RouteForge: AI Research & Planning Agent — Nayab Irfan

1. Executive Summary

RouteForge is an AI travel planner built to handle real, on-the-go needs. Beyond finding attractions, it understands when a traveler wants a **specific stop**—for example, to quickly visit a **pharmacy**, grab **coffee**, pick up items from a **store**, or anything else they explicitly ask for—and **guarantees** that at least one such stop is included in the route. It blends real-time web search for fresh guides, place discovery from map sources, and efficient routing to produce a plan that balances **minimum cost** and **maximum enjoyment**. Even when premium APIs or LLMs rate-limit, the agent falls back to open data (OpenStreetMap + OSRM) and still delivers a complete Markdown itinerary and a JSON dataset with sources, coordinates, and costs.

2. Market / Topic Overview

Travel planning is fragmented across multiple tabs and apps: guides, maps, reviews, transport, and budget. This friction is worst for **time-boxed stopovers** (e.g., "I have 2 hours—find a pharmacy, a coffee, and one nearby sight"). RouteForge solves this by unifying the flow: - **Guides (recent):** Tavily for last-12-months articles and lists. - **Places:** SerpAPI (Google Maps) when key is available; otherwise OpenStreetMap (Overpass) POIs. - **Geocoding:** Nominatim (with viewbox bias for within-city names) + Photon fallback. - **Routing:** OSRM for distance/time matrices and practical multi-stop paths. - **Outputs:** Human-readable Markdown itinerary + machine-traceable JSON with URLs and coordinates. This stack works globally and degrades gracefully when paid services are unavailable.

3. Innovation / Trend Highlights

- **Chatbot-style intent → concrete POIs:** The agent parses free-form user text (e.g., "add a pharmacy and a coffee near the museum") into structured intents (place/category/area/tag), then resolves them to lat/lon via SerpAPI/Nominatim/Overpass.
- **Specific-stop guarantee:** If the traveler asks for something explicit, the nearest matching stop is **forced** into the plan before other recommendations.
- **Merge-don't-overwrite discovery:** New place discovery preserves previously found specific stops (prevents user requests from being lost).
- **City-biased geocoding & robust fallbacks:** Viewbox-bounded Nominatim + Photon ensure short, within-city names resolve; multi-geocode guarantees at least one usable point.
- **Resilient, key-optional architecture:** With keys (Tavily/SerpAPI/OpenAI/Groq) quality improves; without them, OSM + OSRM still deliver a full route.

- **Transparent data trail:** The JSON output stores inputs, sources, picks, order, distances, durations, and cost assumptions with URLs.
- **Multi-step planning:** Tools orchestrated as geo → guides → places → pick & route → itinerary (LangChain + deterministic fallback).

4. Proposed Strategy — How I Approached This Use Case

Problem framing
I targeted realistic travel moments where users need *both* exploration and purpose-driven errands (pharmacy, coffee, store, restroom). The agent must honor explicit requests while still optimizing time and cost.

User input → intents
RouteForge captures a free-form "anything specific?" prompt. An LLM (OpenAI primary, Groq fallback) parses it into structured intents (place/category/cuisine/area/OSM tag). If LLMs are unavailable, the literal text is still searched.

Search orchestration
For each intent: try SerpAPI (Google Maps) near the city center; if absent, use Nominatim with a city viewbox; if a tag is provided, query Overpass directly. This returns normalized POIs with coordinates and traceable URLs.

Preserve specific requests
When general place discovery runs, we **merge** new candidates with any specific items already found—never overwriting them.

Selection & routing
Candidates are scored (fun vs. food) and **force-include one specific stop** when requested. OSRM provides distance/time; we estimate cost as distance×rate + hours×time-value. A greedy path visits stops and ends at the destination.

Resilience & outputs
If agents/LLMs rate-limit, a deterministic pipeline still produces: (1) a Markdown itinerary for humans and (2) a JSON dataset for auditing. Both embed sources and coordinates.

Validation
I verified that a user-requested stop appears in: the candidate list, the selected stops, the ordered route, and the final report. De-duplication prevents near-duplicates from crowding results.

5. References (with URLs)

- LangChain (tools, agents): https://python.langchain.com
- OpenAI API (LLM summarization): https://platform.openai.com/docs
- Groq API (LLM fallback): https://console.groq.com/docs
- Tavily (web search): https://docs.tavily.com

- SerpAPI (Google Maps): https://serpapi.com
- OpenStreetMap Nominatim (geocoding): https://nominatim.org
- Overpass API (OSM POIs): https://overpass-api.de
- OSRM (routing): http://project-osrm.org