

Key Practical Points for AI Travel Planner (Technical Notes)

1. Feasibility Check for Unrealistic Budgets

- If user selects an international city with a very low budget (e.g., ■5,000 for Paris), the backend should detect mismatch before calling AI.

- Return a JSON error:

```
{"error": true, "message": "Budget too low for selected destination."}
```

- Flutter displays a modal: "Increase Budget" or "Choose Another City".

- Prevents wrong or hallucinated AI responses.

2. Hybrid Mode Logic

- If city is in the 5-city dataset → use CSV + AI for accuracy.

- If city is not in dataset → AI-only mode.

- Makes the system professional and realistic.

3. Strict JSON Enforcement

- All AI responses must follow fixed JSON schema.

- Backend must validate JSON before sending to Flutter.

- Reduces random output or formatting errors.

4. Travel-Only Restriction

- System prompt forces AI to answer ONLY travel questions.

- Non-travel questions return:

"I can only assist with travel-related questions."

5. UI-to-Backend Mapping

- City → "city"

- Days → "days"

- Budget → "budget"

- Interest Chips → "interests" list
- Description Box → "user_message"
- Generate Button → calls /plan_trip
- Chat Screen → calls /chat

6. Error Handling

- City empty → inline field error
- AI timeout → toast: "Response delayed, try again."
- Invalid JSON → "Partial results, please regenerate."
- Network error → retry modal

7. Professional System Behavior

- Never generate fake itineraries for impossible budgets.
- Always prefer precise, realistic, verifiable travel data.
- UI stays minimal: NO "Coming Soon" elements.
- Extra features only appear in PPT as "Future Scope".

These points make the project reliable, realistic, and professional without adding unnecessary complexity.