Pilot vertical — Adventure Sports (Okanagan, Canada)

This pilot targets local guides and small operators offering mountain and lake activities: mountain biking, road biking, trail running/walking, kayaking, swimming/relaxing, and winter walking/hiking in the Okanagan region.

MVP feature list (prioritized)

1. Client onboarding
    - Sign up / login (email + password, OAuth optional)
    - Organization profile (business name, contact, service types, operating area)
2. Consultation form (guided)
    - Choose activity type (mountain bike, road bike, kayak, swim/relax, walking trail)
    - Date(s), group size, fitness/skill level, preferred difficulty, duration, equipment owned vs rental
    - Preferences: nature vs adventure vs relaxed, accessibility needs, pet-friendly, weather sensitivity
    - Constraints: budget, time window, transportation limits
3. Prompt constructor & middleware
    - Map form inputs to a structured JSON prompt schema
    - Validate required fields and sanitize inputs
    - Provider-agnostic LLM call with configurable model and cost caps
4. Generated consultation deliverable
    - Normalized JSON from LLM -> rendered itinerary/proposal
    - Components: day schedule, route maps (links), packing/equipment list, difficulty rating, safety notes, nearest services, simple cost estimate
    - Export as PDF and email share
5. Feedback & iteration
    - Client can request alternatives (easier/harder, shorter/longer) and the middleware re-runs prompt with updated constraints
6. Booking / next steps CTA (MVP soft integration)
    - Add a contact/booking request to operator inbox (no full payment flow in MVP)
7. Admin dashboard
    - View consultations, prompt versions, usage and cost tracking, logs
8. Data controls & privacy
    - Export & delete consultation data, per-client isolation

User flow (step-by-step)

1. Operator onboarding
    - Operator registers and configures their profile and service templates (e.g., "Mountain Bike Half-Day" template).
2. Customer/request intake (two modes)
    - Public lead form (embed on website) OR operator staff opens client area and starts a consultation.
3. Fill consultation form
    - Select activity and answer guided questions (date, skill, preferences).
    - System shows suggested defaults and warnings (weather season, trail closures if known).

4. Generate proposal
    - Operator clicks "Generate plan" -> frontend sends structured payload to backend
    - Middleware composes a JSON prompt (includes operator metadata, template id, user answers, local context like season and typical weather)
    - Background job calls LLM and stores raw + normalized response
5. Present results
    - Rendered itinerary with sections (overview, timeline, equipment, safety, local tips)
    - Visual elements: small route map link, simple cost table, difficulty badges
6. Iterate
    - Customer or operator requests variations (e.g., "more family-friendly") -> form updates constraints -> re-run
7. Convert to booking lead
    - If customer accepts, create a lead entry and optionally send a PDF by email
8. Analytics & improvement
    - Track prompts used, success/acceptance rate, cost per prompt, and user feedback to refine templates

Prompt & data schema (example)

- Consultation payload (JSON) — minimal example fields: { "operator_id": "uuid", "template_id": "mountain_bike_halfday", "activity": "mountain_bike", "date": "2025-09-12", "group_size": 4, "skill_level": "intermediate", "duration_hours": 4, "preferences": ["adventure","scenic"], "constraints": {"budget_per_person": 100} }

- LLM prompt contract: request a normalized JSON with keys: overview, timeline[], equipment[], safety[], cost_estimate, alternatives[]

Operational considerations

- Local data enrichment: include lightweight rules or small data sources (season calendars, common trail difficulty mappings, approximate drive times) to supplement LLM outputs.
- Safety & liability: include disclaimers in deliverables; do not provide medical/legal advice.
- Caching & rate control: cache recent similar prompts and enforce per-operator spend caps.

Recommended modern software stack

- Frontend

    - Framework: React + TypeScript (Vite)
    - UI: Tailwind CSS + Radix UI or Chakra UI
    - State & data: TanStack Query (React Query) + Zustand or Jotai for local state
    - Maps & routing: Mapbox GL or Leaflet + Mapbox/OSM tiles, static route links

- Backend & middleware

    - Framework: FastAPI (Python) with Pydantic/SQLModel for schemas (async)
    - Background jobs: Celery or Redis Queue (RQ) with Redis
    - LLM orchestration: a small provider-agnostic middleware layer (use OpenAI-compatible SDKs; optionally integrate LangChain components for multi-step flows)

    - Authentication: JWT + refresh tokens; OAuth optional

- Data storage
    - Primary DB: PostgreSQL (Timescale extension optional for time series)
    - Cache: Redis
    - Object storage: S3-compatible (MinIO for self-hosting)
- Observability & infra
    - Logs: structured logs (JSON) to stdout -> aggregated by Loki/ELK or a hosted provider
    - Metrics: Prometheus + Grafana or hosted metrics
    - Tracing: OpenTelemetry
- Deployment & CI
    - Containerization: Docker + multi-stage builds
    - CI/CD: GitHub Actions
    - Hosting: Render / Railway / Fly.io / DigitalOcean App Platform for quick MVP; Vercel for frontend
    - Secrets: Vault or cloud provider secret store

MVP priorities & success metrics

- Launch a working consultation flow for one operator in 4–8 weeks
- Metrics to track:
    - Number of consultations generated / week
    - Conversion to leads (accepted proposals)
    - Average tokens/cost per consultation
    - Time-to-first-proposal (latency)
    - Feedback rating from operator

Next practical steps for the pilot

1. Choose an operator partner in the Okanagan and collect sample intake answers and common routes.
2. Design one or two form templates (e.g., "Mountain Bike Half-Day", "Kayak Day Trip").
3. Implement a minimal FastAPI backend + one React page for the consultation form and results.
4. Wire a single LLM provider and build the JSON prompt contract; iterate with human-in-the-loop refinement.
5. Run pilot, collect feedback, refine templates and safety rules.