Proposal: Sri Lanka Travel Discovery Website with AI Travel Agent

Project Title: DiscoverLK — Travel discovery & recommendation platform for Sri Lanka

Prepared for: Project Owner / Development Team

Prepared by: (Your name / Team)

Date: November 3, 2025

---

1. Executive summary

DiscoverLK is a web platform to help local and international travelers explore Sri Lanka by area and category (e.g., temples, rivers, beaches, hotels). It combines a searchable, geo-aware place database with user reviews and an AI travel agent that creates personalized itineraries and recommendations by analyzing reviews and place attributes.

This proposal describes the project goals, scope, data and technology plan, AI design, features, deliverables, team roles, risks, and next steps. It is written in clear, actionable language so you can start implementation right away.

---

2. Background & problem statement

Sri Lanka has diverse travel destinations spread across many districts. Travelers often struggle to find reliable, localized recommendations that combine up-to-date reviews, local insights, and personalized itineraries. Existing solutions are fragmented, region-limited, or paywalled. DiscoverLK aims to fill this gap by:

- Aggregating authoritative place data and user reviews for every area of Sri Lanka.
- Using NLP to analyze reviews (multilingual support: Sinhala, Tamil, English) and produce clear pros/cons and aspect-level scores.
- Offering a friendly AI travel agent that produces personalized itineraries and explains recommendations with evidence from reviews.

---

3. Project objectives

Primary objectives

1. Build a searchable, categorized place directory covering Sri Lanka (areas such as Matara, Colombo, Kandy, Galle, etc.).
2. Provide review aggregation, sentiment analysis, and an explainable ranking for "top picks" in each area.
3. Implement an AI travel agent (retrieval-augmented) that generates personalized itineraries and suggestions.
4. Support user contributions (add places, add reviews) with moderation and multilingual handling.

Secondary objectives

- Provide an admin dashboard for moderation and analytics.
- Prepare the system to integrate bookings/affiliate partners later.
- Ensure the platform is SEO-friendly and mobile-first.

---

4. Scope (what's included / excluded)

Included in this project

- Database schema and initial seeding with authoritative sources (SLTDA, OpenStreetMap) for target areas.
- Frontend web application (React/Next.js) with search, area/category browsing, place pages, reviews, and AI chat.
- Backend API (Node.js/Express or PHP/Laravel) with endpoints for search, place data, reviews, and AI requests.
- Multilingual sentiment and aspect extraction pipeline for user reviews.
- RAG-based AI agent that uses embeddings + LLM to generate itineraries and recommendations.
- Basic admin moderation panel and analytics dashboard.

Excluded (can be added later as paid features)

- Direct booking engine and payment gateway integration (can be integrated in a later phase).
- Enterprise-level scaling and distributed systems engineering for extremely high traffic (post-MVP).

---

5. Key features & user stories

Core features

- Browse by Area → Category → Place.
- Place detail page: photos, map, attributes, rating, sentiment summary (pros/cons, aspect tags).
- Full-text search + geo-filtering (radius).

- User-auth system: register, login, submit reviews and places.
- Review moderation: automated filters + admin review queue.
- AI Chat Travel Agent: ask for itineraries or recommendations; save itineraries.

User stories (examples)

- As a traveler, I can search for "family-friendly beaches near Matara" and receive ranked results with review evidence.
- As a user, I can open a place page and see the top 3 pros and cons extracted from reviews in Sinhala or English.
- As a user, I can chat with the AI and get a 3-day custom itinerary that I can save and share.

---

6. Data sources & acquisition plan

Seed data (authoritative & open):

- SLTDA / official tourist attraction lists — authoritative place names and categories.
- OpenStreetMap (OSM) / Geofabrik Sri Lanka extract — POIs, coordinates, and basic metadata.

Enrichment sources (optional / API):

- Google Places API or Mapbox Places for photos, opening hours, and additional metadata (note: billed service).
- User-submitted content (reviews, photos) captured via the app (requires moderation).

Legal & ethical note: respect API terms of service and avoid scraping sites that disallow it. Prefer official APIs or partnerships for review data from commercial vendors.

---

7. System architecture (high-level)

Frontend

- React + Next.js (for SEO-friendly pages and optional SSR)
- Tailwind CSS for UI design

Backend & services

- Node.js + Express (or PHP/Laravel if preferred) for REST API.
- PostgreSQL with PostGIS extension for geo queries and spatial indexing.
- Elasticsearch / OpenSearch for fast full-text search and geo-filtered search.
- Vector database (Pinecone / Weaviate / Milvus) for embeddings used by the AI agent.
- Hugging Face Transformers or cloud embeddings for multilingual embeddings & sentiment.
- Image storage: S3-compatible object storage (DigitalOcean Spaces, AWS S3) + CDN.
- Map tiles: Mapbox or self-hosted OSM tiles (MapTiler) depending on budget.

Infrastructure & hosting

- Frontend: Vercel (for Next.js) or similar.
- Backend: Render, DigitalOcean App Platform, or cloud provider (AWS/GCP).
- Managed Postgres (RDS / DigitalOcean managed DB) to simplify ops.

---

8. AI design & components

Goal: an AI travel agent that provides personalized, explainable recommendations and itineraries using site data and review evidence.

Approach: Retrieval-Augmented Generation (RAG)

1. **Embedding index:** compute vector embeddings for:
2. Place descriptions
3. Aggregated review summaries
4. Key admin notes (like verified events)
5. **Vector DB:** store embeddings in Pinecone/Weaviate/Milvus.
6. **Query flow:** embed user query → retrieve top-K relevant documents → assemble context (place attributes + review excerpts) → pass to LLM with structured system prompt and constraints.
7. **LLM usage:** use an LLM (OpenAI or open-source hosted model) to generate conversational text. Keep templates for itinerary outputs to ensure factual info (addresses, distances) is pulled from the structured DB rather than hallucinated.
8. **Postprocessing:** verify facts (addresses/coords), attach map links and action buttons (save, share).

Multilingual support: use multilingual embeddings & language detection. Translate or request model responses in the user's language where appropriate.

Safety & guardrails:

- Limit LLM's factual assertions by forcing inclusion of retrieved evidence and adding refusal conditions when info is absent.
- Log AI responses for moderation and continuous improvement.

---

9. Review analysis & ranking algorithm

Pipeline:

1. Ingest reviews and detect language.
2. Normalize text (cleaning), then run a sentiment classifier (multilingual model).
3. Run aspect extraction to tag mentions like "parking", "view", "family".
4. Aggregate scores per place: average rating, sentiment-weighted score, recency weight, and verified-visit boost.
5. Composite ranking score: configurable weighted sum of the above to generate "Top picks".

Explainability: show on the place page: an evidence snippet ("75% positive reviews mentioning ‘great view’ — sample: ‘Amazing sunset!’").

---

10. UX / UI & SEO considerations

Design principles

- Mobile-first, clean cards, clear CTAs ("Ask AI", "Save itinerary").
- Use short evidence snippets for trust.

SEO

- Use Next.js SSR for place pages.
- Implement structured data (JSON-LD) for each place and area (schema.org Place).
- Maintain sitemap.xml, human-friendly URLs (`/matara/beaches/umm...`) and meta tags for sharing.

Accessibility & localization

- Support Sinhala, Tamil, English text content and UI.
- Provide alt text for images and keyboard-accessible UI.

---

11. Security, privacy & moderation

- Use secure auth (JWT or session-based) and hashed passwords (bcrypt/argon2).
- Email verification and rate-limits on content submissions.
- Automated profanity and abuse filters; manual moderation queue for flagged content.
- Privacy policy and TOS; store minimal personal data and allow users to delete their data.

---

12. Deliverables

1. Database schema + seed scripts (SLTDA & OSM) for selected areas.
2. Frontend web app with search, area listing, place pages, reviews, and AI chat.
3. Backend API with endpoints for places, reviews, search, and AI.
4. Review ingestion & sentiment pipeline with multilingual support.
5. Admin dashboard for moderation and analytics.
6. Documentation: setup, deployment, and developer guides.

---

13. Project phases & milestones (ordered)

- Phase 1: Requirements finalization & area selection.

- Phase 2: DB schema design, seed scripts (import SLTDA & OSM extracts).
- Phase 3: Basic backend API and frontend listing/search pages.
- Phase 4: Reviews submission, sentiment pipeline, and place pages.
- Phase 5: AI agent (RAG) prototype and integration with frontend chat.
- Phase 6: Admin/moderation dashboard and analytics.
- Phase 7: Testing, SEO tuning, and pre-launch polish.
- Phase 8: Launch & post-launch monitoring; collect user feedback and iterate.

Note: milestones are presented in sequence; specific calendar durations can be defined by the project manager.

---

14. Team & roles

- Project Owner / Product Manager — clarify scope & priorities.
- Frontend Developer (React/Next.js) — implements UI and UX.
- Backend Developer (Node.js/Express or PHP/Laravel) — API, auth, DB.
- Data Engineer — OSM/SLTDA ingestion, ETL pipelines.
- ML/NLP Engineer — embeddings, sentiment, aspect extraction, RAG design.
- UI/UX Designer — visuals, responsive behaviour.
- QA & Ops — testing, deployment, and monitoring.
- Community/Content Manager — review moderation and partnerships.

---

15. Risks & mitigation

Risk: Data licensing or scraping violations. **Mitigation:** Use official APIs and partnerships; avoid scraping where disallowed.

Risk: LLM hallucinations or inaccurate itineraries. **Mitigation:** Use RAG with retrieved evidence and strict templates; post-verify facts from structured DB.

Risk: Spam/fake reviews. **Mitigation:** Email verification, rate-limits, automated duplicate detection, and manual moderation.

Risk: Cost of third-party APIs (maps, LLMs). **Mitigation:** Use open-source alternatives for initial MVP; monitor usage and add billing controls.

---

16. Success metrics (KPIs)

- Number of places indexed and active areas covered.
- Monthly active users and user retention.
- Number of reviews submitted and percent verified.
- AI sessions completed and user satisfaction score (post-session rating).
- Conversion signals (saved itineraries, shares, partner clicks).

17. Estimated costs (categories)

- Hosting (frontend, backend, DB)
- Map API & tile hosting
- LLM & embedding usage (if using cloud provider)
- Vector DB hosting
- CDN & image storage
- Development & maintenance (team salaries or contractor fees)

Provide exact figures based on vendor choices; a cost estimate can be prepared if you provide constraints (budget cap, expected monthly traffic).

---

18. Next recommended actions (concrete)

1. Approve project scope and decide initial target areas (e.g., Colombo, Matara, Galle, Kandy, Ella).
2. Decide on primary tech stack preference (Node.js or PHP backend).
3. Get access keys: SLTDA data (if needed), Google/Mapbox API (optional), and a vector DB provider trial.
4. Start Phase 2: DB schema + seed scripts.

If you want, I can now generate any of the following immediately:

- SQL schema + seed scripts for PostgreSQL/PostGIS to import SLTDA & OSM extracts.
- A minimal Next.js + Express starter repository structure with sample API routes and a basic UI.
- AI agent prompt templates and example RAG workflow with sample prompts and response formats.

Tell me which item to produce next and I will generate it right here.