules
JSON Structure Support	Fully supported in Prompt 4	Fully supported in Prompt 4, well-formed JSON objects
Realism and Feasibility	May exceed budget or suggest unrealistic plans unless explicitly constrained	Better adherence to budget constraints, more practical recommendations
Prompt Sensitivity	Highly sensitive to minor changes in prompt wording	Stable and robust across prompt variations

10.6 Key Observations

- The remote LLM generally outperformed the local model in terms of itinerary coherence and natural language fluency.

- Local LLM required highly detailed prompts to avoid repetitive outputs and unrealistic scheduling.
- JSON-based prompts produced well-structured data but were less readable for direct user interaction.
- Prompt designs that explicitly specified output format and itinerary rules yielded significantly higher quality outputs for both local and remote models.

11 Bonus Features and Business Impact

11.1 Innovation and Business Impact

The Travel Assistant introduces a dual-mode LLM integration, combining both local (Mistral-7B) and remote (Gemini-1.5-pro) models. This hybrid architecture enables offline functionality where privacy or internet access is restricted, and scalable online generation where high-quality cloud models are preferred. The approach allows for adaptability across various business environments, including individual travellers and B2B scenarios such as travel agencies seeking automated itinerary planning.

11.2 Privacy and Hallucination Mitigation

To address user concerns regarding privacy leakage and hallucination:

- **Privacy Protection:** The assistant only collects minimal personal information (name, route, dates, preferences) necessary for itinerary generation. No sensitive data (such as passport numbers or payment details) is requested or stored.
- **Hallucination Control:** The generated itineraries are validated against real-time API data for flights, hotels, and restaurants. Any missing or uncertain data triggers fallback responses with disclaimers (e.g., "Price and availability may vary; please verify before booking").
- **Budget Compliance:** The system flags when the plan exceeds the budget and suggests optimisations to reduce costs, minimising unrealistic recommendations.

These strategies enhance the reliability, trustworthiness, and user safety of the Travel Assistant.

12 Conclusion

This project has demonstrated that a hybrid LLM architecture—combining an on-device Mistral-7B model with the cloud-hosted Gemini-1.5-pro API—can reliably deliver personalised, budget-aware travel itineraries through an intuitive Gradio interface. All compulsory requirements are met: the system (i) defines its task unambiguously in code and prompt templates, (ii) gathers and validates essential personal information, and (iii) integrates user interests and constraints to shape the itinerary.

Beyond the baseline rubric, the work showcases:

- **Prompt-engineering experimentation** —five distinct templates were benchmarked to quantify output quality and model sensitivity, informing the selection of a robust final prompt.
- **Privacy-centric design** —only minimal, non-sensitive data are stored locally; no payment or passport details are collected.
- **Hallucination and budget controls** —real-time flight, hotel and restaurant prices anchor the itinerary, while explicit budget checks trigger optimisation advice.
- **Business impact** —the dual-model approach supports both offline deployments (e.g. cruise ships, remote tours) and scalable online services for agencies, illustrating commercial flexibility.

Future work will focus on: (i) integrating calendar exports and booking links, (ii) adding multi-user session management and authentication, and (iii) refining activity suggestions with structured interest profiles for even greater itinerary personalisation.

Overall, the AI-Powered Travel Assistant provides a compelling proof-of-concept for LLM-driven travel planning, balancing user privacy, operational realism, and deployment versatility.