Flight Booking Meta-Search Platform — PRD (Scope-Frozen)

Document Type: PRD-Only (Convergent / Scope-Frozen)

Model: Flight meta-search + partner redirect (no direct booking/payments)

Prepared for internal review and implementation alignment.

Table of Contents

1. Document Control

- **Product Name**: Flight Booking Meta-Search Platform (Web)
- **PRD Version**: 1.1 (Scope-Frozen + AI Addendum Integrated)
- **Author**: Product / Architecture / Program Office
- **Date**: 2026-02-26
- **Stakeholders**
    - Product: Head of Product, Growth Lead
    - Engineering: Frontend Lead, Backend Lead, Data/ML Lead (Phase 2+), QA Lead
    - Platform/DevOps: DevOps Lead, SRE/Observability
    - Security & Compliance: Security Lead, Legal/Privacy Counsel
    - Commercial: Partnerships Lead (GDS/OTA/Airline), Finance
- **Approval Status**: Draft for signature
- **Scope Version Reference**
    - Baseline Scope: SOW v1.0 (Feb 2026)
    - MVP/Phase 2 partition: finalized prioritized buckets (Must-Build MVP vs Phase 2)
    - Architecture alignment baseline: SOW recommended stack + system architecture
- **Change Log**
    - v1.0 (2026-02-25): Initial scope-freeze PRD aligned to SOW v1.0 + prioritized MVP/Phase 2 buckets.
- v1.1 (2026-02-26): Integrated AI Workstream capabilities from AI Feature Documentation v1.0 under Phase 2+ with strict guardrails.

---

2. Product Overview

2.1 Product Summary

A web-based flight meta-search platform that queries multiple flight supply sources (GDS/OTAs/airlines), normalizes results, displays comparisons, and **redirects users to partners for booking** with **price verification and affiliate attribution**.

2.2 Problem Statement

Users face fragmented flight pricing across airlines/OTAs. The platform reduces search cost by consolidating and comparing offers and sending users to book with a selected provider.

2.3 Target Market

- **Primary**: Consumer travelers searching flights on web (mobile-first responsive).
- **Secondary**: Repeat travelers who use saved trips + alerts to convert later.

2.4 Business Model

- **Affiliate partnerships** (tracked redirects + commission management) and advertising as defined in objectives.

2.5 In-Scope Features (Frozen)

Core platform features, admin panel, SEO/marketing pages, and technical integrations as explicitly listed in SOW v1.0 and finalized prioritization.

2.6 Out-of-Scope Features (Frozen)

- Direct booking/payment processing (meta-search redirect model)
- Native mobile apps (iOS/Android)
- Multilingual beyond English
- Hotels, car rentals, other travel products
- GDS/OTA partnership fees + API subscription costs

2.7 Release Scope Definition (MVP vs Phase 2)

MVP Scope (Scope-Frozen)

MVP is the **minimum commercially operable meta-search** that supports:

- Core search + results + redirect booking flow + price verification + partner attribution
- User management + alerts foundation
- Admin panel foundation for partner/content/user management + reporting
- SEO route/city pages + sitemap + schema + meta tags
- Analytics integration

Prioritized MVP must-build (implementation priority inside MVP)

- Supplier integration + caching + polling (aggregation core)
- Price confirmation/mismatch handling (price verification)
- Attribution + conversion tracking
- SEO route pages v1

Phase 2 Scope (Scope-Frozen)

Phase 2 is **revenue/CTR acceleration** that does not change the redirect model:

- Best-value ranking + explainability
- Sponsored placements marketplace
- Partner quality scoring + suppression rules
- Fare calendar / whole-month view
- Explore/Anywhere discovery
- Price alerts v2 (watchlist/digest improvements)

Phase 2+ AI Workstream (Scope-Frozen, additive; behind feature flags)

- Natural language search (NL→Query) with schema-validated parsing + transparent editable parameters
- Personalized "Best Value" ranking (ML/LTR) with explainability tags and hard policy constraints
- Offer explanations ("Why this", inclusions: baggage/refundability) grounded in supplier + fare data
- Price prediction guidance ("Buy / Wait / Uncertain" with confidence) + "price stability" label
- Supplier call optimization (fan-out planner) to reduce cost/search while maintaining coverage
- Fraud / IVT detection to protect ad budgets and partner trust; exclude invalid clicks from billing where applicable
- Support Copilot (session-aware) for Ops/Admin; Partner Onboarding Copilot for integration acceleration
- AI-assisted SEO at scale (template-first, data-grounded) + Analytics Copilot in Admin

————————————————————————————————

3. Success Metrics (KPIs)

All KPIs below are **targets** used for sprint planning, QA exit criteria, and investor diligence.

3.1 Acquisition

- **Monthly sessions (web)**: MVP ≥ **600,000**; Phase 2 ≥ **1,800,000**
- **Monthly searches**: MVP ≥ **1,000,000**; Phase 2 ≥ **3,000,000**
- **SEO ranking target**
    - MVP: **Top-20** Google rank for **50** priority "Origin→Destination flights" keywords
    - Phase 2: **Top-10** Google rank for **150** priority route keywords

3.2 Conversion

- **Search → click (CTR)**: MVP ≥ **12.0%**; Phase 2 ≥ **15.0%**

- **Redirect conversion % (click → booking)** (via partner postback/pixel): MVP ≥ **3.0%**; Phase 2 ≥ **3.5%**
- **Price mismatch rate** (selected offer differs at partner landing): MVP ≤ **2.5%**; Phase 2 ≤ **1.5%**

3.3 Revenue

- **Revenue per 1,000 searches (RPS)**: MVP ≥ **$25.00**; Phase 2 ≥ **$40.00**
- **Monthly revenue target**: MVP ≥ **$25,000**; Phase 2 ≥ **$120,000**

3.4 System Performance / Reliability

- **Page load**: ≤ **3.0s** (p75 LCP on mid-tier mobile)
- **API response:** ≤ **500ms** (p95 for non-search endpoints)
- **Search time-to-first-results**: ≤ **800ms p95** (first partial chunk)
- **Search time-to-stable-results**: ≤ **2.5s p95** (hard cutoff aligned to orchestrator deadline model)
- **Uptime**: ≥ **99.9%** monthly

3.5 AI/ML Effectiveness & Safety (Phase 2+)

- **NL→Query parse success rate** (structured params produced + validated): ≥ **95%** of NL inputs; fallback to classic form otherwise
- **NL parse correction rate** (user edits parsed params): tracked; target ≤ **20%** after tuning
- **Ranking quality (offline)**: NDCG@10 tracked weekly; non-regression gate on major releases
- **Best-value CTR lift (online A/B)**: ≥ **+3%** relative to control for eligible cohorts with stop-loss if mismatch rate rises > **+0.5%** absolute
- **Price prediction directional accuracy**: tracked; must display confidence; **no "guarantees"** allowed
- **Supplier calls per search**: reduce vs baseline while maintaining result coverage (coverage drop ≤ **1%**)
- **Invalid traffic rate**: decreasing trend; partner dispute rate ≤ **baseline**
- **AI safety/fallback rate**: 100% of AI outputs must be schema-validated; on failure system uses deterministic fallback

4. User Personas (Execution-Oriented)

P1 — Anonymous Searcher

- **Primary goals**: Find cheapest/fastest options quickly; compare providers; click out to book.
- **Core interactions**: Search → filter/sort → provider list → redirect.

- **Key features used**: Search engine, results filters, fare breakdown, redirect flow.
- **Conversion drivers**: Fast results, total-price transparency, trust (price verification).
- **Edge case behaviors**: Supplier timeouts; no results; invalid dates; airport ambiguity.

P2 — Logged-In Planner

- **Primary goals**: Save routes/trips; revisit searches; reduce repeated entry.
- **Core interactions**: Login → search → save trip → revisit → click out.
- **Key features used**: User profiles, search history, saved trips, trip dashboard.
- **Conversion drivers**: Saved content reduces friction; alerts trigger return.

P3 — Price Alert User

- **Primary goals**: Track route/date; receive price-drop notifications; convert at the right time.
- **Core interactions**: Create alert → receive email/push → return → click out.
- **Key features used**: Alerts, notifications, route tracking.
- **Edge case behaviors**: Notification fatigue; stale prices; duplicates; opted-out messaging.

P4 — Admin / Ops User

- **Primary goals**: Configure partners, commissions, API configs; monitor KPIs; manage content/users; export reports.
- **Core interactions**: Admin login → dashboard → partner mgmt → content updates → report exports.
- **Key features used**: Admin dashboard, partner mgmt, reports, settings/templates.
- **Edge case behaviors**: Misconfigured credentials, commission disputes, suspicious click patterns.

——————————————————————————————

5. User Journeys (Deterministic Flow)

5.1 Anonymous User Flow (Search → Redirect)

1. User loads home page.
    - **System**: Serve from CDN; log page_view (anon session id).
2. User enters route/dates/pax/cabin; submits.
    - **System**: Validate inputs; create SearchSession; return searchSessionId.
3. UI displays skeleton; polls session.
    - **System**: Orchestrator fans out to suppliers; streams partial results; caches.

4. User applies filters/sort.
    - **System**: Apply client-side filters when possible; server-side if required; log filter_applied.
5. User selects flight; opens provider list.
    - **System**: Log result_select with itinerary_id and rank_position.
6. User clicks provider.
    - **System**: Run price verification; generate click_id; persist RedirectEvent; redirect with tracking params.
7. Failure handling
    - Price verify fails → show "price changed" state; offer refresh; log price_mismatch.

Redirect tracking logic: click_id generated server-side; included in deeplink parameters; partner conversion postback reconciles to CommissionEvent (supports pixel/report/S2S).

5.2 Logged-In User Flow (Save + Return)

Same as anonymous plus:

- Save route/trip
- Store search in SearchQuery and favorite in SavedTrip
- On return: show "recent searches" and saved trips; allow one-click rerun search

5.3 Price Alert User Flow

8. User toggles "Track price" on a search.
9. System creates PriceAlert with baseline price and rules.
10. Background job evaluates alert on schedule; sends notification on threshold condition.
11. User clicks notification deep link back to route page / search session.
12. System logs notification events and resulting clickouts.

5.4 Admin User Flow

13. Admin logs in (RBAC).
14. Views KPI dashboard (searches, CTR, revenue, conversions).
15. Adds/edits partner (API config + commission).
16. Publishes SEO page content and triggers sitemap refresh.
17. Exports partner performance report; audits suspicious click IDs.

6. Functional Requirements (Atomic & Testable)

> ***Scope rule**: Each FR below is **MVP** unless explicitly tagged "Phase 2".*

FR-01 — Flight Search Engine (MVP)

Description: Search form and session creation supporting one-way, round-trip, multi-city; passenger selection; cabin; airport autocomplete; flexible date grid; currency localization.

User Story: As a traveler, I want to search flights by route/dates/pax/cabin so that I can compare available options.

Preconditions

- Supplier integrations configured and enabled.
- Airport database loaded (IATA + city + nearby airports).

Functional Behavior

- Search types: one-way, round-trip, multi-city
- Pax: adult/child/infant; cabin: economy/business/first
- Autocomplete returns airports/cities + "nearby airports" suggestions
- Flexible dates: ±3 days grid (UI + query representation)
- Create search session; progressive retrieval via polling

Edge Cases

- Invalid pax counts (e.g., infants > adults)
- Multi-city legs missing airport/date
- Same origin/destination; return before departure
- Autocomplete ambiguity (city with multiple airports)

Validation Rules

- Dates must be ISO; depart_date <= return_date when round-trip
- Pax totals: adults>=1, infants<=adults, total_pax<=9
- Airports must be valid IATA or internal airport id

API Contract Implications

- POST /api/search/sessions
    - body: { legs[], tripType, pax, cabin, currency, locale }

  - response: { searchSessionId, expiresAt }
- GET /api/search/sessions/{id}/poll?cursor=...
  - response: { status, results[], nextCursor, stats }

Data Model Implications

- SearchQuery, SearchSession, SearchLeg, SearchSessionState

Security Considerations

- Rate limit session creation; bot detection at edge
- No PII required for anonymous search

Logging & Monitoring Requirements

- Log: search_create, search_poll, search_complete, latency, suppliers attempted
- Metric: p95 time-to-first-results; p95 completion time

Acceptance Criteria

- Supports all search modes and validations listed above
- First partial results returned ≤ 800ms p95; stable results ≤ 2.5s p95
- Flexible date grid queries return results for each day where supply exists

---

FR-02 — Fare Aggregation (Supplier Fan-out, Normalization, Dedup, Cache) (MVP)

Description: Query multiple suppliers (GDS/OTAs/airlines), normalize offers, deduplicate itineraries, enforce timeouts and caching.

User Story: As a traveler, I want one results list even though data comes from multiple providers.

Preconditions

- At least 1–2 providers integrated for MVP baseline

Functional Behavior

- Orchestrator executes supplier plan; parallel calls; progressive completion

- Cache keys include route + dates + pax + cabin + currency
- Normalize into canonical schema: itineraries → segments → fares → provider options
- Dedup rule: same itinerary cluster; preserve multiple providers/prices

Edge Cases

- Partial supplier failures; timeouts
- Duplicate itineraries with slightly different times/flight numbers
- Currency mismatch from supplier

Validation Rules

- Any offer displayed must include: total price, currency, itinerary segments, provider deeplink
- Reject offers missing mandatory fields or failing schema validation

API Contract Implications

- Internal: `SupplierAdapter.search()` contract normalized to platform schema
- External: search poll returns combined list with `providerOptions[]`

Data Model Implications

- `FlightOption`, `FareDetail`, `Partner`, `PartnerOfferRaw` (optional, for debugging)

Security Considerations

- Secrets storage for partner credentials in vault/secret manager
- Strict outbound allowlist; per-supplier rate limiting

Logging & Monitoring Requirements

- Supplier-level: latency, timeout %, error taxonomy, circuit breaker state
- Cache: hit ratio; average results per query; cost-per-search proxy

Acceptance Criteria

- Results show at least one provider option per itinerary when supply exists
- Supplier timeouts do not break search; degraded mode returns partial results with disclosure
- Cache hit ratio ≥ 25% at steady state (post warm-up)

————————————————————————————————

FR-03 — Search Results & Filters (MVP) + Phase 2 Ranking Enhancements

Description: Results list with filters, sorting, fare breakdown, baggage info, “best value” sort mode, and transparent display.

User Story: As a traveler, I want to filter/sort flights and understand total cost so that I choose confidently.

Preconditions

- Normalized offers include baggage + price breakdown fields when available

Functional Behavior (MVP)

- Filters: price, stops, airline, depart time, duration, layover airports
- Sorting: price, duration, depart, arrive, best value
- Fare breakdown: base + tax + fees
- Baggage info: cabin/checked baggage per fare
- Provider comparison: show multiple providers per itinerary

Phase 2 Additions

- Best-value ranking explainability (“why this is recommended”)
- Partner quality scoring used in ranking & suppression

Edge Cases

- Filter yields zero results → deterministic empty state + “clear filters”
- Missing baggage details → show “Not provided by supplier”

Validation Rules

- Sorting must be stable and deterministic
- All prices displayed must include currency symbol and total price

API Contract Implications

- GET /api/search/sessions/{id}/poll returns:
  - results[] with filterable attributes
  - facets (min/max price, airlines list, stops distribution)

Data Model Implications

- Persist impression events for ranking evaluation (Phase 2 needs training data)

Security Considerations

- Prevent parameter tampering on facet endpoints; server verifies query session

Logging & Monitoring Requirements

- Log: impression, filter_applied, sort_changed, result_select
- KPI: CTR by rank position; filter usage; bounce rate

Acceptance Criteria

- All filters and sorting modes function correctly with a 10,000 results QA fixture
- Fare breakdown and baggage info displayed when present; never display incomplete numeric totals

———————————————————————————————

FR-04 — Booking Redirect Flow (MVP)

Description: Deep-link redirect to partner for booking, with price verification and booking confirmation notification (only when measurable via partner events).

User Story: As a traveler, I want to click a provider and book on their site with correct tracking so that the platform can attribute the booking.

Preconditions

- Partner deeplink templates configured
- Partner attribution contract & required parameters defined

Functional Behavior

- On provider click:
  1. Verify price immediately prior to redirect
  2. Generate click_id
  3. Persist RedirectEvent
  4. Redirect (HTTP 302) to partner deeplink with tracking params
- Booking confirmation email:
  - Trigger **only if** conversion event is received (postback/pixel) mapped to click_id

Edge Cases

- Price verification fails → show updated price and allow proceed
- Partner deeplink unavailable → fail with provider unavailable message
- Conversion event missing → no booking confirmation sent

Validation Rules

- Redirect must preserve tracking parameters end-to-end
- Price verification must run against the selected provider

API Contract Implications

- POST /api/redirect
  - body: { searchSessionId, itineraryId, partnerId }
  - response: { redirectUrl } or server-side 302

Data Model Implications

- RedirectEvent, Partner, CommissionEvent

Security Considerations

- Open redirect protection: allowlist partner domains; sign redirect payload
- No PII embedded in tracking parameters

Logging & Monitoring Requirements

- Log: redirect initiated, redirect success, verify outcome, mismatch
- Metric: click success rate; price mismatch rate; postback match rate

Acceptance Criteria

- 100% of redirects generate a persisted RedirectEvent
- 0 open-redirect vulnerabilities (validated by security test)
- Price verification executes within 300ms p95

FR-05 — Affiliate Tracking & Attribution (MVP) + Phase 2 Enhancements

Description: Click ID generation, UTM preservation, commission management, and reconciliation of partner-reported conversions.

User Story: As the business, I want accurate attribution so revenue and partner performance can be measured and billed.

Preconditions

- Partner supports at least one: pixel, report upload, or server-to-server postback

Functional Behavior

- Generate click_id per provider click
- Store mapping: click → search/session → itinerary → partner
- Ingest conversions via:
  - POST /api/attribution/postback (S2S) (preferred)
  - pixel endpoint (fallback)
  - CSV report import (admin tool)
- Reconciliation:
  - Match conversions to click_id; compute variance vs partner report

Phase 2 Additions

- Fraud/IVT scoring incorporated into attribution decisions
- Partner tiering based on mismatch, conversion, and complaint signals

Edge Cases

- Duplicate postbacks → idempotent upsert
- Late conversions (up to 30 days) → still match if click_id exists

Validation Rules

- Postback endpoint requires HMAC signature per partner
- Idempotency key on conversion_id

API Contract Implications

- POST /api/attribution/postback
- GET /api/admin/partners/{id}/performance

Data Model Implications

- CommissionEvent, AttributionSource, PartnerPayoutRule

Security Considerations

- Strict auth for postback; WAF rules
- Never store partner payload containing PII unless required; default reject

Logging & Monitoring Requirements

- Log: postback received, validated, matched, rejected
- Alert: postback failure rate > 2% in 15 minutes

Acceptance Criteria

- ≥ 98% of conversion events match a click_id (for partners providing click_id)
- Duplicate postbacks do not inflate revenue

FR-06 — Price Alerts & Notifications (MVP) + Phase 2 “Digest/Watchlist” Upgrade

Description: Email/push alerts on price drops; route tracking.

User Story: As a traveler, I want alerts when prices drop so I can book at a better time.

Preconditions

- Email service configured; push configured for users with tokens

Functional Behavior (MVP)

- Create alert on (origin, destination, date range/fixed date, pax, cabin)
- Set baseline price at creation time
- Evaluator job runs every 6 hours and checks:
    - If current lowest price ≤ baseline − (absolute threshold)
- Send notification via email; push if enabled

Phase 2 Additions

- Watchlist + weekly digest
- Smarter thresholds and suppression

Edge Cases

- No current price found → no alert
- Duplicate alerts for same route/date → merge or reject

Validation Rules

- User must verify email before enabling alerts
- Frequency cap: max 2 alerts/day/user

API Contract Implications

- POST /api/alerts
- GET /api/alerts
- DELETE /api/alerts/{id}

Data Model Implications

- PriceAlert, NotificationEvent

Security Considerations

- Standard auth required; opt-out compliance and audit

Logging & Monitoring Requirements

- Log alert evaluations, triggered alerts, delivery results
- Metric: alert open rate; conversion after alert click

Acceptance Criteria

- Alerts trigger correctly when threshold met in test fixtures
- Delivery success ≥ 99% for valid addresses/tokens (excluding hard bounces)

———————————————————————————————

FR-07 — User Management (MVP)

Description: Registration (email + OAuth), profiles, search history, saved trips, upcoming trips dashboard.

User Story: As a user, I want an account to save trips and manage alerts.

Preconditions

- OAuth providers configured; email verification enabled

Functional Behavior

- Register/login via email + OAuth providers (Google/Facebook)
- Profile fields: preferences + frequent traveler details
    - **Scope guard**: do not store payment methods in MVP
- Search history: store searches and viewed flights
- Saved trips: wishlist/favorites
- Trip dashboard: upcoming trips (derived from saved trips + conversion events where available)

Edge Cases

- OAuth account without verified email
- User deletes account → deletion/anonymization

Validation Rules

- Password policy (if email/password): min 10 chars + rate limit attempts
- Email verification required for alerts

API Contract Implications

- /api/auth/*, /api/me, /api/users/{id}/search-history, /api/users/{id}/saved-trips

Data Model Implications

- User, UserIdentity, SavedTrip, UserPreference

Security Considerations

- OWASP Top 10 coverage required at release
- JWT/session security, CSRF protection (if cookie-based)

Logging & Monitoring Requirements

- Auth events, suspicious logins, account changes

Acceptance Criteria

- Login flows pass QA for email and OAuth
- Account deletion removes/anonymizes personal data within 30 days

---

FR-08 — Admin Panel (MVP) + Phase 2 Monetization Controls

Description: Admin dashboard KPIs; partner management (commission + API configs); content management; user management; reports; settings.

User Story: As an admin, I want to manage partners, content, users, and reporting so that operations run reliably.

Preconditions

- Admin RBAC configured

Functional Behavior (MVP)

- Dashboard: searches, conversions, revenue, traffic analytics
- Partner management: add/edit partners, commission rates, API configs
- Content management: SEO pages, blog, FAQs, static content
- User management: accounts (support/disputes placeholders)
- Reports: revenue, partner performance, funnels
- Settings: platform config, email templates, notification settings

Phase 2 Additions

- Sponsored placement configuration and review workflow
- Partner quality score thresholds and suppression rules

Edge Cases

- Invalid partner credentials break searches → admin can disable partner immediately
- Role separation: content editors cannot edit commissions

Validation Rules

- All admin changes audited (immutable audit log)
- Commission config changes require 2-person approval (optional governance baseline)

API Contract Implications

- /api/admin/partners/*, /api/admin/reports/*, /api/admin/content/*

Data Model Implications

- AdminUser, Partner, AuditLog

Security Considerations

- RBAC: Admin vs Editor vs Analyst
- MFA required for Admin accounts

Logging & Monitoring Requirements

- Audit all admin actions: who/what/when/before/after

Acceptance Criteria

- Admin can disable a partner in < 60 seconds; search excludes it within next poll cycle
- All admin writes produce an AuditLog entry

———————————————————————————

FR-09 — SEO Route Pages & Programmatic Marketing Pages (MVP)

Description: Dynamic SEO route pages, city pages, sitemap generation, schema markup, blog/content hub, dynamic meta tags.

User Story: As a traveler coming from Google, I want a relevant route page so I can search quickly.

Preconditions

- Route/city content templates defined

Functional Behavior

- Route pages for popular routes (Origin→Destination)
- City pages: destination guide with flight info
- Automated sitemap regeneration on content changes
- JSON-LD schema for flights/FAQs/breadcrumbs
- Dynamic OG/Twitter meta tags

Edge Cases

- Thin pages → must include data-driven blocks (min content rule)
- Duplicate route pages (alias airports/cities) → canonicalization

Validation Rules

- Each route page must have: unique title/meta, canonical URL, schema validation passing

API Contract Implications

- /api/seo/routes/{origin}/{dest} (optional SSR data source)
- /api/sitemap/regenerate (admin only)

Data Model Implications

- RoutePage (publish status, canonical, metadata)

Security Considerations

- Protect admin endpoints; sanitize rich text

Logging & Monitoring Requirements

- Log content publishes; sitemap generation; crawl errors

Acceptance Criteria

- Sitemap contains all published route/city pages
- Schema markup validates with no critical errors

——————————————————————————————

FR-10 — Analytics & Attribution Instrumentation (MVP)

Description: Integrate analytics, event taxonomy, conversion funnels, and experiment readiness.

User Story: As product/growth, I want measurable funnels so we can improve CTR and revenue.

Preconditions

- Analytics accounts configured

Functional Behavior

- Track: search_create, results_impression, filter_applied, provider_click, redirect_success, conversion_received
- Funnel: Search → Impression → Click → Conversion
- Attribution dimensions: route, device, locale, partner, rank position

Edge Cases

- Ad blockers: server-side event fallback for key events

Validation Rules

- Event payload contains no PII
- Consistent naming and schema versioning

API Contract Implications

- /api/events (optional server-side collector)

Data Model Implications

- EventLog (warehouse) or streaming sink

Security Considerations

- PII redaction middleware

Logging & Monitoring Requirements

- Monitor ingestion lag; dropped events

Acceptance Criteria

- Funnel dashboard computes CTR and click-to-book conversion by partner
- < 1% event schema validation failures

———————————————————————————————

FR-11 — Notification System (Email + Push) (MVP)

Description: Transactional email and push pipeline for alerts and booking confirmations (only if conversion signal exists).

User Story: As a user, I want timely notifications so I don’t miss price changes.

Preconditions

- Email via SendGrid/AWS SES; push via FCM

Functional Behavior

- Templates: verification, price drop, digest, booking confirmation
- Delivery retries with exponential backoff
- Suppression list and unsubscribe compliance

Edge Cases

- Email bounces; push token invalid

Validation Rules

- All outbound includes required footer + unsubscribe
- Rate caps per user

API Contract Implications

- Internal: `NotificationService.send(type, userId, payload)`

Data Model Implications

- `NotificationEvent`

Security Considerations

- DKIM/SPF/DMARC configured; secrets secured

Logging & Monitoring Requirements

- Delivery success rate; queue depth; provider errors

Acceptance Criteria

- ≥ 99% successful sends for valid addresses/tokens
- Retries do not exceed 24 hours

———————————————————————————————

FR-12 — Error Handling & Fallback Logic (MVP) + Phase 2 Governance

Description: Resilience patterns for supplier outages, timeouts, degraded mode, and user-safe messaging.

User Story: As a traveler, I want the system to still work even if some suppliers fail.

Preconditions

- Circuit breaker and timeout strategy defined

Functional Behavior

- Two-deadline model:
  - Soft timeout ~1.8s; hard cutoff ~2.3s; drop late results
- Degraded mode: return partial results with disclosure banner
- Supplier circuit breaker: open on high failure; auto-retry after cooldown
- Central error taxonomy: SUPPLIER_TIMEOUT, SUPPLIER_AUTH_FAIL, RATE_LIMIT, NORMALIZATION_FAIL

Edge Cases

- 0 suppliers return results → deterministic no-results + retry

Validation Rules

- Never crash search poll endpoint; always return status + errors list

API Contract Implications

- Search poll includes errors[] and degradedMode=true/false

Data Model Implications

- SupplierHealth, SupplierIncident

Security Considerations

- Do not leak supplier credentials or raw error payloads to clients

Logging & Monitoring Requirements

- Alerts on supplier timeout spikes; search error rate spikes

Acceptance Criteria

- Any supplier failure still yields valid response
- Degraded mode triggers correctly under simulated 50% supplier outage

---

FR-13 — AI Gateway Service (Phase 2+)

Description: Central service for AI/ML inference used by the product (LLM + ML models). Provides **redaction**, **schema enforcement**, **rate limiting**, **caching**, and **audit logging** so that AI can be used safely without compromising deterministic core flows.

User Story: As the platform, I want a controlled AI access layer so that AI features can be rolled out safely and observed end-to-end.

Preconditions

- Model provider(s) configured (self-hosted or managed); secrets stored in vault/secret manager
- Redaction/tokenization policy approved by Security/Privacy
- JSON schema definitions registered for each structured output

Functional Behavior

- Single “AI Gateway” API for all AI calls (NL parsing, explanations, predictions, copilots)
- Mandatory request envelope includes: prompt_template_id, use_case, tenant/env, trace_id
- Redaction layer:
  - Strip / hash PII (email, phone, full name) before external model calls unless explicitly approved
- Structured output enforcement:
  - LLM outputs must be valid JSON matching schema; on parse fail → deterministic fallback

- Policy layer:
  - Reject unsafe requests (prompt injection indicators, disallowed content types)
- Caching:
  - Cache safe, non-personal outputs where appropriate (e.g., generic explanations for route stats blocks)
- Rate limiting:
  - Per user/session and per internal service; separate budgets for consumer vs admin copilots

Edge Cases

- Model provider outage → immediate fallback without blocking core search results
- Schema drift → reject and fallback; alert engineering
- Prompt injection attempt via user input → sanitize and fallback to classic UI/behavior

Validation Rules

- No AI feature may call model providers directly; must go through AI Gateway
- Any AI decision affecting ranking/monetization must log decision path (ai_applied vs fallback_used)

API Contract Implications

- Internal service endpoints (behind auth):
  - POST /internal/ai/parse_nl_query
  - POST /internal/ai/explain_offer
  - POST /internal/ai/predict_price
  - POST /internal/ai/copilot_query
- All AI gateway responses include: { output, confidence?, model, model_version, latency_ms, fallback_used }

Data Model Implications

- AIInferenceLog (immutable) with: id, use_case, model, model_version, prompt_template_id, input_hash, output_hash, confidence, latency_ms, decision_path, trace_id, created_at
- Optional: PromptTemplateRegistry for versioned templates + approvals

Security Considerations

- Never store raw prompts that contain PII; store only redacted inputs + hashes
- Protect model endpoints with mTLS/service auth; rotate keys
- Strict outbound allowlist for model provider domains

Logging & Monitoring Requirements

- Logs include: ai_model, ai_model_version, prompt_template_id, confidence, latency_ms, decision_path
- Alerts:
  - AI parse failure rate > 2% (15 min)
  - AI latency p95 > 800ms (15 min) for user-facing inference paths

Acceptance Criteria

- 100% AI calls in the system route through AI Gateway (verified by egress allowlist + code review)
- Schema enforcement prevents invalid outputs from reaching user-facing flows
- Fallback used automatically on any gateway failure without breaking MVP flows

---

FR-14 — Natural Language Flight Search (NL→Query) (Phase 2+)

Description: Support free-text flight search queries (“cheap weekend next month DXB to IST, return Monday morning”) that are parsed into the existing structured search model used by FR-01.

User Story: As a traveler, I want to describe my flight intent in plain language so that I can search faster without manually filling all fields.

Preconditions

- Airport/city dataset available for validation and disambiguation
- AI Gateway (FR-13) available with NL parser schema

Functional Behavior

- UI supports:
  - NL input box with example chips + “Try natural language” placeholder
  - Display parsed parameters in a structured panel (origin/destination/dates/pax/cabin/constraints) with edit controls
- Backend supports:
  - Parse NL query into validated SearchQueryDraft JSON
  - If ambiguous: return multiple suggestions (airport/city disambiguation) requiring user selection
  - After validation: proceed with standard POST /api/search/sessions (FR-01)

- Parsing supports (minimum):
  - Origin/destination; one-way/round-trip; date or date window; passenger counts; cabin
  - Constraints: max stops, layover duration, time-of-day (“after 6pm”), budget (if provided)
  - Locale/currency aware parsing

Edge Cases

- Ambiguous city with multiple airports → require selection; do not guess silently
- Invalid date statements (“next month 31st”) → return correction prompt + fallback to calendar
- Missing origin/destination → request completion (UI prompt), do not call suppliers

Validation Rules

- Never fabricate airports: codes must match dataset
- Dates must be ISO after validation; depart <= return for round-trip
- Pax rules: adults>=1, infants<=adults, total_pax<=9

API Contract Implications

- POST /api/search/parse-nl
  - body: { queryText, locale, currency, userContext?: { homeAirport? } }
  - response: { status: OK|NEEDS_CLARIFICATION|FAILED, parsed: SearchQueryDraft?, suggestions?, errors? }
- Existing POST /api/search/sessions remains canonical execution path

Data Model Implications

- NLQueryEvent (optional): store nl_query_hash, parse outcome, corrections applied

Security Considerations

- Treat NL query as untrusted input; sanitize; detect injection patterns
- No PII collection inside NL; if present (email/phone), redact and ignore

Logging & Monitoring Requirements

- Log: nl_parse_request, nl_parse_success, nl_parse_fail, nl_parse_needs_clarification, correction actions
- Metric: parse success rate; correction rate; time-to-search

Acceptance Criteria

- ≥ 95% of NL queries produce a validated draft or explicit clarification request (no silent failures)
- Users can always fall back to classic search form without losing progress
- No supplier calls occur from NL parsing without validated structured params

---

FR-15 — Personalized “Best Value” Ranking (Learning-to-Rank) (Phase 2+)

Description: Rank offers based on predicted utility rather than price only. Ranking objective combines conversion probability, expected revenue yield, and quality/trust penalties while respecting deterministic constraints.

User Story: As a traveler, I want the default ordering to show the best overall option for me, not just the cheapest.

Preconditions

- Event instrumentation (impressions, clicks, conversions) in place (FR-10)
- Provider quality signals available (FR-20) and mismatch signals available (FR-22)

Functional Behavior

- Ranking outputs:
  - Default sort: Recommended (best value)
  - Deterministic alternative sorts remain available: Cheapest/Fastest/Departure time
- Ranking approach:
  - v1 heuristic scorer (weights) with feature store readiness
  - v2 ML/LTR model (LambdaMART/XGBoost rank or equivalent) using historical events
- Inputs (minimum):
  - Offer: price, duration, stops, depart/arrive times, baggage/refundability, airline, provider
  - User/session: device, locale, prior preferences (if logged-in & consented)
  - Provider: quality score, latency, mismatch rate, historical conversion
- Hard constraints (policy):
  - If mismatch risk above threshold → demote or label (cannot be #1 unless no alternatives)
  - Sponsored offers must pass relevance and quality floors (FR-23)

Edge Cases

- Cold start routes with sparse data → fall back to heuristic scorer
- Missing fields (baggage/refundability) → no hallucinated assumptions; reduce score gracefully

Validation Rules

- Sorting must be stable and deterministic given the same inputs
- Ranking must be reproducible with logged feature vector snapshots for audits

API Contract Implications

- Search poll response adds optional fields:
  - rank_score, rank_reason_tags[], quality_flags[], sponsored_flag
- Admin endpoint: /api/admin/ranking/weights (feature-flagged; governance controlled)

Data Model Implications

- OfferImpressionEvent, RankingDecisionLog (or reuse AIInferenceLog with use_case=ranking)
- Feature store tables (see Section 9 additions)

Security Considerations

- Ranking parameters changes audited; RBAC gating
- Sponsored constraints enforced server-side (no client override)

Logging & Monitoring Requirements

- Log: ranking version, weights/model id, NDCG@k (offline), CTR lift (online)
- Alert: mismatch rate or complaint rate spike after ranking changes (stop-loss)

Acceptance Criteria

- Recommended sort improves CTR ≥ +3% in controlled experiment without increasing mismatch rate > +0.5% absolute
- All alternative sorts remain correct and unchanged by ranking rollout

---

FR-16 — Offer Explanations & Inclusion Tags (Phase 2+)

Description: Provide grounded explanations for offers (“why recommended”, “what’s included”) and extract structured tags (baggage included, refundable, short layover) to increase trust and reduce decision friction.

User Story: As a traveler, I want to understand why an option is recommended and what is included so that I can choose confidently.

Preconditions

- Fare rules / inclusion attributes available from suppliers where possible (FR-02 normalization)
- AI Gateway (FR-13) for summarization with strict grounding rules

Functional Behavior

- Inline tags on cards (deterministic when possible):
    - “Includes 20kg checked bag”, “Non-stop”, “Refundable”, “Short layover”
- Expandable explanation panel:
    - 3–5 bullets max, referencing measurable signals only (duration, stops, baggage, price vs median)
    - If confidence low or signals missing → “Not enough data to explain”
- AI is used only for summarization; facts must be taken from normalized data and route stats

Edge Cases

- Supplier missing baggage/refundability → show “Not provided by supplier”
- Contradictory fare rules → show safest message + omit uncertain claim

Validation Rules

- Explanations cannot introduce new numeric values not present in data
- Explanation output must pass schema: { tags[], bullets[], confidence }

API Contract Implications

- GET /api/offers/{offerId}/explanation
    - response: { tags[], bullets[], confidence, sources: [data_fields_used] }

Data Model Implications

- OfferExplanationCache (optional) keyed by offer hash + locale

Security Considerations

- No PII in explanation generation
- Protect against prompt injection in supplier text fields by stripping HTML and unsafe tokens

Logging & Monitoring Requirements

- Metric: explanation availability rate; CTR lift; bounce reduction; support tickets trend

Acceptance Criteria

- Explanations render for ≥ 80% of offers where required fields exist; otherwise safe fallback shown
- No explanation contains unsupported facts (validated via automated tests with fixtures)

——————————————————————————————

FR-17 — Price Prediction Guidance (“Buy / Wait / Uncertain”) (Phase 2+)

Description: Provide probabilistic guidance to users about price movement for a route/date and drive alert subscriptions. This is advisory only and must never be positioned as a guarantee.

User Story: As a traveler, I want help deciding whether to book now or wait so I can reduce regret and plan better.

Preconditions

- Price history store exists (Section 9 additions) for route/date series
- Prediction service available (may be heuristic baseline initially)

Functional Behavior

- Display guidance module on:
    - Route SEO pages and/or results page for applicable searches
- Output:
    - Recommendation: BUY / WAIT / UNCERTAIN
    - Confidence: Low/Medium/High
    - Short reason string grounded in history (volatility, seasonality, days-to-departure)
- CTA:
    - If uncertain or wait: offer to “Track price” (ties to alerts)

Edge Cases

- Sparse historical data → show “Uncertain” with low confidence; default CTA to alerts
- Extreme volatility (sale periods) → cap confidence and show warning banner

Validation Rules

- Must include confidence; never claim certainty or guaranteed savings
- Reasons must reference data-derived signals only (no external claims)

API Contract Implications

- GET /api/predictions/price?origin=&dest=&depart=&return?
  - response: { recommendation, confidence, reason, model_version, computed_at }

Data Model Implications

- RoutePriceHistoryDaily, PricePredictionEvent

Security Considerations

- No user PII required; route-level only
- Predictions are cached; rate limit public endpoint

Logging & Monitoring Requirements

- Track prediction views, alert opt-ins, booking outcomes (where postback exists)
- Offline evaluation: MAE/MAPE + directional accuracy

Acceptance Criteria

- Prediction module never blocks results rendering; falls back gracefully
- All outputs include confidence and disclaimers; verified in UI tests

---

FR-18 — Smart Alerts v2 (Meaningful Notifications) (Phase 2+)

Description: Upgrade alerts to reduce notification fatigue and increase alert-to-booking conversion using meaningful-change thresholds and suppression rules.

User Story: As a user, I want fewer but more meaningful alerts so I don’t ignore them.

Preconditions

- Alerts v1 (FR-06) live; notification service (FR-11) live

- Price history and/or volatility signals available (FR-17)

Functional Behavior

- Trigger logic:
  - Baseline rule still supported (absolute drop threshold)
  - Add “meaningful change” threshold based on route volatility bands
- Suppression:
  - Quiet hours
  - Frequency caps (max 2/day/user, plus weekly digest option)
  - De-duplicate similar alerts across flexible date windows
- Personalization (optional):
  - Route suggestions (nearby airports, alternate dates) where supported by cached fare grid

Edge Cases

- No current price available → skip notification
- Multiple drops in short window → send summary digest instead of spam

Validation Rules

- Email verification required; push only with valid token + opt-in
- All notifications include unsubscribe/opt-out links per compliance

API Contract Implications

- POST /api/alerts extended with:
  - mode: BASIC|SMART, quiet_hours, digest_opt_in
- Admin: notification templates and caps configurable (audited)

Data Model Implications

- PriceAlertRule, AlertEvaluationEvent, NotificationSuppressionState

Security Considerations

- GDPR/DPDP compliance; consent logs for notifications

Logging & Monitoring Requirements

- Alert evaluation latency, trigger rates, unsubscribe rates, downstream clicks/bookings

Acceptance Criteria

- Smart alerts demonstrate lower unsubscribe rate vs v1 without reducing alert CTR
- Frequency caps and quiet hours enforce correctly (automated tests)

———————————————————————————————

FR-19 — Supplier Call Optimization (Fan-out Planner) (Phase 2+)

Description: Optimize which suppliers are called per search to reduce cost/search and latency while maintaining result coverage and competitiveness.

User Story: As the platform, I want to avoid wasted supplier calls so that unit economics improve without harming user experience.

Preconditions

- Supplier performance telemetry available (latency, timeouts, competitiveness)
- Orchestrator supports supplier plan injection (FR-02)

Functional Behavior

- Pre-search cache probe remains first (FR-02)
- Fan-out planner produces:
    - ordered supplier list
    - concurrency budget and per-supplier timeout budget
    - exploration rate (sample lower-used suppliers to refresh priors)
- Planner inputs:
    - route/date/cabin features, historical coverage likelihood, cost-per-call, provider quality, expected yield
- Planner outputs are constrained by a global call budget per search

Edge Cases

- New route with no priors → fall back to default supplier set
- Supplier outage → planner excludes automatically based on health state

Validation Rules

- Planner may not reduce below minimum supplier count for MVP coverage (configurable)

- Coverage regression must be monitored with stop-loss rollback

API Contract Implications

- Internal: GET /internal/suppliers/plan?searchSessionId=... returns plan

Data Model Implications

- SupplierPlanDecision, SupplierCoveragePrior, SupplierRouteStats

Security Considerations

- No PII inputs required; route/session-level only

Logging & Monitoring Requirements

- Calls per search, p95 latency, coverage metrics, revenue per infra dollar

Acceptance Criteria

- Reduces calls/search vs baseline by measurable delta without reducing coverage > 1% absolute

---

FR-20 — Provider Quality Scoring & Suppression v2 (Phase 2+)

Description: Compute a quality score for each provider and apply suppression/demotion rules to protect trust and conversion.

User Story: As a traveler, I want to avoid providers that often change prices or have poor handoff experiences.

Preconditions

- Redirect and postback tracking operational (FR-04/05/10)
- Issue signals captured (price mismatch, redirect errors, complaints)

Functional Behavior

- Score inputs:
    - mismatch rate, redirect error rate, conversion rate, latency, complaint tags, postback discrepancies

- Score outputs:
  - quality score (0–100) and tier (A/B/C/D)
- Actions:
  - demote, label (“Low reliability”), throttle, or suppress
  - stricter verification for low-quality providers (force verify-before-redirect)
- Admin controls:
  - view score history, override (time-bound), suppression reason codes (audited)

Edge Cases

- Small sample size providers → conservative scoring (wide confidence intervals)
- Sudden provider degradation → rapid suppression (incident mode)

Validation Rules

- Suppression rules must be explainable and auditable; no silent suppression without reason code

API Contract Implications

- GET /api/admin/providers/{id}/quality
- Search results include provider_quality_tier per provider option

Data Model Implications

- ProviderQualityScoreDaily, ProviderSuppressionRule, ProviderIncident

Security Considerations

- RBAC gating; audit logs for overrides

Logging & Monitoring Requirements

- Alert on mismatch spikes by provider; suppression action logs

Acceptance Criteria

- Providers with sustained mismatch above threshold are suppressed within 24 hours (or faster in incident mode)
- User-visible labeling is consistent with policy

FR-21 — Fraud / Invalid Traffic (IVT) Detection (Phase 2+)

Description: Detect and mitigate invalid clicks and suspicious traffic patterns to protect ad budgets and partner trust.

User Story: As the business, I want to prevent click fraud so revenue and partner settlements remain credible.

Preconditions

- Event taxonomy includes click + postback linkage (FR-05/10)

Functional Behavior

- Risk scoring per session/click using:
    - click bursts, abnormal IP/ASN/device patterns
    - impossible geo movement, repetitive paths
    - mismatch between clicks and postbacks
- Actions:
    - throttle, challenge (captcha), block, exclude from billing, flag for review
- Admin UI:
    - suspicious traffic dashboard, rule tuning, export for partner disputes

Edge Cases

- Shared IP environments (mobile carriers) → avoid false positives via multi-signal scoring
- Partners missing postbacks → rely more on behavior + latency + dwell time signals

Validation Rules

- Risk decisions must be logged with features used and action taken
- “Exclude from billing” requires admin governance config

API Contract Implications

- GET /api/admin/fraud/overview, GET /api/admin/fraud/clicks?risk>=...

Data Model Implications

- FraudRiskEvent, FraudRule, FraudActionLog

Security Considerations

- Do not expose full IPs to non-admin roles; store hashed or truncated where possible

Logging & Monitoring Requirements

- Invalid click rate, false positive review outcomes, partner dispute rate

Acceptance Criteria

- Fraud detection reduces partner disputes or invalid click rate trend without significant user friction increase

---

FR-22 — Price Mismatch Prediction & "Price Stability" Labels (Phase 2+)

Description: Predict probability that a selected offer price will change on the provider site, and surface a "price stability" label to manage expectations and reduce support load.

User Story: As a traveler, I want to know if a price is likely to change before I click out so I can avoid frustration.

Preconditions

- Price verification outcomes logged (FR-04) and mismatch signals stored
- Provider quality scoring available (FR-20)

Functional Behavior

- Compute mismatch_risk for each provider option using:
    - provider mismatch history, offer volatility, cache age, freshness of verification
- UX:
    - label "Price stable" vs "Price likely to change"
    - if high risk: encourage refresh/verify before redirect
- Ranking integration:
    - high risk offers demoted unless no alternatives; still visible with label

Edge Cases

- No mismatch history → label "Unknown"; do not claim stability

Validation Rules

- Labels must be derived from model outputs + thresholds; thresholds configurable and audited

API Contract Implications

- Search results include mismatch_risk and stability_label per provider option

Data Model Implications

- MismatchModelScore, PriceVerifyEvent

Security Considerations

- No PII required

Logging & Monitoring Requirements

- Track mismatch incidents vs predicted risk; calibration

Acceptance Criteria

- High-risk label correlates with higher observed mismatch rate in offline evaluation (directional correctness)

FR-23 — Sponsored Placement Optimizer (Relevance + Yield) (Phase 2+)

Description: Optimize sponsored offer selection/ordering to maximize monetization while enforcing relevance and trust constraints.

User Story: As the business, I want sponsored placements that increase revenue without harming user trust.

Preconditions

- Sponsored placements MVP available (Phase 2 FR-P2-02)
- Provider quality scoring (FR-20) and fraud controls (FR-21) available

Functional Behavior

- Sponsored candidate selection based on:
  - predicted CTR, predicted conversion, expected CPC/CPA, relevance score, provider quality score
- Guardrails:
  - relevance floor required
  - quality floor required
  - frequency caps per session/user
  - clear “Sponsored” labeling (never ambiguous)
- Reporting:
  - sponsored impressions, CTR, revenue, complaint rates; partner-level breakdown

Edge Cases

- Insufficient eligible sponsored offers → no sponsored placement shown (do not force fill)

Validation Rules

- Sponsored must never override user-selected deterministic sorts without disclosure; “Sponsored” module is separate slot or clearly integrated

API Contract Implications

- /api/admin/sponsored/* includes campaign config + performance reporting

Data Model Implications

- SponsoredImpressionEvent, SponsoredClickEvent, Campaign, BudgetSpendEvent

Security Considerations

- Governance workflow for campaign approvals; RBAC and audit logs

Logging & Monitoring Requirements

- Alerts on complaint rate spikes or relevance floor violations

Acceptance Criteria

- Sponsored module is always labeled; relevance floor enforced server-side
- Sponsored revenue increases without decreasing retention/CSAT beyond guardrails

————————————————————————————————

FR-24 — Support Copilot (Session-Aware) (Phase 2+)

Description: Admin tool that summarizes a user's session timeline and drafts grounded support responses using internal facts (click_id, timestamps, error codes).

User Story: As support staff, I want faster, more accurate responses so I can resolve issues with fewer escalations.

Preconditions

- Session timeline data available (search → click → verify → redirect → postback)

Functional Behavior

- Given click_id or searchSessionId, copilot shows:
  - timeline view, key events, mismatches, provider details
  - suggested response drafts + recommended next steps
- Strict separation of:
  - user-visible response
  - internal-only notes

Edge Cases

- Missing postback data → copilot must state "not confirmed" and avoid assumptions

Validation Rules

- Copilot responses must cite internal fact IDs used (event ids, timestamps)

API Contract Implications

- POST /api/admin/support/copilot (RBAC protected)

Data Model Implications

- SupportCopilotQueryLog (redacted)

Security Considerations

- Role-based access; PII masking; audit logs

Logging & Monitoring Requirements

- AHT, resolution rate, escalation rate

Acceptance Criteria

- Copilot never claims booking status unless confirmed via postback event

---

FR-25 — Partner Onboarding Copilot (Phase 2+)

Description: Internal tool to accelerate new partner integrations by producing field mappings, integration checklists, and QA test cases from partner API docs.

User Story: As engineering/ops, I want integrations to be faster and less error-prone.

Functional Behavior

- Upload/link partner API docs (internal only)
- Output:
    - mapping template to platform canonical schema
    - checklist for deeplinks, price verify, postbacks
    - sample request/response stubs
    - QA test plan for edge cases

Validation Rules

- Outputs are drafts; require human review and sign-off
- Tool cannot publish to production configs automatically

Security Considerations

- Partner docs are confidential; store securely; limit access

Acceptance Criteria

- Generated checklist covers: search, normalization, deeplink construction, price verify, attribution, error taxonomy, rate limits

FR-26 — AI-Assisted SEO at Scale (Phase 2+)

Description: Generate route/city page blocks and FAQs using template-first approach grounded in internal aggregated stats to avoid thin content.

User Story: As the business, I want to scale SEO pages safely without producing low-quality content.

Functional Behavior

- Template definitions:
  - required data-driven blocks (price bands, seasonality, schedules where available)
  - optional narrative blocks with strict grounding
- Workflow:
  - AI fills template placeholders from approved data sources
  - human review required for new templates and high-traffic pages
- Safeguards:
  - minimum unique content rule per page
  - canonicalization and duplication detection

Acceptance Criteria

- No page can be published without required data blocks present and schema validation passing

FR-27 — Analytics Copilot in Admin (Phase 2+)

Description: Role-gated “ask-your-data” interface that answers questions over approved metrics, suggests actions (ranking weight changes, provider suppression candidates), and cites dashboards/metric IDs used.

User Story: As an operator, I want to understand what changed in KPIs without writing SQL.

Functional Behavior

- Use cases:
  - “Why did CTR drop for DXB→IST in last 7 days?”
  - “Which provider had highest mismatch rate yesterday?”

- Guardrails:
  - only query approved semantic layer
  - citations to metric IDs; no raw PII exposure
  - outputs are suggestions; changes require admin approval workflow

Acceptance Criteria

- Copilot cannot execute changes; it can only propose and link to relevant admin controls

---

Phase 2 Feature Specs (Scope-Frozen)

> *These are **scope-frozen Phase 2** outcomes (commercial + CRO acceleration) and their **AI Workstream** implementation references.*
>
> *AI-driven components are **behind feature flags** and must follow "AI proposes, system decides" with deterministic fallbacks.*

FR-P2-01 — Best Value Ranking + Explainability (Refs: FR-15, FR-16, FR-20, FR-22)

- **Objective**: Improve CTR and downstream booking yield by ranking on utility (not just price) while protecting trust.
- **User-visible**:
  - Default "Recommended" sort uses best-value ranking.
  - Each recommended card shows 1–3 explanation tags (e.g., "Short layover", "Includes baggage").
  - If "price likely to change", label is shown and ranking penalizes high-risk offers.
- **System behavior**:
  - Ranking model versioned; all decisions logged; stable alternative sorts preserved.
  - Stop-loss guardrails: rollback if mismatch spikes or complaint rate increases.
- **Acceptance criteria**:
  - CTR lift ≥ +3% vs control (A/B) with mismatch increase ≤ +0.5% absolute.
  - Explanation tags never contain unsupported facts (fixture validation).

FR-P2-02 — Sponsored Placements Marketplace (Refs: FR-23, FR-21, FR-20)

- **Objective**: Increase monetization yield without harming relevance or trust.
- **User-visible**:
  - Sponsored slots are always labeled "Sponsored".
  - Sponsored offers must remain relevant to the search query and meet quality floors.

- **System behavior**:
    - Campaign configuration + budgets + reporting in Admin.
    - Fraud/IVT controls exclude suspicious clicks from billing where contractually supported.
- **Acceptance criteria**:
    - Sponsored revenue increases while retention and complaint rate remain within guardrails.
    - No “forced fill” when no eligible sponsored offers exist.

FR-P2-03 — Partner/Provider Quality Scoring + Suppression (Refs: FR-20, FR-22)

- **Objective**: Protect trust and improve conversion by reducing exposure to unreliable providers.
- **System behavior**:
    - Daily quality scores + incident mode for rapid degradation.
    - Suppression and labeling rules with audit logs and override workflow.
- **Acceptance criteria**:
    - High-mismatch providers suppressed within 24h (or faster under incident mode).
    - All suppressions have reason codes and are reviewable.

FR-P2-04 — Fare Calendar (Whole-Month View)

- **Objective**: Increase flexible-date conversions and reduce search friction for price-sensitive users.
- **Behavior**:
    - Month grid shows lowest fare per day (cached/derived from aggregated supply).
    - Clicking a day launches a structured search session (FR-01) with selected dates.
- **Acceptance criteria**:
    - Month grid loads within 1.0s p95 from cache for popular routes.

FR-P2-05 — Explore / Anywhere Discovery

- **Objective**: Improve engagement for users without a fixed destination.
- **Behavior**:
    - User provides origin and date range/budget; system suggests destinations and cheapest fares.
    - Must be grounded in cached/aggregated pricing (no hallucinations).
- **Acceptance criteria**:
    - All suggested destinations link to valid route pages/search sessions.

FR-P2-06 — Natural Language Search (Refs: FR-14)

- **Objective**: Reduce top-of-funnel friction; increase search completion rate.

- **Acceptance criteria**:
    - Parse success (validated draft or explicit clarification) ≥ 95%; otherwise safe fallback.

FR-P2-07 — Price Prediction + "Price Stability" Labels (Refs: FR-17, FR-22)

- **Objective**: Drive higher-quality clicks and alert opt-ins with confidence-based guidance.
- **Acceptance criteria**:
    - Predictions always show confidence and disclaimers; never block results rendering.

FR-P2-08 — Supplier Call Optimization (Refs: FR-19)

- **Objective**: Reduce supplier cost/search while preserving coverage and competitiveness.
- **Acceptance criteria**:
    - Calls/search reduced with coverage drop ≤ 1% absolute (stop-loss enforced).

FR-P2-09 — Fraud / IVT Detection (Refs: FR-21)

- **Objective**: Protect CPC/CPA economics and partner trust.
- **Acceptance criteria**:
    - IVT actions are logged and reviewable; false positives tracked and tuned.

FR-P2-10 — Internal Ops Copilots (Refs: FR-24, FR-25, FR-26, FR-27)

- **Objective**: Scale operations without linear headcount growth.
- **Acceptance criteria**:
    - Copilots are role-gated; do not execute changes automatically; outputs are grounded and auditable.

---

7. Non-Functional Requirements (Strict)

7.1 Performance

- API (non-search): p95 ≤ **500ms**
- Search:
    - first chunk ≤ **800ms p95**
    - hard cutoff ≤ **2.5s p95**
- Page load: ≤ **3.0s** p75 LCP (mobile)

7.2 Capacity / Concurrency

- Peak POST /search/sessions: **200 RPS** burst (60s), **80 RPS** sustained
- Concurrent active search sessions (polling): **10,000**
- Supplier fanout budget: max **3 suppliers** per search in MVP average (route-budgeted; enforced at orchestrator)

7.3 Caching Strategy

- Search results cache TTL: **10 minutes**
- Autocomplete cache TTL: **7 days**
- Route page cache TTL: **24 hours** (CDN), purge on publish

7.4 Scalability

- Horizontal scaling via stateless API services behind load balancer and CDN
- Stateful systems: Postgres, Redis, Elasticsearch (cluster/managed)

7.5 Reliability

- Uptime ≥ **99.9%**
- Retry strategy:
    - supplier calls: max **1 retry** for idempotent searches
    - postback: idempotent; accept duplicates
- Circuit breaker: open after **25% failures in 60s**, cooldown **90s**

7.6 Security

- OWASP Top 10 security audit passed before go-live
- Encryption:
    - In transit: TLS 1.2+
    - At rest: AES-256 for DB volumes and backups
- Auth:
    - Users: OAuth + email login
    - Admin: RBAC + MFA (mandatory)

- AI security (Phase 2+):

- All model calls must go through AI Gateway (FR-13) with redaction + schema enforcement
- Prompt injection protection for any user/supplier text inputs; sanitize HTML and strip unsafe tokens
- Model endpoints protected with service auth (mTLS or signed tokens) + outbound allowlist
- Rate limit AI endpoints separately from core APIs; fail closed to deterministic fallback

7.7 Compliance

- GDPR/DPDP compliance controls included
- Data retention:
    - SearchQuery (anon): 13 months
    - RedirectEvent / CommissionEvent: 24 months
    - AuditLog: 24 months
    - PII: retained until deletion; deleted/anonymized within 30 days of request
- Cookie consent logic: required prior to non-essential tracking

7.8 Observability

- Logs: structured JSON; include trace_id, searchSessionId, partnerId, click_id
- Metrics: latency, error rate, cache hit ratio, CTR, mismatch rate, postback match rate
- Alerts:
    - Search error rate > 2% (5 min)
    - Supplier timeout rate > 30% for any tier-1 supplier (5 min)

- AI/ML observability (Phase 2+):
    - Log fields (minimum): ai_model, ai_model_version, prompt_template_id, input_hash, output_hash, confidence, latency_ms, decision_path
    - Metrics: AI fallback rate, schema-parse failure rate, model latency p95, ranking CTR delta by model version
    - Alerts:
        - AI parse failure rate > 2% (15 min)
        - AI latency p95 > 800ms (15 min) for user-facing inference endpoints

7.9 AI Governance & Safety (Phase 2+)

- Principle: **deterministic core, probabilistic assist**; AI proposes, system decides
- No hallucinations on transactional surfaces:
    - Any user-facing “facts” must be derived from supplier responses, fare rules, verified aggregates, or explicit user inputs

- Schema enforcement:
  - Structured AI outputs validated against JSON schema; invalid outputs rejected with deterministic fallback
- Traceability:
  - All AI decisions logged with model version + confidence + downstream impact
- Privacy-first:
  - Do not send raw PII to third-party models unless explicitly approved; tokenize/redact by default

---

8. Technical Architecture Alignment

8.1 Confirmed Tech Stack (Baseline)

- Frontend: Next.js 14+ (React) + TypeScript; Tailwind CSS
- Backend: Node.js (NestJS/Express) **or** Python FastAPI (choose one for MVP; do not mix)
- Data: PostgreSQL + Redis + Elasticsearch
- Queue: RabbitMQ/Bull (async aggregation + notifications)
- CDN: Cloudflare/CloudFront; Hosting: AWS/GCP
- Observability: Sentry/Datadog/CloudWatch
- Analytics: GA4, Mixpanel

8.2 Service / Module Mapping (No Redesign Beyond Confirmed Needs)

- **Frontend (Next.js)**
  - Search UI + Results UI + Route pages (SSR/ISR) + Account UI + Admin UI
- **Backend APIs**
  - Search Session Service
  - Orchestrator / Aggregation Engine
  - Partner Adapter Layer
  - Attribution Service
  - Alerts + Notification Service
  - Admin APIs
  - AI Gateway Service (Phase 2+) — redaction, schema enforcement, audit logs
  - Ranking Service (Phase 2+) — best-value ranking (heuristic + ML/LTR)
  - Prediction Service (Phase 2+) — price movement and mismatch risk
  - Quality Service (Phase 2+) — provider scoring + suppression rules
  - Fraud / IVT Service (Phase 2+) — risk scoring + actions
- **Data**

- Postgres: core entities
- Redis: cache/session state
- Elasticsearch: autocomplete
- Feature Store (Phase 2+): user/route/provider/offer features for ranking and predictions
- AI Inference Logs (Phase 2+): immutable audit trail for AI decisions
- Price history store (Phase 2+): route/date series for forecasting

———————————————————————————————

9. Data Model Requirements

9.1 Core Entities (Required Fields + Indexing + Retention)

User

- Fields: id, email, email_verified, created_at, status
- Index: email unique
- Retention: until deletion request + 30 days

SearchQuery

- Fields: id, anon_session_id, user_id?, origin, destination, depart_date, return_date?, pax_json, cabin, currency, locale, created_at
- Index: (origin, destination, depart_date), created_at
- Retention: 13 months

FlightOption

- Fields: id, searchSessionId, itinerary_hash, segments_json, stops, duration, carrier_codes, depart_ts, arrive_ts
- Index: searchSessionId, itinerary_hash

FareDetail

- Fields: id, flightOptionId, partnerId, total_price, currency, base, tax, fees, baggage_json, deeplink_url, refundable_flag
- Index: (flightOptionId, total_price), partnerId

Partner

- Fields: id, name, type (OTA/GDS/Airline), status, commission_rules, deeplink_template, auth_config_ref
- Index: status

RedirectEvent

- Fields: id, click_id, searchSessionId, partnerId, flightOptionId, price_at_click, currency, created_at, user_id?
- Index: click_id unique, (partnerId, created_at)

CommissionEvent

- Fields: id, partnerId, click_id, conversion_id, status, amount, currency, booked_at, reported_at, raw_payload_ref
- Index: conversion_id unique, click_id, (partnerId, booked_at)
- Retention: 24 months

PriceAlert

- Fields: id, userId, route, date_window, baseline_price, threshold, frequency_cap, status, created_at
- Index: (userId, status)

RoutePage

- Fields: id, origin, destination, canonical_url, title, meta_desc, schema_json, status, published_at
- Index: canonical_url unique, (status, published_at)

AdminUser

- Fields: id, email, role, mfa_enabled, status, last_login_at
- Index: email unique

AuditLog

- Fields: id, actor_admin_id, action, entity_type, entity_id, before_json, after_json, created_at
- Index: (entity_type, entity_id), created_at
- Retention: 24 months

———————————————————————

9.2 AI/ML Data Entities (Phase 2+)

> *These entities are **additive** and required to support the AI Workstream features while maintaining auditability and privacy.*

SearchEvent (Event Stream / Warehouse)

- Fields: event_id, event_type, searchSessionId, user_id? (pseudonymous), anon_session_id, timestamp, route, dates, pax, cabin, currency, locale, properties_json
- Index: (event_type, timestamp), searchSessionId
- Retention: 24 months (aggregated warehouse); raw events may be shorter per policy

OfferImpressionEvent

- Fields: event_id, searchSessionId, offer_id, provider_id, rank_position, price_total, currency, sponsored_flag, timestamp
- Index: (provider_id, timestamp), (searchSessionId, rank_position)

PriceVerifyEvent

- Fields: id, click_id, provider_id, offer_id, price_before, price_after, mismatch_flag, verified_at, verification_latency_ms
- Index: click_id, (provider_id, verified_at)

ProviderQualityScoreDaily

- Fields: provider_id, score, tier, mismatch_rate_7d, redirect_error_rate_7d, conversion_rate_7d, computed_at
- Index: (provider_id, computed_at)
- Retention: 24 months

FraudRiskEvent

- Fields: id, click_id?, anon_session_id, risk_score, signals_json, action_taken, created_at
- Index: created_at, (risk_score, created_at)

AIInferenceLog (Immutable)

- Fields: id, use_case, model, model_version, prompt_template_id, input_hash, output_hash, confidence, latency_ms, decision_path, trace_id, created_at
- Index: (use_case, created_at), trace_id
- Retention: 24 months (or as approved by Legal/Security)

FeatureStore (Logical)

- **User features**: airline affinity, departure-time preference, stop tolerance (consent-gated)
- **Route features**: typical price bands, seasonality, volatility
- **Provider features**: mismatch rate, latency distribution, conversion rate
- **Offer features**: price delta vs median, historical volatility indicators
- Storage: either dedicated tables or warehouse materializations with versioning

RoutePriceHistoryDaily

- Fields: origin, destination, depart_date, return_date?, min_price, median_price, price_p10, price_p90, currency, observed_at
- Index: (origin, destination, depart_date, return_date)
- Retention: 24 months (aggregated)

---

10. SEO & Programmatic Page Logic

Route Page Generation Logic (MVP)

- Route page exists when (origin, destination) is marked “SEO-enabled” in admin.
- Page content must include:
  - Search widget prefilled
  - Structured data blocks (FAQ/breadcrumbs schema)
  - Canonical + meta tags
- Sitemap regeneration on publish/unpublish.

Canonical Structure

- Canonical URL format: /flights/{origin}-{destination}
- Airport vs city disambiguation:
  - Prefer city pages; airport pages canonicalize to city route where applicable.

Internal Linking Structure

- City page links to top routes from that city.
- Route page links to origin city and destination city pages.

———————————————————————————————

11. Reporting & Analytics Requirements

11.1 Revenue Dashboard Metrics (Admin)

- Total searches, redirects, conversions, revenue (CPC/CPA)
- RPS (revenue per 1,000 searches)
- Revenue by partner, route, device

11.2 Partner Performance Metrics

- Redirect volume, conversion rate, mismatch rate, postback match rate
- SLA: latency p95, timeout %

11.3 Conversion Funnel Tracking

- Search → results impression → provider click → conversion
- Drop-off by route + device

11.4 Fraud Detection Signals (MVP baseline)

> *MVP must include baseline signals for monitoring. Phase 2+ extends this into a full IVT scoring and action framework (FR-21).*

Baseline signals (MVP)

- High click volume from single IP / ASN
- Abnormal click-to-conversion ratio by traffic source
- Rapid repeat clicks with zero dwell time
- Unusual spike in redirects for a single route/provider combination (burstiness)

Phase 2+ signals (for scoring)

- IP/device repetition across many sessions; impossible geo movement patterns
- High mismatch between clicks and postbacks by cohort
- Excessive “refresh/verify” attempts suggesting automation
- Partner redirect error bursts (suggesting bot probing)

Required actions (Phase 2+ enablement)

- Throttle / block suspicious traffic

- Challenge flows (e.g., captcha) behind feature flag
- Mark clicks as IVT and exclude from billing where contractually supported
- Flag for admin review with exportable evidence bundle

11.5 Data Export Requirements

- CSV exports for partner reconciliation:
  - clicks by day/partner
  - conversions by day/partner
  - variance report

11.6 AI/ML Model Monitoring & Experimentation (Phase 2+)

- Feature flags:
  - Cohort-based rollout for ranking, NL search, explanations, predictions, sponsored optimizer
- A/B testing:
  - Persistent assignment per user/session
  - Pre-registered success metrics + guardrails (stop-loss on mismatch spikes)
- Offline evaluation:
  - Ranking: NDCG@k, MAP, calibration of predicted conversion
  - Predictions: MAE/MAPE + directional accuracy
  - Fraud: precision/recall tradeoffs + cost-weighted evaluation
- Model version tracking:
  - All online metrics must be segmented by `model_version` / `ranking_version`

11.7 AI Decision Logging Requirements (Phase 2+)

- For each AI-influenced decision (ranking, labels, explanations, predictions, copilots):
  - store `prompt_template_id` (or `model_config_id` for non-LLM models)
  - store input/output hashes (redacted)
  - store confidence + latency + decision_path (`ai_applied` / `fallback_used`)
  - link to the user-visible event trail (`searchSessionId`, `click_id`, `trace_id`)
- Audit use-cases:
  - investigate a mismatch incident
  - validate sponsored relevance floors
  - reproduce ranking decisions for partner disputes

————————————————————————————————

12. Risk & Mitigation (Execution-Focused)

Risk	Impact	Likelihood	Mitigation	Owner
---	---	---:	---	---
GDS/OTA API delays or rejection	Blocks supply coverage	Medium	Start partnerships early; maintain fallback providers	Partnerships Lead
Third-party API changes	Breaks search/redirect	Medium	Modular adapter layer to swap providers	Eng Lead
Performance under load	User drop + SEO hit	Medium	Load testing + autoscaling	DevOps Lead
Scope creep	Timeline/cost overrun	Medium	Formal change request process	TPM
Data privacy non-compliance	Legal + reputational	Low–Med	GDPR/DPDP built-in controls	Security/Legal
AI hallucinations on transactional surfaces	Trust loss + legal disputes	Medium	Grounded outputs + schema enforcement + deterministic fallback (FR-13/16/17)	Product + Eng
Prompt injection / adversarial inputs	Security incident / data leakage	Medium	Input sanitization + AI Gateway policy layer + allowlists	Security
Model latency/outage impacts UX	Increased bounce/CTR drop	Medium	Strict timeouts + fallback + caching for safe outputs	Eng + DevOps
Training/feature-store data quality issues	Poor ranking/predictions	Medium	Data quality checks + offline evaluation + stop-loss A/B guardrails	Data/ML Lead
AI changes create untraceable decisions	Partner disputes hard to resolve	Low–Med	Mandatory decision logging + model versioning	Data/ML Lead

			(Section 11.7)	

13. Phased Delivery Plan

- **Sprint length**: 2 weeks
- **MVP target**: 10 sprints (20 weeks) + 1 hardening sprint
- **Phase 2 target**: 6 sprints (12 weeks)

13.1 MVP Milestones

- **Sprint 1–2**: Foundation (CI/CD, DB schema, auth, frontend scaffold, admin foundation)
- **Sprint 3–5**: Core search + 1–2 supplier integrations + results filters/sorting + aggregation engine
- **Sprint 6–7**: Redirect tracking + user profiles/history + alerts + email notifications + trip dashboard
- **Sprint 8–9**: SEO route/city pages + schema + sitemap + analytics integration + performance optimization
- **Sprint 10**: Admin reports + content workflows + stabilization
- **Sprint 11 (Hardening)**: UAT, security audit, load testing, prod deploy, docs, KT

13.2 Phase 2 Milestones

- **Sprints 12–13**: Best value ranking explainability + partner quality scoring
    - Enable AI Gateway foundation (FR-13) for controlled inference + audit logs
    - Implement provider quality scoring v2 + suppression workflow (FR-20)
    - Implement supplier call optimization v1 (rules + priors) (FR-19)
- **Sprints 14–15**: Sponsored placements MVP (labeling, reporting, admin controls)
    - Add sponsored placement optimizer guardrails (FR-23)
    - Roll out fraud/IVT scoring baseline + admin dashboards (FR-21)
- **Sprint 16**: Fare calendar + Explore/Anywhere
    - Add offer explanations + inclusion tags (FR-16) where data available
    - Add price stability labels (FR-22) in results UI
- **Sprint 17**: Alerts digest/watchlist + experimentation ramp
    - Smart alerts v2 (suppression + meaningful thresholds) (FR-18)
    - Natural language search beta (FR-14) behind feature flag
    - Price prediction v1 (“buy/wait/uncertain”) beta (FR-17)
- **Sprint 18**: Optimization + QA regression + investor-ready reporting pack
    - Model monitoring + experimentation dashboards (Section 11.6)

- Support Copilot beta and Analytics Copilot beta (FR-24/FR-27) (role-gated)
- AI-assisted SEO pipeline beta (FR-26) with human review workflow

13.3 Release Criteria / UAT Conditions

- All in-scope features functional + tested
- Page load < 3s; API < 500ms
- Cross-browser: Chrome/Firefox/Safari/Edge; responsive 320–1920px
- OWASP audit passed
- Documentation delivered

13.4 Go-Live Checklist

- Prod infra + CDN + WAF enabled
- Partner creds configured; at least 1–2 suppliers live
- Analytics verified (funnels working)
- Alerts + email deliverability verified
- Rollback plan tested (blue/green or canary)

13.5 Rollback Plan

- CDN rollback to previous build
- API rollback via versioned deployment
- Feature flags: supplier enablement, sponsored placements (Phase 2)

---

14. Acceptance Criteria (Global)

System-Level Acceptance Criteria

- Performance thresholds met (Section 7)
- Browser compatibility and mobile responsiveness verified (Section 13.3)
- Security audit completion (OWASP Top 10) verified
- Documentation completeness verified
- Monitoring configured (logs, metrics, alerts) verified
- AI governance verified (Phase 2+ readiness):
    - AI Gateway enforced for all model calls; egress allowlist confirms no direct model calls
    - Structured AI outputs schema-validated with deterministic fallback (no malformed output reaches UI)
    - AI decision logging enabled with model versions and decision_path (Section 11.7)

- Attribution integrity:
  - Redirect events persisted for every click
  - ≥ 98% postbacks matched where click_id present
  - No open redirects (partner allowlist enforced)

———————————————————————————————

Before Engineering Kickoff — Final Validation Checklist

- [ ] PRD v1.0 signed by Product, Eng, QA, DevOps, Security
- [ ] MVP vs Phase 2 backlog mapped 1:1 to FR IDs (no orphan tickets)
- [ ] Supplier integration plan confirmed (which 1–2 providers for MVP)
- [ ] Tracking contract confirmed per partner (click_id + postback method)
- [ ] Data retention + deletion policy approved by Legal
- [ ] Observability dashboards and alert thresholds defined in monitoring stack
- [ ] Load test plan includes worst-case supplier timeout scenarios
- [ ] UAT plan includes redirect validation + attribution reconciliation

———————————————————————————————

Source Baseline (Internal)

- SOW v1.0 (Feb 2026) — baseline scope and architecture targets
- Feature Research and ROI Exploration — finalized MVP vs Phase 2 prioritization buckets
- Aggregation/Orchestration blueprint — deadline model guidance
- AI Capability Strategy & Detailed Feature Documentation v1.0 (Feb 2026) — Phase 2+ AI Workstream scope and guardrails