MANUAL TÉCNICO — SISTEMA DE LOCAIS PARCEIROS E EXPERIÊNCIAS COMERCIAIS (FRIENDAPP)

📘 CAMADA 0 — FILOSOFIA DE EXECUÇÃO (O FIO DA MEADA) — VERSÃO EXECUTÁVEL

> Objetivo da Camada: alinhar estratégia, ordem de execução, critérios de qualidade, segurança, observabilidade, antifraude e governança do Sistema de Locais Parceiros e Experiências Comerciais. Esta camada elimina retrabalho: define como construir, medir e aprovar cada parte antes de seguir.

0.1 Princípios Inegociáveis (North Star)

1. **Ordem de execução invariável**: **Dados → APIs → Lógicas (regras/IA/antifraude) → UI/UX → Integrações**.
2. **Contrato dirige código**: endpoints, payloads e erros são aprovados **antes** de implementar.
3. **Score vibracional = ativo regulado**: tratá-lo como "saldo financeiro" (imutabilidade, trilha de auditoria, antifraude, transparência).
4. **Privacidade e ética**: feedback é **anônimo por design**; nenhuma inferência identifica indivíduos.
5. **Observabilidade de produto**: cada evento crítico (check-in, feedback, reclassificação, impulsionamento, denúncia, suspensão) gera **log semântico** e **métrica de produto**.
6. **Resiliência por default**: escrita **assíncrona + idempotente**; leitura em **cache**; degradação graciosa.
7. **Cálculo assíncrono/pre-agregado**: recomputar scores e estados vibracionais em **jobs**; UI consome **snapshots** consistentes.
8. **Transparência para parceiros**: painel mostra **composição do score** e razões de variação.
9. **Segurança vibracional**: estados "Colapso" não são punidos; acionam **Missões de Harmonização** (gamificadas).
10. **Portabilidade**: schemas versionados; dumps reprodutíveis para staging/benchmarks.

0.2 Ordem de Execução (Fio da Meada)

1. **Modelagem de Dados (Cam. 08)**

- **PostgreSQL** (cadastro, contratos, tiers, assinaturas, eventos, auditoria).
- **Firestore** (check-ins/feedbacks em tempo real, painéis).
- **Neo4j** (malha de relações: usuários ↔ locais ↔ eventos).
- **Redis** (snapshots de score/estado, proximidade geográfica, rate-limits).

1. **Contratos de API (Cam. 09)**

- Especificação OpenAPI: rotas, payloads, erros, **Idempotency-Key**.
- **Escopos**: `user`, `partner`, `admin`, `system/ia`.
- **Rate limits** por rota.

1. **Lógica de Negócio e IA (Cam. 06/10/20/24/25)**

- **Pipelines** de cálculo (`score_vibracional`, `estado_vibracional`).
- **Antifraude** (detector de anomalias, reputação do usuário, quorum geográfico).
- **Missões** e **Sugestões da Aurah** (orientação data-driven).

1. **UI/UX (Cam. 07/12/21/22/23)**

- Painel B2B (parceiro) + App (usuário).
- Estados vazios, loading, degradação off-line.

1. **Integrações (Cam. 11/14/17/18/19/...)**

- Pagamentos (webhooks, revogação de acesso), CDN, mapas, push, e-mail.

0.3 Non-Functional Requirements (NFRs)

Domínio	Requisito	Meta
Disponibilidade	Uptime	**≥ 99,9%** mensal
Latência	GET painéis / mapa	p95 ≤ **300ms** (cache)
Latência	POST check-in/feedback	p95 ≤ **600ms** (fila assíncrona)
Consistência	Score exibido	Snapshot ≤ **15 min**
Segurança	TLS, JWT, RBAC, assinaturas de webhooks	Obrigatório
Privacidade	Pseudonimização, K-anonymity em agregados	Obrigatório
Observabilidade	Tracing, métricas, logs estruturados	100% rotas core
Escalabilidade	Autoscaling (HPA)	CPU/latência
Backups	PG/Firestore/Neo4j/Redis	Diários + testes de restore

0.4 Versionamento & Backward Compatibility

- **API**: `/v1`, **Deprecation-Policy** ≥ 90 dias; cabeçalho `Sunset`.
- **Schemas DB**: migrações com **chaves expand/contract** (coluna nova → preencher → trocar leitura → remover legado).
- **Score**: `algorithm_version` salvo junto a cada snapshot para replays/explicabilidade.

0.5 Padrões de Segurança

- **JWT** curto (≤ 15min) + refresh; **JTI** para revoke.
- **RBAC** (tabelas `roles`, `permissions`); guardas por endpoint.
- **Idempotency-Key** em POST críticos (check-in, feedback, boost, subscribe).
- **Assinaturas HMAC** em webhooks (Stripe etc.) + **replay protection** (timestamp window 5 min).
- **PII** segregada (schema dedicado); **hash salgado** para pseudônimos.
- **LGPD/GDPR**: consentimentos, export/delete por subject request.
- **Rate-Limit** por IP+user+partner (Redis Leaky-Bucket).
- **Geo-fencing** (check-ins exigem distância ≤ 100m e sinal de GPS plausível).

0.6 Observabilidade (SRE + Produto)

- **Tracing distribuído** (W3C Traceparent).
- **Logs estruturados JSON** com `event_type`, `actor`, `entity`, `correlation_id`.
- **Métricas técnicas**: latência p50/p95/p99, taxa de erro, saturação, retries.
- **Métricas de produto**: #check-ins, #feedbacks, variação média de `frequency_score`, % colapso → transição, adesão a missões.
- **Alertas** (exemplos):
  - **p95 POST /checkins > 1s** (10 min)
  - **erro 5xx > 2%** (5 min)

    - **explosão de feedbacks negativos** (Z-score > 3 em 30 min)
    - **webhook pagamentos falhando** consecutivamente
- **Dashboards**: Painel Técnico + Painel Vibracional.

0.7 Antifraude & Integridade (alto nível)

Sinais coletados

- Reputação do usuário (**VTS – Vibrational Trust Score**).
- Antiguidade da conta e **entropia de feedbacks**.
- **Coesão geográfica** (cluster DBSCAN dos check-ins).
- Similaridade textual/horária (ataques coordenados).
- Telemetria (device-id hash, IP ASN, latência anômala).

Ações

- **Peso dinâmico** por confiabilidade (ex.: novo usuário = peso 0,3).
- **Quarentena** de lotes suspeitos (não contam no score até revisão).
- **Shadow-ban** para padrões maliciosos reincidentes.
- **Explanação** no painel: “parte dos feedbacks está sob revisão por detecção de anomalia”.

0.8 Transparência do Score (para parceiros)

Composição recomendada (ajustável por versão):

- **Feedbacks & Check-ins**: 70% (ponderados por VTS, geofencing, entropia)
- **Eventos recentes**: 20% (impacto pós-evento com meia-vida)
- **Qualidade de mídia/descrição**: 10% (semântico/estético)

Exibição no painel

- **Score atual** (0–100) + **estado** (Colapso/Transição/Expansão/Pico).
- **Breakdown** com percentuais e **explicações human-readable**.
- Histórico (últimos 30/90 dias) e **marcos** (eventos, picos, missões).
- **Changelog vibracional** (o que alterou o score e por quê).

0.9 Pipelines de Cálculo (assíncrono)

Entrada: eventos (check-in, feedback, evento criado, benefício, denúncia) → **filas**

Processo: normalização, antifraude, ponderação, agregação por janela (tumbling 15 min), recomputar score/estado

Saída: snapshot em **PostgreSQL** (histórico) + **Redis** (leitura rápida) + **Firestore** (painel/móvel)

Pseudocódigo (resumo)

```
def recompute_location(location_id, window=15min):
    ev = read_events(location_id, window)
    clean = anti_fraud(ev)            # remove/quarentena
    w = weight_by_vts(clean)          # pesos por reputação
    f = aggregate_feedback(w)         # média ponderada
    e = event_impact(location_id)     # decaimento exponencial
    m = media_quality_signal(location_id)
    score = 0.7*f + 0.2*e + 0.1*m
    state = classify(score)           # limiares calibrados
    persist_snapshot(location_id, score, state, algorithm_version)
```

```
    cache_redis(location_id, score, state, ttl=15min)
```

0.10 Contratos Obrigatórios (padrões de API)

- **Idempotency-Key** (header) em POST críticos.
- **X-Client-Version** para controle de compatibilidade.
- Erros padronizados:

```
{ "code": "VALIDATION_ERROR", "message": "...", "fields": { "lat": "required" } }
```

- **Tracing headers** propagados.
- **Paginação** cursor-based (`next_cursor`).

0.11 Tabelas de Auditoria (exemplos)

`audit_events`

| id | occurred_at | actor_type | actor_id | entity | entity_id | action | payload_json | correlation_id |

`score_snapshots`

| id | location_id | score | state | algorithm_version | computed_at | window_minutes |

`anti_fraud_flags`

| id | location_id | type | severity | count | first_seen | last_seen | status |

0.12 Governança de Algoritmo

- **Versions**: `algorithm_version` imutável por snapshot.
- **Change Logs** públicos para parceiros (sumário).
- **A/B** controlado por feature flags.
- **Rollback** automatizado se p95 latência ↑ ou satisfação ↓.
- **Explainability** (regras/fatores dominantes) disponível via painel.

0.13 Missões de Harmonização (quando colapso/transição)

- **Gatilho**: score < limiar por X dias.
- **Tarefas recomendadas** (exemplos):
  - Adicionar 2 fotos novas (estéticas validadas)
  - Responder 3 feedbacks com cordialidade (NLP positivo)
  - Criar evento “reconexão” em 7 dias
- **Recompensa**: pontos de gamificação + 1 crédito de boost (não cumulativo) ao sair de colapso → transição/expansão.

0.14 Definições de Pronto (DoR) e de Concluído (DoD)

DoR por épico/módulo

- OpenAPI aprovado + schemas DB migrados + cenários de antifraude definidos + observabilidade especificada.

DoD por feature

- Testes **unit (≥ 85%)**, **contract**, **load** (p95), **security** (OWASP), **e2e** críticos.

- Dashboards e alertas criados.
- Playbooks de incidentes publicados.
- Documentação no manual (esta base) atualizada.

0.15 Testes (matriz mínima)

Tipo	Escopo
Unit	cálculo de score, classificadores, antifraude
Contract	conformidade OpenAPI + erros
Property-based	invariantes (ex.: score ∈ [0,100])
Load	1000 check-ins/min (auto-scale)
Chaos	perda temporária de Firestore/Redis
Security	JWT replay, webhook signature, rate-limit
UX	estados vazios, geofencing off, offline

0.16 Dados Sintéticos & Ambientes

- **Seed** com 500 locais, 50k check-ins, 120k feedbacks distribuídos.
- **Ambientes**: `dev` (hot-reload), `staging` (paridade), `prod` (locks).
- **Toggles** via feature-flags (Aurah insights v2, antifraude agressivo, scores v3).

0.17 Política de Erros & Degradação

- Falha de recomputação → exibir **último snapshot válido** + banner "dados atualizados a cada ~15 min".
- Falha de Firestore → **fila em memória** + retry/backoff; UI mostra status **pendente**.
- Pagamento indisponível → bloquear **upgrade** (gracioso), manter direitos até retentativas.

0.18 Ética & Conformidade

- **Sem profiling sensível**; clusters são anônimos.
- **K-anonymity** mínimo (exibir agregados ≥ 10 contribuições).
- **Right to Explanation** para parceiros (por que o score caiu/subiu).
- Canal de recurso/contestação com SLA.

0.19 Tabelas "SE → ENTÃO" (exemplos críticos)

Estados e Ações

SE...	ENTÃO...	Módulo
`score` < L1 por 7 dias	Iniciar **Harmonização**	IA/Advisor
Padrão anômalo detectado	Quarentena + weight↓	Antifraude
Plano Visionário Anual ativo	`bonus_event_boosts = 1/mês`	Assinaturas
Denúncia confirmada	Ocultar do feed + revisar	Moderação

Erros API

Condição	Resposta
Geofencing inválido	`403 GEO_CONSTRAINT_FAILED`
Idempotency duplicada	`409 DUPLICATE_REQUEST`
Tier insuficiente	`403 TIER_NOT_ALLOWED`

0.20 Checklist de Implantação (Go/No-Go)

- ☐ Migrações rodadas/validadas
- ☐ Dashboards (técnico e vibracional) on-line
- ☐ Alertas críticos ativos
- ☐ Webhooks de pagamento validados (assinatura + retry)
- ☐ Testes e2e passed
- ☐ Playbooks incidentes publicados
- ☐ Feature-flags predefinidas

0.21 Roadmap Técnico (alto nível)

- v1.0: score v1 + antifraude v1 + tiers + painel básico/avançado
- v1.1: explicabilidade ampliada + A/B de pesos regionais
- v1.2: media-understanding v2 (estética/cores/luz)
- v1.3: boost inteligente (budget-aware) + learning-to-rank em recomendações

---

0.22 Glossário Rápido

- **VTS**: Vibrational Trust Score (confiabilidade do usuário)
- **Snapshot**: estado consolidado do score/estado exibido na UI
- **Missões**: tarefas gamificadas de recuperação/expansão
- **Advisor**: camada da IA que sugere ações data-driven

🔚 Fechamento da Camada 0

Esta **Filosofia de Execução** é o contrato de como construir **sem lacunas**. A partir daqui, cada camada vai referenciar estes padrões (ordem, NFRs, antifraude, observabilidade, transparência e ética). O resultado é um sistema **justo, escalável e explicável**, pronto para ser base do plano de negócio e da expansão global do FriendApp.

📘 CAMADA 1 — PROPÓSITO E VISÃO GERAL DO SISTEMA DE LOCAIS PARCEIROS (FRIENDAPP)

1.1 Objetivo da Camada

Definir de forma **estratégica e técnica** o papel do módulo de Locais Parceiros dentro do ecossistema FriendApp. Explicar **o que ele é, por que existe, quais dores resolve, quais são seus objetivos estratégicos**, além de deixar claro **o escopo funcional incluído e excluído**. Esta camada funciona como o "contrato de propósito" do sistema.

1.2 Definição Precisa

O **Sistema de Locais Parceiros e Experiências Comerciais** é o **núcleo B2B** do FriendApp, conectando espaços físicos (restaurantes, cafés, coworkings, retiros, bares, hotéis, centros culturais, baladas, templos, estúdios, etc.) à **malha vibracional e social digital**.

Cada local torna-se um **Hotspot Vibracional Certificado** que pode:

- Aparecer no **Mapa de Frequência** em tempo real.
- Interagir com usuários via **Feed Sensorial**.

- Ser palco para **Eventos e Encontros**.
- Participar do **Sistema de Matching** da IA Aurah Kosmos.
- Receber **Check-ins e Feedbacks Energéticos** que alimentam seu score vibracional.
- Usar um **Painel B2B** para gerenciar presença, eventos, benefícios e métricas.

1.3 Propósito Estratégico

1. **Conectar o físico ao digital vibracional**

   Tornar o mundo físico parte da rede consciente, com locais que respiram dados energéticos em tempo real.

2. **Criar a Economia da Consciência**

   Estabelecer um mercado onde a visibilidade e reputação de um local são definidas pelo **estado vibracional**, não por anúncios invasivos ou preços.

3. **Segurança energética e emocional**

   Garantir que apenas locais alinhados, autênticos e verificados entrem no ecossistema, protegendo usuários de experiências tóxicas.

4. **Monetização progressiva e consciente**

   Através dos tiers (Essencial, Premium, Visionário), criar um modelo freemium robusto que sustente o ecossistema sem comprometer sua essência.

5. **Construir um mapa vibracional planetário**

   Cada local parceiro alimenta a malha global, permitindo que o FriendApp seja um reflexo vivo da vibração coletiva da humanidade.

1.4 Escopo Funcional

✅ Incluído

- Cadastro guiado de locais (com leitura vibracional inicial pela IA).
- Verificação documental (DUC) + validação vibracional.
- Integração completa com **Mapa, Feed, Matching e Eventos**.
- Check-ins energéticos e feedbacks anônimos dos usuários.
- Painel B2B para gestão de perfil, métricas, eventos e benefícios.
- Tiers progressivos (freemium + assinaturas mensais/anuais).
- Aurah Advisor for Partners (exclusivo para Visionário).
- Sistema de impulsionamento (com bônus e FriendCoins).
- Webhooks de pagamento, billing e controle antifraude.
- Moderação automática (colapsos, incoerências, ataques coordenados).
- Missões de Harmonização gamificadas para locais em “Colapso”.

❌ Não Incluído

- Gestão de estoque, reservas ou delivery (isso é feito por integrações externas).
- Marketplace para produtos físicos.
- CRM ou ERP genéricos que não se relacionem à malha vibracional.

1.5 Entregas Estratégicas do Módulo

- **Para usuários:** confiança, segurança, novas experiências vibracionais reais.

- **Para parceiros:** ferramentas de visibilidade, métricas, insights da IA, economia vibracional justa.
- **Para investidores:** motor B2B de monetização sustentável, baseado em dados inéditos (frequência e intenção humana).

1.6 Integrações Críticas

- **IA Aurah Kosmos** → leitura vibracional, matching, antifraude.
- **Mapa de Frequência** → exibição de estados vibracionais em tempo real.
- **Sistema de Eventos** → locais como palcos da experiência.
- **Feed Sensorial** → visibilidade dinâmica.
- **Modo Bora** → sugestão de pontos de encontro.
- **Sistema de Pagamentos** → tiers e impulsionamentos.

1.7 Governança

- O módulo é auditável: logs estruturados + snapshots vibracionais históricos.
- Score vibracional possui **versões de algoritmo**, garantindo explicabilidade e possibilidade de rollback.
- Parceiros têm direito de contestação e explicação (“por que meu score caiu?”).

⬅️ END Fechamento da Camada

O Sistema de Locais Parceiros é o **motor B2B e comercial** do FriendApp, mas sua base não é propaganda — é **energia**. Ele conecta a vibração do mundo físico à malha digital consciente, transformando cada café, cada bar, cada espaço em **portais de conexão autêntica**.

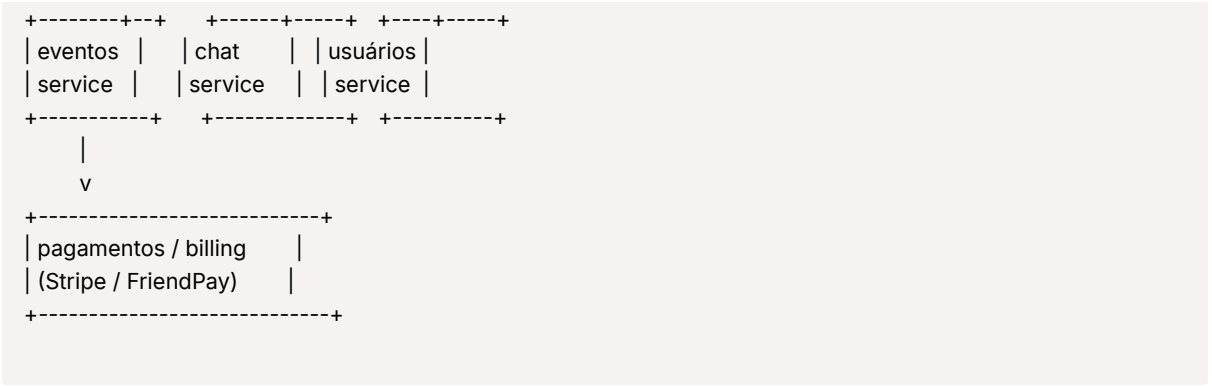
📘 CAMADA 2 — ARQUITETURA GERAL DO MÓDULO (SISTEMA DE LOCAIS PARCEIROS)

2.1 Objetivo da Camada

Definir **a arquitetura técnica e operacional** do módulo de Locais Parceiros, detalhando **como os microsserviços, bancos, filas e caches se comunicam**, quais tecnologias são usadas e como garantir **escalabilidade, observabilidade e resiliência**. Esta camada é a fundação para os desenvolvedores arquitetarem o sistema de forma segura e eficiente.

2.2 Arquitetura em Microsserviços

Diagrama Lógico



Funções

- **locais-parceiros-service**: núcleo (cadastro, verificação, tiers, cálculo de score vibracional, impulsionamentos).
- **eventos-service**: permite que locais criem e gerenciem eventos.
- **chat-service**: ativa chats vibracionais ligados a locais e eventos.
- **users-service**: sincroniza perfis, check-ins e preferências.
- **pagamentos-service**: controla assinaturas, planos, webhooks e FriendCoins.

2.3 Tecnologias-Chave

Backend

Tecnologia	Função
Node.js	APIs REST + orquestração de microsserviços
Python	Engine da IA para leitura vibracional (NLP, Sentimento)
Go	Rotinas críticas de baixa latência (check-ins massivos)
WebSockets	Comunicação em tempo real (chats, eventos)
Kubernetes	Orquestração, escalabilidade automática, isolamento de cargas

Frontend

Tecnologia	Função
Flutter	Aplicativo mobile (iOS/Android)
React Admin	Painel B2B Web para parceiros
FriendFX	Motor sensorial (animações de aura, feed dinâmico)

Bancos de Dados

Banco	Função	Justificativa
PostgreSQL	Estruturado: cadastro, contratos, billing, histórico	Consistência e ACID
Firestore	NoSQL em tempo real: check-ins, feedbacks, feed	Latência mínima
Neo4j	Grafo: conexões entre locais, usuários, eventos	Suporte a consultas de malha vibracional
Redis	Cache: scores, proximidade, rate-limits	Baixa latência, failover seguro

Observabilidade & Infra

Tecnologia	Função
AWS + GCP	Multi-cloud redundancy
Cloudflare CDN	Assets, imagens, mapas vibracionais
Load Balancer Global	Distribuição planetária de tráfego

Tecnologia	Função
Grafana + Prometheus	Métricas técnicas e vibracionais
Datadog	Logs, alertas, tracing distribuído

2.4 Fluxos Técnicos Principais

1. **Cadastro de Local** → API grava em PostgreSQL + trigger inicial de leitura da IA.
2. **Check-in / Feedback** → escrito em Firestore, processado em filas, recalculado assíncronamente e salvo em snapshot (PG + Redis).
3. **Cálculo de Score** → job periódico (15min) agrega, detecta fraudes e atualiza snapshots.
4. **Aurah Kosmos Advisor** → lê snapshots + padrões de rede em Neo4j para sugerir ações.
5. **Eventos e Impulsionamentos** → locais criam eventos (PG), podem impulsionar (boletado via Stripe/FriendPay).
6. **Visualização no App** → UI mobile consome snapshots via Firestore/Redis para garantir baixa latência.

2.5 Resiliência e Degradação

- Falha em Firestore → fallback para snapshots em Redis.
- Falha em pagamentos → parceiro mantém direitos até retry/backoff.
- Falha na IA Aurah Kosmos → sistema exibe último estado válido + aviso “dados em atualização”.

2.6 Segurança e Compliance

- **JWT** com escopos granulares (`user`, `partner`, `admin`).
- **Idempotency-Key** em operações críticas (check-ins, boosts, billing).
- **Geo-fencing** para check-ins (≤100m + GPS plausível).
- **Hash SHA-256** em logs para integridade.
- **LGPD/GDPR**: pseudonimização, direito à exclusão, K-anonymity nos insights.

🔚 Fechamento da Camada 2

A arquitetura garante que o **Sistema de Locais Parceiros** seja **resiliente, escalável e auditável**. É um módulo distribuído, apoiado em multi-cloud, com bancos especializados e camadas de cache/observabilidade. Cada interação vibracional se transforma em dado confiável, pronto para alimentar tanto a IA Aurah quanto os painéis de usuários e parceiros.

📘 CAMADA 3.1 — JORNADA DO PARCEIRO (CADASTRO + VERIFICAÇÃO)

> Objetivo: especificar o fluxo B2B inicial para entrada de um local parceiro no ecossistema: criação de conta, cadastro do estabelecimento, verificação documental e leitura vibracional inicial. Inclui modelos de dados, contratos de API, regras de negócio, antifraude, auditoria e observabilidade.

3.1.1 Mapa do Fluxo (alto nível)

```
[Landing Parceiros]
  → [Criar Conta (owner)]
  → [Login + Aceite ToS/DP]
  → [Criar Local]
```

→ [Upload Docs + Verificação]
→ [Leitura Vibracional Inicial IA]
→ [Aprovação Automática ou Revisão Manual]
→ [Ativar Perfil no Mapa/Feed]

Estados do **local** (`partner_locations.status`):

`DRAFT` → `PENDING_VERIFICATION` → `UNDER_REVIEW` → (`ACTIVE` | `REJECTED` | `BLOCKED`)

3.1.2 Modelagem de Dados (PostgreSQL)

Tabela: `partner_owners`

Coluna	Tipo	Regra
id	UUID PK	gerado no signup
email	CITEXT UNIQUE NOT NULL	validação MX + regex
phone_e164	VARCHAR(20)	opcional
password_hash	TEXT NOT NULL	bcrypt(12+)
is_email_verified	BOOLEAN DEFAULT FALSE	verificação por token
created_at / updated_at	TIMESTAMPTZ	default now()

Tabela: `partner_locations`

Coluna	Tipo	Regra
id	UUID PK	
owner_id	UUID FK→partner_owners.id	NOT NULL
name	VARCHAR(120)	NOT NULL
category	VARCHAR(60)	NOT NULL (enum lógico)
description	TEXT	
address_full	TEXT	NOT NULL
geo	GEOGRAPHY(Point,4326)	NOT NULL (lat/long)
images	TEXT[]	1..10 URLs
intention	VARCHAR(40)	enum lógico (cura, social, etc.)
tier	ENUM('essencial','premium','visionario')	default 'essencial'
status	ENUM('DRAFT','PENDING_VERIFICATION','UNDER_REVIEW','ACTIVE','REJECTED','BLOCKED')	default 'DRAFT'
frequency_score	NUMERIC(5,2)	default 0.00
vibrational_state	ENUM('COLAPSO','TRANSICAO','EXPANSAO','PICO')	default 'TRANSICAO'
algorithm_version	VARCHAR(12)	ex: 'v1.0'
created_at / updated_at	TIMESTAMPTZ	

Índices:

- `idx_partner_locations_geo` (GIST) para proximidade.
- `idx_partner_locations_status` (BTREE).
- `idx_partner_locations_owner` (BTREE).

Tabela: `partner_verifications`

Coluna	Tipo	Regra
id	UUID PK	
location_id	UUID FK→partner_locations.id	UNIQUE NOT NULL
cnpj	VARCHAR(18)	mascarado em exibição
doc_owner_type	ENUM('RG','CNH','PASSAPORTE')	
doc_owner_number	VARCHAR(32)	criptografado at-rest
proof_address_url	TEXT	opcional
ocr_confidence	NUMERIC(4,3)	0..1
status	ENUM('PENDING','AUTO_APPROVED','REVIEW','REJECTED')	default 'PENDING'
reason	TEXT	explicação em caso de REJECTED
created_at / updated_at	TIMESTAMPTZ	

Tabela: `audit_events`

Coluna	Tipo	Regra
id	UUID PK	
occurred_at	TIMESTAMPTZ	now()
actor_type	ENUM('OWNER','SYSTEM','ADMIN')	
actor_id	UUID	
entity	TEXT	ex: 'partner_location'
entity_id	UUID	
action	TEXT	ex: 'CREATE','SUBMIT_VERIFICATION'
payload_json	JSONB	diff/minuta
correlation_id	UUID	encadeamento

3.1.3 Regras de Negócio (SE → ENTÃO)

SE...	ENTÃO...	Módulo
`owner` não verificou e-mail	bloquear criação de local	Auth
local sem 1+ imagem válida	impedir submissão para verificação	UI/API
OCR < 0.85 OU inconsistência CNPJ	`status=REVIEW`, notificar time compliance	Verificação
verificação `AUTO_APPROVED`	`partner_locations.status=ACTIVE`	Workflow
`ACTIVE` sem geo válido	impedir exibição no mapa	Mapa
muitas tentativas de upload doc	rate-limit + flag antifraude	Segurança

3.1.4 Validações de Formulário (UI + API)

- **Nome**: 2–120 chars, sem caps lock contínuo.
- **Categoria**: do catálogo (backed por tabela `categories`).
- **Endereço/Geo**: resolver via geocoder; exigir confirmação no mapa.
- **Imagens**: formatos `jpeg/png/webp`, máx 4MB, min resolução 800px.
- **Descrição**: até 1200 chars (suporte markdown básico).
- **Intenção**: seleção única; influencia UX e IA.

3.1.5 Contratos de API (OpenAPI-ready)

POST `/api/partners/auth/signup`

Body

```
{ "email": "owner@local.com", "password": "Str0ng!Pass", "phone_e164": "+5511999999999" }
```

201

```
{ "owner_id": "uuid", "requires_email_verification": true }
```

Erros: `409 EMAIL_IN_USE`, `422 WEAK_PASSWORD`

POST `/api/partners/auth/login`

Body

```
{ "email": "owner@local.com", "password": "Str0ng!Pass" }
```

200

```
{ "access_token": "jwt", "refresh_token": "jwt", "owner_id": "uuid" }
```

Erros: `401 INVALID_CREDENTIALS`, `423 ACCOUNT_LOCKED`

POST `/api/partners/locations` *(Idempotency-Key)*

Body

```
{
  "name":"Café Alma Zen",
  "category":"cafe",
  "description":"...",
  "address_full":"Rua X, 123 - SP",
  "geo": { "lat": -23.561, "lng": -46.655 },
  "images":[ "https://cdn/.../1.jpg" ],
  "intention":"cura"
}
```

201

```
{ "location_id":"uuid", "status":"DRAFT" }
```

Erros: `422 VALIDATION_ERROR`, `429 RATE_LIMITED`

POST `/api/partners/locations/{id}/submit-verification` *(Idempotency-Key)*

Body

```
{
  "cnpj":"12.345.678/0001-90",
  "doc_owner_type":"CNH",
  "doc_owner_number":"99999999999",
  "proof_address_url":"https://cdn/.../comprovante.pdf"
```

```
}
```

202

```
{ "verification_id":"uuid", "status":"PENDING" }
```

Erros: `400 MISSING_DOCS`, `409 ALREADY_SUBMITTED`, `403 NOT_OWNER`

GET `/api/partners/locations/{id}/status`

200

```
{
  "status":"PENDING_VERIFICATION",
  "verification_status":"PENDING",
  "vibrational_state_preview":"TRANSICAO",
  "next_steps":[ "aguardar_analise", "completar_perfil" ]
}
```

Webhook `/webhooks/verification` *(SYSTEM→BACKEND)*

Body (exemplo AUTO-APPROVED)

```
{
  "event":"VERIFICATION_RESULT",
  "location_id":"uuid",
  "result":"AUTO_APPROVED",
  "ocr_confidence":0.92,
  "signals": { "cnpj_ok": true, "doc_match": true }
}
```

Resposta 200 → Back-end promove `partner_locations.status=ACTIVE`.

3.1.6 Antifraude (fase de cadastro/verificação)

Sinais:

- Repetição de CNPJ/documento em múltiplos owners.
- Uploads em bursts (entropia baixa de EXIF/metadata).
- IPs de datacenter/ASN suspeitos.
- Mismatch entre endereço e geo (distância > 2km).

Ações:

- **Quarentena** da verificação (`status=REVIEW`).
- **Shadow-review** por compliance.
- Registro em `anti_fraud_flags` com severidade.

3.1.7 Leitura Vibracional Inicial (IA)

Inputs

- Imagens (cores, luz, composição – CNN/CLIP-like).

- Descrição (NLP semântico, intenção).
- Categoria/Intenção.
- Contexto urbano (ruído, densidade – dados públicos se disponíveis).

Saídas

- `frequency_score_initial` (0–100).
- `vibrational_state_initial` (`COLAPSO` / `TRANSICAO` / `EXPANSAO` / `PICO`).
- `explainability`: top-3 fatores (ex.: “paleta quente + descrição acolhedora + categoria cura”).

Persistência

- Gravar no `partner_locations` + lançar **snapshot** (histórico inicial).
- `algorithm_version` registrado.

3.1.8 Pseudocódigo (workflow de submissão)

```
def submit_verification(owner_id, location_id, payload, idem_key):
    assert_owner(location_id, owner_id)
    require_status(location_id, allowed=['DRAFT','REJECTED'])
    validate_docs(payload)
    with idempotency(idem_key):
        vid = create_partner_verification(location_id, payload)
        set_location_status(location_id, 'PENDING_VERIFICATION')
        emit_event('verification_submitted', location_id, vid)
    return vid
```

Webhook handler:

```
def on_verification_result(evt):
    loc = evt['location_id']
    if evt['result'] == 'AUTO_APPROVED':
        set_verification_status(loc, 'AUTO_APPROVED')
        set_location_status(loc, 'ACTIVE')
        run_initial_vibration_read(loc)  # IA
        emit_event('location_activated', loc)
    elif evt['result'] == 'REVIEW':
        set_verification_status(loc, 'REVIEW')
        set_location_status(loc, 'UNDER_REVIEW')
    else:
        set_verification_status(loc, 'REJECTED', reason=evt.get('reason'))
        set_location_status(loc, 'REJECTED')
```

3.1.9 Email/Push Templates (transacionais)

- **Confirmação de Conta**: link + expiração 24h.
- **Recebemos seus documentos**: protocolo, prazo estimado.
- **Aprovado automaticamente**: “Seu local agora está **ATIVO** no mapa ✨”.
- **Revisão manual**: pedido de doc complementar.
- **Rejeição**: motivo + como recorrer.

3.1.10 Observabilidade & Auditoria

Logs estruturados

- `event_type`: `signup`, `create_location`, `submit_verification`, `verification_result`.
- `actor`: owner_id / system.
- `correlation_id`: por requisição.
- `geo_consistency_score`: 0..1.

Métricas (Prometheus):

- `verification_auto_approve_ratio`
- `verification_avg_time_seconds`
- `verification_review_rate`
- `fraud_quarantine_rate`

Alertas:

- pico de `REVIEW` > X% em 1h.
- latência p95 POST `/locations` > 800ms.
- falhas em OCR > Y% (quebra do provedor).

3.1.11 UX & Estados NA UI (parceiro)

- **DRAFT**: checklist com "% de prontidão" (nome, fotos, geo, docs).
- **PENDING_VERIFICATION**: barra de progresso + SLA estimado.
- **UNDER_REVIEW**: "em análise humana" + botão anexar docs extra.
- **ACTIVE**: call-to-action para completar perfil e **criar 1º evento**.
- **REJECTED**: motivo explícito + "corrigir e reenviar".

3.1.12 Políticas e SLAs

- **SLA verificação**: 95% ≤ 5 min (automação); 5% revisão humana ≤ 24h úteis.
- **Retries**: webhooks (exponencial até 24h).
- **Rate-limit**: criação de locais (5/dia/owner), submissão de verificação (3/semana/local).

3.1.13 Critérios de Aceite (DoD desta camada)

- OpenAPI de todas as rotas desta camada aprovado.
- Migrações de `partner_owners`, `partner_locations`, `partner_verifications`, `audit_events` aplicadas.
- **Testes**: unit (≥85%), contract, e2e feliz + bordas (geo inválido, docs faltantes).
- Dashboards e alertas configurados.
- Templates de e-mail/push publicados.
- Playbook de incidentes (falha no OCR/provedor KYC) documentado.

3.1.14 Segurança & Privacidade

- Documentos criptografados em repouso (KMS).
- Acesso a docs restrito a `role:compliance` (RBAC).
- Logs não incluem PII sensível (somente referências).
- Eliminação/anonymização de docs conforme LGPD/GDPR após prazos legais.

🔚 Fechamento da Camada 3.1

Com esta camada, garantimos uma **porta de entrada confiável, auditável e justa** para os parceiros. A verificação é rápida para quem está alinhado e rigorosa contra fraude, enquanto a **IA** já define uma **leitura vibracional inicial** explicável. A partir daqui, seguimos para a **3.2 (Primeiro Acesso ao Painel)** e **3.3 (Gerenciamento + Eventos)** no mesmo nível de profundidade.

📘 CAMADA 3.2 — PRIMEIRO ACESSO AO PAINEL DO PARCEIRO

> Objetivo: especificar a experiência técnica e operacional do parceiro no primeiro login após a aprovação de verificação. Inclui fluxo de autenticação, telas iniciais, missões gamificadas, permissões por tier, auditoria, APIs de sessão e observabilidade.

3.2.1 Fluxo de Primeiro Acesso

```
[Login Owner]
  → [Validação JWT / Refresh]
  → [Verificação Tier e Status do Local]
  → [Exibir Tela de Boas-Vindas + Tour Sensorial]
  → [Liberar Painel com Módulos conforme Tier]
  → [Ativar Missões Iniciais]
```

3.2.2 Autenticação & Sessão

- **Fluxo:** e-mail + senha → JWT curto (15min) + Refresh (7 dias).
- **Headers:** `Authorization: Bearer <jwt>` , `X-Client-Version` .
- **Rotas-chave:**
  - `POST /api/partners/auth/login`
  - `POST /api/partners/auth/refresh`
  - `POST /api/partners/auth/logout`
- **Regras:**
  - `owner` precisa ter status `email_verified=true` .
  - `partner_location.status` deve ser `ACTIVE` .

3.2.3 Tela de Boas-Vindas (UI/UX)

Elementos obrigatórios:

- Mensagem energética da IA:

  "Bem-vindo ao campo vibracional do FriendApp. Sua presença pulsa agora no mapa global."
- Card com estado inicial do local:
  - `Tier` (Essencial / Premium / Visionário).
  - `Energia Atual` (Expansão, Pico, etc.).
  - `Score Vibracional Atual` .
- Mini-mapa com a aura do local em destaque.

3.2.4 Permissões por Tier

Módulo	Essencial	Premium	Visionário
Perfil do Local	✓	✓	✓
Visualização no Mapa	✓	✓	✓
Check-ins & Feedbacks	✓	✓	✓
Eventos (criar/gerir)	❌	✓	✓
Benefícios/Promoções	❌	✓	✓
Dashboard Avançado	❌	❌	✓
Aurah Advisor	❌	❌	✓
API Integrations	❌	❌	✓

3.2.5 Missões Iniciais (Gamificação)

A cada ação inicial concluída, o parceiro recebe **pontos e badges**:

Missão	Ação	Recompensa
Perfil Consciente	Completar 100% do perfil	Badge + 10 pontos
Galeria Viva	Adicionar 3+ fotos	Aura ativada no mapa
Primeira Resposta	Responder 1 feedback	5 pontos
Primeiro Evento	Criar 1 evento (Premium/Vis.)	Badge “Criador de Frequências”

3.2.6 APIs Relacionadas

GET `/api/partners/panel/overview`

200

```
{
  "tier":"essencial",
  "status":"active",
  "vibrational_state":"EXPANSAO",
  "frequency_score":72.5,
  "missions_pending":[ "completar_perfil", "adicionar_fotos" ]
}
```

POST `/api/partners/missions/complete`

Body

```
{ "mission":"completar_perfil" }
```

200

```
{ "status":"ok", "points_awarded":10, "new_badge":"Perfil Consciente" }
```

3.2.7 Auditoria & Logs

- `event_type`: `login_success`, `login_failed`, `panel_loaded`, `mission_completed`.
- `actor`: owner_id / partner_id.

- `payload_json`: detalhes (tier, missão, etc.).
- Correlation IDs para cada sessão.

3.2.8 Observabilidade

Métricas:

- `login_success_count`
- `login_failed_count`
- `panel_load_latency_p95`
- `missions_completed_total`

Alertas:

- falhas de login > 10% em 15min
- p95 painel > 1s

3.2.9 UX & Estados Vazios

- **Sem fotos**: placeholder + CTA “Adicione sua primeira imagem”.
- **Sem eventos**: card “Crie seu primeiro evento vibracional”.
- **Sem feedbacks ainda**: mensagem “Seu campo aguarda as primeiras ressonâncias ✨”.

🔚 Fechamento da Camada 3.2

O primeiro acesso deve transformar o parceiro em **protagonista vibracional**. É onboarding, tutorial, gamificação e ativação em um só fluxo. Tecnologicamente, garante autenticação segura, logs completos e transparência de permissões por tier. Vibracionalmente, dá o tom de que cada ação impacta o mapa coletivo do FriendApp.

📘 CAMADA 3.3 — GERENCIAMENTO DO PERFIL + EVENTOS DO LOCAL PARCEIRO

> Objetivo: detalhar como o parceiro administra seu perfil B2B, interage com feedbacks, edita informações, cria eventos e gerencia benefícios. Esta camada é o coração operacional do painel, onde cada ação impacta diretamente no score vibracional, visibilidade no mapa e engajamento de usuários.

3.3.1 Estrutura do Fluxo

```
[Login Painel Parceiro]
  → [Acessar Módulo Perfil]
  → [Editar Campos / Upload Imagens]
  → [Salvar → Trigger IA Aurah Kosmos → Nova Leitura]
  → [Gerenciar Benefícios]
  → [Criar Evento]
  → [Monitorar Métricas em Tempo Real]
```

3.3.2 Modelagem de Dados (PostgreSQL)

Tabela: `partner_locations` (campos relevantes)

- **name**, **description**, **category**, **intention**: editáveis via painel.

- **images**: array de URLs, 1..10 imagens.
- **updated_at**: atualizado a cada alteração.

Tabela: `partner_events`

Coluna	Tipo	Regras
id	UUID PK	
partner_location_id	FK	NOT NULL
title	VARCHAR(255)	obrigatória
description	TEXT	
start_time / end_time	TIMESTAMPTZ	obrigatórios
frequency_theme	VARCHAR(80)	opcional (cura, diversão, etc.)
is_highlighted	BOOLEAN	default false
created_at / updated_at	TIMESTAMPTZ	

Tabela: `partner_benefits`

Coluna	Tipo	Regra
id	UUID PK	
partner_location_id	FK	
type	ENUM('desconto','brinde','combo')	
description	TEXT	
valid_until	TIMESTAMPTZ	opcional
active	BOOLEAN	default true

3.3.3 Regras de Negócio (SE → ENTÃO)

SE...	ENTÃO...	Módulo
Local altera **nome/categoria/intenção**	Trigger leitura vibracional IA	Aurah Kosmos
Upload de novas imagens	Análise de cor/luz → influencia score	IA Visão
Evento criado	Gera card no Feed Sensorial + mapa	Feed/Eventos
Evento marcado como `is_highlighted`	Precisa de boost (pago/bônus)	Billing
Local em Colapso cria evento	Evento só exibido após harmonização ativa	Moderação

3.3.4 Contratos de API (exemplos)

PUT `/api/partners/locations/{id}`

Body

```
{
  "name": "Espaço Harmonia",
  "description": "Ambiente de cura vibracional",
  "intention": "cura",
  "images": ["https://cdn/img1.jpg","https://cdn/img2.jpg"]
}
```

200

```
{ "status": "updated", "triggered_vibrational_scan": true }
```

POST `/api/partners/events`

Body

```
{
  "partner_location_id": "uuid-local",
  "title": "Círculo de Som",
  "description": "Experiência sonora para elevar a vibração",
  "start_time": "2025-09-20T19:00:00Z",
  "end_time": "2025-09-20T22:00:00Z",
  "frequency_theme": "som"
}
```

201

```
{ "event_id": "uuid-evento", "status": "created" }
```

3.3.5 Benefícios e Promoções

Tipos suportados:

- **Desconto Direto** (% em serviços/produtos).
- **Brinde por Check-in** (usuário recebe ao registrar presença).
- **Combo Exclusivo** (pacote vibracional, ex.: refeição + drink temático).

API:

`POST /api/partners/benefits` → cria benefício.

`GET /api/partners/benefits` → lista.

`PATCH /api/partners/benefits/{id}` → ativa/desativa.

3.3.6 Gestão de Feedbacks

- Parceiro visualiza feedbacks anônimos.
- Pode **responder** (resposta visível no app).
- Feedbacks impactam score vibracional.
- IA detecta anomalias → peso reduzido em fraudes.

3.3.7 Observabilidade

Logs:

- `event_type`: `profile_update`, `image_upload`, `event_created`, `benefit_created`.
- `actor`: owner_id.
- `correlation_id`: encadeamento.

Métricas (Prometheus):

- `partner_profile_update_count`
- `events_created_total`
- `benefits_active_total`
- `avg_feedback_score`

Alertas:

- falha API PUT profile > 5%
- evento criado sem localização válida

3.3.8 UX & Estados Vazios (Painel)

- Sem fotos → CTA: “Adicione imagens para ativar sua aura visual no mapa”.
- Sem eventos → CTA: “Crie seu primeiro evento vibracional”.
- Sem benefícios → CTA: “Ofereça algo exclusivo para visitantes”.

🔚 Fechamento da Camada 3.3

O gerenciamento de perfil e eventos é o **núcleo da atuação do parceiro**. Cada alteração é lida pela IA e reflete no ecossistema: no mapa, no feed, no score e nas recomendações. É aqui que o local se torna um **ator ativo na malha vibracional do FriendApp**, cocriando experiências com usuários.

📘 CAMADA 4.1 — JORNADA DO USUÁRIO (DESCOBERTA + INTERAÇÃO COM LOCAIS)

> Objetivo: especificar como os usuários do FriendApp descobrem, exploram e interagem com os locais parceiros. Esta camada conecta front-end mobile, IA Aurah Kosmos e back-end de locais, garantindo que a experiência seja vibracional, personalizada e auditável.

4.1.1 Fluxo da Jornada

```
[App Mobile]
  → [Mapa Vibracional | Feed Sensorial | Sugestões no Bora]
  → [Card Local → Perfil Detalhado]
  → [Visualizar Aura + Benefícios + Eventos]
  → [Decidir interagir: Check-in, Favoritar, Entrar em Evento]
```

4.1.2 Pontos de Descoberta

Origem	Descrição Técnica	API / Query
Mapa Vibracional	Exibe locais com `frequency_score` + aura animada	GET `/api/partners/locations/public?lat,lng,radius`
Feed Sensorial	Lista dinâmica curada pela IA (Aurah)	GET `/api/feed/locals`
Sugestões no Bora	Locais próximos compatíveis com a energia do grupo	GET `/api/partners/matching?group_id=...`

4.1.3 Perfil do Local (UI/UX)

Elementos que o usuário vê ao abrir um local:

- **Nome + Categoria**
- **Aura Animada**: cores e intensidade conforme `vibrational_state`.
- **Frequência Atual (score 0–100)** + estado (Colapso/Transição/Expansão/Pico).
- **Intenção Principal** (cura, diversão, networking, etc.).
- **Benefícios Ativos** (se Premium/Visionário).
- **Galeria de Fotos** (processadas pela IA para análise semântica).

- **Eventos Ativos** vinculados ao local.
- **Feedbacks Vibracionais Recentes** (anônimos, curados pela IA).

4.1.4 Regras de Negócio (SE → ENTÃO)

SE...	ENTÃO...	Módulo
Local em `COLAPSO`	Card aparece esmaecido no mapa/feed	Moderação IA
Usuário com perfil "introvertido"	Aurah prioriza locais em `EXPANSAO` suave	Matching
Local Visionário com evento ativo	Card do evento fixado no feed por 24h	Feed
Feedbacks recentes < 2.0	Score reduzido (peso maior)	Cálculo Score
Usuário favoritou local	Local aparece no "Meus Espaços"	App

4.1.5 Contratos de API

GET `/api/partners/locations/public`

Query: `lat,lng,radius=3km`

200 Response:

```
[
 {
  "id": "uuid-local",
  "name": "Café Alma Zen",
  "category": "cafe",
  "frequency_score": 82.1,
  "vibrational_state": "EXPANSAO",
  "intention": "cura",
  "geo": { "lat": -23.55, "lng": -46.63 },
  "is_highlighted": true}
]
```

GET `/api/partners/locations/{id}`

200 Response:

```
{
 "id": "uuid-local",
 "name": "Café Alma Zen",
 "category": "cafe",
 "intention": "cura",
 "frequency_score": 82,
 "vibrational_state": "EXPANSAO",
 "benefits": [
  { "type":"desconto", "description":"10% no menu vibracional" }
 ],
 "events": [ { "id":"uuid-evento", "title":"Roda de Som" } ],
 "feedbacks":[ { "score":5, "keywords":["leve","acolhedor"] } ]
}
```

4.1.6 Observabilidade

Logs estruturados

- `event_type`: `location_viewed`, `card_opened`, `feed_loaded`.
- `actor`: user_id.
- `entity`: partner_location_id.
- `payload`: score atual, estado vibracional.

Métricas

- `feed_locals_impressions_total`
- `location_profile_views_total`
- `avg_time_on_location_profile`
- `location_interaction_rate`

Alertas

- p95 GET `/locations/public` > 400ms
- Erro 5xx > 1%
- Latência feed > 800ms

4.1.7 UX & Estados Vazios

- **Sem Benefícios** → placeholder: *"Este espaço ainda não adicionou benefícios."*
- **Sem Eventos** → *"Nenhum evento ativo, siga o local para ser avisado."*
- **Sem Feedbacks** → *"Seja o primeiro a deixar sua impressão vibracional ✨."*

⬅️ END Fechamento da Camada 4.1

A descoberta e interação do usuário com os locais parceiros é **curada pela IA Aurah Kosmos** e ancorada em dados vibracionais. Cada card, cada aura, cada detalhe do perfil é **dinâmico e auditável**, garantindo que o usuário viva não apenas uma navegação, mas uma **ressonância consciente**.

📘 CAMADA 4.2 — CHECK-IN ENERGÉTICO E FEEDBACK VIBRACIONAL

> Objetivo: especificar a mecânica, lógica técnica e UX do check-in energético feito pelos usuários nos locais parceiros. Essa camada descreve como registrar presença, como coletar feedback vibracional anônimo e como esses dados são processados pela IA Aurah Kosmos para atualizar o estado energético de cada local.

4.2.1 Fluxo do Check-in

```
[App Usuário]
  → [Abrir Perfil do Local]
  → [Tocar em "Fazer Check-in Energético"]
  → [Validação Geo + Autenticação]
  → [Registro em Firestore + Log]
  → [Convite para Deixar Feedback Vibracional]
  → [Dados processados → Recalcular Score]
```

4.2.2 Regras Técnicas do Check-in

- **Geo-fencing**: distância máxima ≤ 100m do ponto cadastrado.

- **1 check-in por local / 24h** por usuário.
- **Usuários não verificados** → check-in permitido, mas com peso reduzido no cálculo.
- **Device-id + IP ASN** usados para detectar fraudes (ex: múltiplos check-ins de mesmo aparelho).
- **Idempotency-Key** obrigatória no POST para evitar duplicações.

4.2.3 Modelagem de Dados

Firestore (tempo real)

Coleção: `checkins`

```
{
  "checkin_id":"uuid",
  "user_id":"uuid",
  "location_id":"uuid",
  "geo": { "lat":-23.55,"lng":-46.63 },
  "timestamp":"2025-09-14T20:00:00Z",
  "valid":true}
```

PostgreSQL (snapshot histórico)

Tabela: `partner_checkins`

Coluna	Tipo	Observações
id	UUID PK	
user_id	UUID FK	pseudonimizado em agregados
partner_location_id	UUID FK	
timestamp	TIMESTAMPTZ	
vibration_feedback	SMALLINT	0–5
comment	TEXT	opcional
geo_validated	BOOLEAN	default true

4.2.4 Feedback Vibracional

Após o check-in, o app abre modal para o usuário deixar feedback:

Inputs:

- Slider 0–5: “Como você sentiu a vibração?”
- Emojis sensoriais (alegria, leveza, caos, etc.)
- Palavras-chave sugeridas pela IA (ex.: *acolhedor*, *introspectivo*)
- Campo opcional de comentário livre

Anonimato: feedbacks não exibem identidade do usuário; apenas agregados.

4.2.5 API de Check-in e Feedback

POST `/api/partners/checkin`

Body

```
{
  "location_id":"uuid-local",
  "geo":{"lat":-23.55,"lng":-46.63}
```

```
}
```

201

```
{ "checkin_id":"uuid", "status":"recorded" }
```

POST `/api/partners/checkin/{id}/feedback`

Body

```
{
  "vibration_feedback":4,
  "emojis":["✨","🌿"],
  "keywords":["leve","acolhedor"],
  "comment":"Ambiente tranquilo e inspirador"
}
```

201

```
{ "status":"saved", "impact_on_score":true }
```

4.2.6 Processamento pela IA Aurah Kosmos

Inputs:

- Score médio dos feedbacks.
- Emojis/keywords → processados por NLP semântico.
- Padrões anômalos → detectados (ataque de feedbacks).
- Peso do usuário via VTS (Vibrational Trust Score).

Pipeline:

1. Normalizar check-ins recentes (últimos 15 min).
2. Aplicar antifraude (quarentena de feedbacks suspeitos).
3. Calcular média ponderada.
4. Atualizar `frequency_score` e `vibrational_state`.
5. Persistir snapshot (PG + Redis).

4.2.7 Observabilidade

Logs estruturados:

- `event_type`: `checkin_created`, `feedback_submitted`.
- `actor`: user_id (hash).
- `entity`: partner_location_id.

Métricas:

- `checkins_per_location_total`
- `feedbacks_per_location_total`
- `avg_vibration_feedback`

- `anomaly_feedback_detected_total`

Alertas:

- Explosão de feedbacks negativos (Z-score > 3).
- Mais de 50 check-ins de um mesmo device_id/dia.
- Latência POST `/checkin` > 800ms.

4.2.8 UX (estados na UI)

- **Check-in confirmado**: animação de aura dourada pulsando.
- **Feedback enviado**: partículas se unindo ao campo vibracional do local.
- **Sem conexão**: mensagem "Check-in salvo, será enviado quando você se reconectar."

🔚 Fechamento da Camada 4.2

O check-in energético é a forma de o usuário **anotar sua presença no campo coletivo**. O feedback vibracional transforma essa presença em dado vivo, que a IA processa para atualizar em tempo real a frequência de cada espaço. Esse ciclo é o coração do FriendApp: **usuário → local → IA → mapa global**.

📘 CAMADA 5 — MODELO DE TIERS (ESTRATÉGIA HÍBRIDA + PLANOS ANUAIS)

> Objetivo: especificar o sistema de monetização B2B via tiers progressivos (Essencial, Premium e Visionário), detalhando funcionalidades por nível, lógica técnica, billing mensal/anual, bônus, antifraude e integração com APIs de pagamento. Esta camada transforma o propósito vibracional em sustentação comercial escalável.

5.1 Estrutura de Tiers

Tier	Tipo	Objetivo Estratégico
Essencial	Gratuito	Popular mapa, gerar massa crítica inicial.
Premium	Pago (mensal/anual)	Monetização inicial + visibilidade extra.
Visionário	Pago (mensal/anual)	Inteligência avançada (IA Advisor), impulsionamentos e integrações.

5.2 Funcionalidades por Tier

Funcionalidade	Essencial	Premium	Visionário
Cadastro básico + Perfil	✓	✓	✓
Aparecer no mapa	✓	✓	✓
Receber check-ins & feedbacks	✓	✓	✓
Perfil completo (galeria, redes sociais, descrição expandida)	❌	✓	✓
Destaque visual (aura especial)	❌	✓	✓
Criar eventos	❌	✓	✓
Benefícios/promos	❌	✓	✓
Dashboard básico	❌	✓	✓
Dashboard avançado + IA Aurah Advisor	❌	❌	✓
Impulsionamento de eventos/perfil	❌	❌	✓
API Integration (integrações externas)	❌	❌	✓
Suporte prioritário	❌	❌	✓

5.3 Ciclos de Pagamento

- **Mensal**: cobrança recorrente via Stripe/FriendPay.
- **Anual**: pagamento único, com desconto (pague 10 → receba 12 meses).

Bônus por plano anual

- Premium: selo “Parceiro Fundador” + prioridade leve no matching.
- Visionário: 1 impulsionamento gratuito por mês + insights exclusivos da Aurah Advisor.

5.4 Modelagem de Dados

Tabela: `partner_plans`

Coluna	Tipo	Regra
id	UUID PK	
partner_id	UUID FK	NOT NULL
tier	ENUM('essencial','premium','visionario')	default 'essencial'
billing_cycle	ENUM('monthly','annual')	
subscription_status	ENUM('active','inactive','pending_payment','cancelled')	
subscription_end	TIMESTAMPTZ	
bonus_event_boosts	INT	default 0
last_bonus_applied_at	TIMESTAMPTZ	

5.5 APIs

POST `/api/partners/subscribe`

Body

```
{
  "tier":"visionario",
  "billing_cycle":"annual",
  "payment_token":"tok_abc123"
}
```

200

```
{
  "status":"active",
  "subscription_end":"2026-09-14T00:00:00Z",
  "bonus_event_boosts":1
}
```

POST `/api/partners/events/boost`

- Verifica se há `bonus_event_boosts > 0`.
- Se sim: aplica boost e decrementa contador.
- Se não: cobra via Stripe/FriendCoins.

5.6 Cron Job — Bônus Mensal Visionário

Executado todo dia 1º do mês:

```
for partner in visionarios_anuais:
    if partner.subscription_status == 'active':
        partner.bonus_event_boosts = 1
        partner.last_bonus_applied_at = now()
```

5.7 Antifraude

- Validação de billing via webhooks assinados (Stripe HMAC).
- Tentativas de “downgrade silencioso” → logadas e auditadas.
- Impulsionamentos só válidos em eventos futuros (>6h).
- Score vibracional não pode ser manipulado via upgrade de tier.

5.8 Observabilidade

Logs

- `event_type`: `subscription_created`, `plan_upgraded`, `bonus_applied`.
- `actor`: partner_id.
- `payload`: tier, ciclo, status.

Métricas

- `active_subscriptions_total`
- `plan_upgrade_rate`
- `bonus_boosts_used_total`

Alertas

- Falha em webhooks > 5%
- Divergência entre Stripe e DB local

5.9 UX no Painel do Parceiro

- Tela “Meu Plano” → exibe tier atual, billing, vencimento, bônus disponíveis.
- Botão de upgrade → abre tela com comparativo de tiers.
- Indicação clara do bônus mensal (ex.: “Você tem 1 boost gratuito este mês”).

🔚 Fechamento da Camada 5

O Modelo de Tiers é o **motor de monetização consciente** do FriendApp.

Ele garante acesso gratuito para criar massa crítica, oferece valor real no Premium e posiciona o Visionário como **guardião vibracional estratégico**, com ferramentas exclusivas e bônus. Do ponto de vista técnico, integra billing, antifraude, APIs seguras e transparência para parceiros.

📘 CAMADA 6 — IA DE LEITURA VIBRACIONAL DE LOCAIS (O ALGORITMO SECRETO)

> Objetivo: especificar o funcionamento do algoritmo da IA Aurah Kosmos que analisa cada local parceiro e define seu frequency_score (0–100) e o estado vibracional (Colapso, Transição, Expansão, Pico). Esta camada garante

> robustez antifraude, transparência para parceiros e performance escalável.

6.1 Fluxo de Processamento

```
[Inputs: fotos, descrições, check-ins, feedbacks, eventos]
  → [Normalização e Antifraude]
  → [Ponderação por Reputação do Usuário]
  → [Análise Semântica e Emocional]
  → [Agregação de Dados]
  → [Cálculo do Score Vibracional]
  → [Classificação do Estado]
  → [Persistência de Snapshot + Cache]
  → [Exibição no App / Painel B2B]
```

6.2 Inputs Utilizados

Fonte	Tipo	Descrição
Feedbacks Energéticos	Numérico + semântico	Slider 0–5 + palavras-chave processadas via NLP
Check-ins	Numérico	Volume, frequência e consistência geográfica
Fotos do Local	Visual	Paleta de cores, luz, enquadramento, estética
Descrição	Textual	Análise semântica: palavras de alta/baixa frequência
Eventos	Contextual	Presença/ausência e avaliações pós-evento
Histórico do Local	Temporal	Curva vibracional das últimas semanas
Perfis de Usuários	Ponderado	Reputação (VTS) dos que interagem

6.3 Regras de Ponderação

Input	Peso Base	Observações
Feedback de usuários	50%	Ajustado por VTS e detecção de anomalias
Check-ins válidos	20%	Clusterizados geograficamente
Eventos e avaliações	15%	Peso maior logo após eventos
Fotos e descrição	10%	Analisadas via CNN/NLP
Histórico	5%	Evita variação abrupta

6.4 Antifraude no Algoritmo

- **Ponderação por VTS**: usuários novos ou com histórico incoerente têm peso reduzido (ex.: 0.3).
- **Clusterização DBSCAN**: check-ins devem ter coerência geográfica.
- **Detecção de ataques coordenados**: explosão de feedbacks negativos semelhantes → feedbacks entram em quarentena.
- **Logs de antifraude** gravados em tabela `anti_fraud_flags`.

6.5 Cálculo do Score Vibracional

Fórmula Base

```
score = (0.5 * feedbacks) + (0.2 * checkins) + (0.15 * eventos) + (0.1 * midia) + (0.05 * historico)
```

Estados

Score	Estado
0–39	Colapso
40–59	Transição
60–79	Expansão
80–100	Pico

6.6 Transparência para Parceiros

No Painel B2B, o parceiro vê:

```
{
  "frequency_score": 82,
  "vibrational_state": "EXPANSAO",
  "composition": {
    "feedbacks": 50,
    "checkins": 20,
    "eventos": 15,
    "midia": 10,
    "historico": 5
  },
  "last_updated":"2025-09-14T18:00:00Z"
}
```

6.7 Performance e Escalabilidade

- **Cálculo assíncrono**: rodado a cada 15min em batch.
- **Cache Redis**: leitura rápida para app e painel.
- **Persistência**: snapshots históricos salvos em PostgreSQL.
- **Firestore**: usado para atualizar UI em tempo real.

6.8 Observabilidade

Métricas

- `avg_frequency_score`
- `state_distribution{state=...}`
- `anomaly_feedback_detected_total`

Alertas

- p95 cálculo > 5s
- 10% feedbacks em quarentena em 1h

6.9 Pseudocódigo Simplificado

```
def compute_vibrational_score(location_id):
    feedbacks = weighted_feedbacks(location_id)
    checkins = valid_checkins(location_id)
    events = event_impact(location_id)
    media = media_analysis(location_id)
    history = historical_factor(location_id)
```

```
score = 0.5*feedbacks + 0.2*checkins + 0.15*events + 0.1*media + 0.05*history
state = classify(score)

persist_snapshot(location_id, score, state)
cache_redis(location_id, score, state)
return score, state
```

🔚 Fechamento da Camada 6

A IA de Leitura Vibracional garante que cada local seja **avaliado de forma justa, antifraude e transparente**. O score é calculado com múltiplos inputs, explicado no painel e auditável via logs. Esse sistema é a **coluna vertebral da economia vibracional** do FriendApp.

📘 CAMADA 7 — PAINEL DO PARCEIRO (FERRAMENTA B2B)

> Objetivo: especificar a ferramenta B2B utilizada pelos locais parceiros para gerenciar perfil, eventos, benefícios e métricas vibracionais. O painel é a principal interface de gestão comercial + vibracional para os parceiros dentro do ecossistema FriendApp.

7.1 Fluxo de Uso

```
[Login no Painel]
  → [Visão Geral do Local (Dashboard)]
  → [Editar Perfil / Benefícios / Eventos]
  → [Acessar Métricas e Feedbacks]
  → [Receber Sugestões da IA Aurah Advisor]
  → [Executar Ações (boost, criar eventos, harmonização)]
```

7.2 Estrutura do Painel

Seções principais

1. **Dashboard Inicial** (status vibracional, KPIs, missões ativas).
2. **Perfil do Local** (editar dados, fotos, intenção).
3. **Benefícios e Promoções** (criar/gerenciar descontos, combos, brindes).
4. **Eventos** (criar, impulsionar e monitorar resultados).
5. **Feedbacks** (visualizar, responder, insights agregados).
6. **Aurah Advisor** (sugestões personalizadas de IA).
7. **Meu Plano** (tier atual, ciclo, bônus disponíveis).
8. **Configurações** (segurança, permissões, dados de cobrança).

7.3 Modelagem de Dados

PostgreSQL

- `partner_events` → eventos criados/gerenciados.
- `partner_benefits` → promoções ativas.

- `feedback_snapshots` → feedbacks agregados exibidos.
- `partner_missions` → missões gamificadas concluídas ou em andamento.

Firestore

- Atualizações em tempo real (feedbacks recentes, novos check-ins).
- Dados do painel exibidos ao vivo.

Redis

- Cache para métricas do dashboard (`frequency_score`, `state`, KPIs).

7.4 APIs Essenciais

GET `/api/partners/panel/overview`

200

```
{
  "tier":"visionario",
  "status":"active",
  "frequency_score":85.4,
  "vibrational_state":"PICO",
  "checkins_last_7d":120,
  "feedback_avg":4.6,
  "missions_pending":[ "adicionar_fotos", "criar_evento" ]
}
```

POST `/api/partners/events`

Criação de evento (ver Camada 3.3).

GET `/api/partners/feedbacks`

Lista feedbacks agregados:

```
{
  "average_score":4.4,
  "keywords":["acolhedor","leve","criativo"],
  "recent":[ { "score":5,"keywords":["leve"] } ]
}
```

7.5 Regras de Negócio

SE...	ENTÃO...	Módulo
Parceiro Essencial	Bloquear módulos de eventos/benefícios	Controle de Tier
Parceiro Premium	Liberar eventos e promoções, bloquear IA Advisor	Billing
Parceiro Visionário	Liberar todos os módulos	Painel
Local em Colapso	Exibir alerta + sugerir missão de harmonização	IA Advisor
Feedback médio < 3.0	Painel sugere ações corretivas	Feedbacks

---

7.6 Observabilidade

Logs:

- `event_type`: `panel_loaded`, `profile_updated`, `event_created`, `benefit_created`, `advisor_suggestion_accepted`.
- `actor`: partner_id.
- `correlation_id`: por sessão.

Métricas:

- `panel_load_latency_p95`
- `events_created_total`
- `benefits_active_total`
- `advisor_suggestions_accepted_total`

Alertas:

- falhas >5% em POST `/events`
- latência > 1s em GET `/panel/overview`

7.7 UX e Gamificação

- **Missões exibidas** (checklist com pontos e recompensas).
- **Badges desbloqueados** (Guardião da Frequência, etc.).
- **Estados vazios bem tratados**:
  - Sem eventos → "Crie seu primeiro evento vibracional ✨".
  - Sem feedbacks → "Convide clientes a compartilhar energia".
  - Sem fotos → "Adicione imagens para ativar sua aura no mapa".

7.8 Segurança

- Sessão protegida por JWT curta + refresh.
- RBAC: permissões diferentes para *owner*, *manager*, *staff*.
- Logs de auditoria em toda alteração crítica (nome, categoria, docs).

⬅️ Fechamento da Camada 7

O Painel do Parceiro é o **centro de controle B2B do FriendApp**, onde cada ação do parceiro tem reflexo direto no ecossistema vibracional. Ele reúne gestão operacional, monetização e inteligência da IA Aurah Advisor, garantindo que os parceiros não apenas usem o sistema, mas **cresçam junto com a malha vibracional**.

📘 CAMADA 8 — ESTRUTURA DE BANCO DE DADOS DETALHADA

Stack: PostgreSQL (core relacional) • Firestore (tempo real) • Neo4j (grafo) • Redis (cache)

> Objetivo: definir todas as entidades, colunas, tipos, constraints, relacionamentos, índices e camadas auxiliares (views, MVs, triggers, auditoria, CDC, particionamento) para sustentar cadastro de locais, tiers/assinaturas, check-ins, feedbacks, eventos, benefícios, snapshots de score, antifraude, moderação e auditoria.

8.1 Visão ER (texto)

```
partner_owners (1)——<(N) partner_locations (1)——<(N) partner_events
                    |  └——<(N) partner_benefits
                    |  └——<(N) partner_checkins
```

```
                    │  └────<(N) partner_feedbacks
                    │  └────<(N) score_snapshots
                    │  └────<(N) anti_fraud_flags
                    └────(1) partner_verifications (1:1)
partners (jurídico) (1)────<(N) partner_plans (assinaturas)
audit_events (append-only) relaciona-se com quaisquer entidades
```

8.2 Convenções e Tipos

- **TZ:** `TIMESTAMPTZ` sempre.
- **IDs:** `UUID` gerados no app (`gen_random_uuid()` recomendado).
- **Enums:** criados via `CREATE TYPE`.
- **Geo:** `GEOGRAPHY(Point,4326)` (extensão PostGIS).
- **PII sensível:** schema separado `pii` com **Column-Level Encryption** (KMS).

8.2.1 Enums

```
CREATE TYPE tier AS ENUM ('essencial','premium','visionario');
CREATE TYPE billing_cycle AS ENUM ('monthly','annual');
CREATE TYPE sub_status AS ENUM ('active','inactive','pending_payment','cancelled','expired');
CREATE TYPE loc_status AS ENUM ('DRAFT','PENDING_VERIFICATION','UNDER_REVIEW','ACTIVE','REJECTE
D','BLOCKED');
CREATE TYPE vib_state AS ENUM ('COLAPSO','TRANSICAO','EXPANSAO','PICO');
CREATE TYPE verif_status AS ENUM ('PENDING','AUTO_APPROVED','REVIEW','REJECTED');
CREATE TYPE benefit_type AS ENUM ('desconto','brinde','combo');
CREATE TYPE actor_type AS ENUM ('OWNER','SYSTEM','ADMIN','USER');
```

8.3 DDL das Tabelas Principais (PostgreSQL)

> Observação: trechos a seguir estão prontos para migração inicial (001_init.sql). Ajuste nomes de schemas conforme padrão do seu projeto.

8.3.1 Usuários B2B (proprietários)

```
CREATE TABLE partner_owners (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  email CITEXT UNIQUE NOT NULL,
  phone_e164 VARCHAR(20),
  password_hash TEXT NOT NULL,
  is_email_verified BOOLEAN NOT NULL DEFAULT FALSE,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now()
);

CREATE INDEX idx_partner_owners_email ON partner_owners (email);
```

8.3.2 Parceiro jurídico (opcional se usar entidade distinta de owner)

```
CREATE TABLE partners (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
  legal_name TEXT NOT NULL,
  tax_id VARCHAR(32) UNIQUE, -- CNPJ
  created_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
```

8.3.3 Locais Parceiros

```
CREATE TABLE partner_locations (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  owner_id UUID NOT NULL REFERENCES partner_owners(id) ON DELETE CASCADE,
  partner_id UUID REFERENCES partners(id) ON DELETE SET NULL,
  name VARCHAR(120) NOT NULL,
  category VARCHAR(60) NOT NULL,
  description TEXT,
  address_full TEXT NOT NULL,
  geo GEOGRAPHY(Point,4326) NOT NULL,
  images TEXT[] CHECK (cardinality(images) BETWEEN 1 AND 10),
  intention VARCHAR(40),
  tier tier NOT NULL DEFAULT 'essencial',
  status loc_status NOT NULL DEFAULT 'DRAFT',
  frequency_score NUMERIC(5,2) NOT NULL DEFAULT 0.00,
  vibrational_state vib_state NOT NULL DEFAULT 'TRANSICAO',
  algorithm_version VARCHAR(12) NOT NULL DEFAULT 'v1.0',
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now()
);

-- Índices
CREATE INDEX idx_locations_owner ON partner_locations(owner_id);
CREATE INDEX idx_locations_status ON partner_locations(status);
CREATE INDEX idx_locations_geo ON partner_locations USING GIST (geo);
CREATE INDEX idx_locations_score_state ON partner_locations(vibrational_state, frequency_score DESC);
```

8.3.4 Verificação do Local (1:1)

```
CREATE TABLE partner_verifications (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  location_id UUID UNIQUE NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  cnpj VARCHAR(18),
  doc_owner_type VARCHAR(12),
  doc_owner_number TEXT, -- encriptado em pii.partner_verifications_sec
  proof_address_url TEXT,
  ocr_confidence NUMERIC(4,3) CHECK (ocr_confidence BETWEEN 0 AND 1),
  status verif_status NOT NULL DEFAULT 'PENDING',
  reason TEXT,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now()
);
```

> PII sensível: números de documentos devem ir para pii.partner_verifications_sec:

```
CREATE SCHEMA IF NOT EXISTS pii;

CREATE TABLE pii.partner_verifications_sec (
  verification_id UUID PRIMARY KEY REFERENCES partner_verifications(id) ON DELETE CASCADE,
  doc_owner_number_enc BYTEA NOT NULL -- ciphertext (KMS/PGP/pgcrypto)
);
```

8.3.5 Assinaturas / Planos

```
CREATE TABLE partner_plans (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  partner_id UUID NOT NULL REFERENCES partners(id) ON DELETE CASCADE,
  location_id UUID REFERENCES partner_locations(id) ON DELETE SET NULL,
  tier tier NOT NULL,
  billing_cycle billing_cycle,
  subscription_status sub_status NOT NULL,
  subscription_start TIMESTAMPTZ NOT NULL DEFAULT now(),
  subscription_end TIMESTAMPTZ,
  bonus_event_boosts INT NOT NULL DEFAULT 0,
  last_bonus_applied_at TIMESTAMPTZ,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now()
);

CREATE INDEX idx_plans_partner ON partner_plans(partner_id, subscription_status);
```

8.3.6 Eventos do Local

```
CREATE TABLE partner_events (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  partner_location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  title VARCHAR(255) NOT NULL,
  description TEXT,
  start_time TIMESTAMPTZ NOT NULL,
  end_time TIMESTAMPTZ NOT NULL,
  frequency_theme VARCHAR(80),
  is_highlighted BOOLEAN NOT NULL DEFAULT FALSE,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  CHECK (end_time > start_time)
);

CREATE INDEX idx_events_location_time ON partner_events(partner_location_id, start_time DESC);
```

8.3.7 Benefícios do Local

```
CREATE TABLE partner_benefits (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  partner_location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  type benefit_type NOT NULL,
  description TEXT NOT NULL,
```

```
  valid_until TIMESTAMPTZ,
  active BOOLEAN NOT NULL DEFAULT TRUE,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  updated_at TIMESTAMPTZ NOT NULL DEFAULT now()
);

CREATE INDEX idx_benefits_active ON partner_benefits(partner_location_id, active);
```

8.3.8 Check-ins (histórico relacional)

> Dado bruto chega no Firestore; consolidado vem para PG para relatórios e antifraude.

```
CREATE TABLE partner_checkins (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  user_id UUID NOT NULL,              -- pseudonimizado em views
  partner_location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  timestamp TIMESTAMPTZ NOT NULL DEFAULT now(),
  geo_validated BOOLEAN NOT NULL DEFAULT TRUE,
  device_hash TEXT,
  ip_asn INT,
  UNIQUE (user_id, partner_location_id, date_trunc('day', timestamp)) -- 1/dia
);

CREATE INDEX idx_checkins_location_time ON partner_checkins(partner_location_id, timestamp DESC);
```

8.3.9 Feedbacks (agregado relacional)

```
CREATE TABLE partner_feedbacks (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  checkin_id UUID REFERENCES partner_checkins(id) ON DELETE SET NULL,
  user_id UUID NOT NULL,
  partner_location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  vibration_feedback SMALLINT CHECK (vibration_feedback BETWEEN 0 AND 5),
  emojis TEXT[],
  keywords TEXT[],
  comment TEXT,
  quarantined BOOLEAN NOT NULL DEFAULT FALSE, -- antifraude
  created_at TIMESTAMPTZ NOT NULL DEFAULT now()
);

CREATE INDEX idx_feedbacks_location_time ON partner_feedbacks(partner_location_id, created_at DESC);
CREATE INDEX idx_feedbacks_quarantine ON partner_feedbacks(partner_location_id, quarantined);
```

8.3.10 Snapshots de Score (histórico)

```
CREATE TABLE score_snapshots (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  score NUMERIC(5,2) NOT NULL CHECK (score BETWEEN 0 AND 100),
  state vib_state NOT NULL,
  algorithm_version VARCHAR(12) NOT NULL,
  computed_at TIMESTAMPTZ NOT NULL,
```

```
  window_minutes INT NOT NULL DEFAULT 15
);

CREATE INDEX idx_snapshots_loc_time ON score_snapshots(location_id, computed_at DESC);
```

8.3.11 Flags de Antifraude

```
CREATE TABLE anti_fraud_flags (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  location_id UUID NOT NULL REFERENCES partner_locations(id) ON DELETE CASCADE,
  type TEXT NOT NULL,       -- ex: "attack_coordenado", "geo_mismatch"
  severity SMALLINT CHECK (severity BETWEEN 1 AND 5),
  count INT NOT NULL DEFAULT 1,
  first_seen TIMESTAMPTZ NOT NULL DEFAULT now(),
  last_seen TIMESTAMPTZ NOT NULL DEFAULT now(),
  status TEXT NOT NULL DEFAULT 'open' -- open|mitigated|closed
);

CREATE INDEX idx_af_flags_loc ON anti_fraud_flags(location_id, status);
```

8.3.12 Auditoria (append-only)

```
CREATE TABLE audit_events (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  occurred_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  actor_type actor_type NOT NULL,
  actor_id UUID,
  entity TEXT NOT NULL,
  entity_id UUID,
  action TEXT NOT NULL,
  payload_json JSONB,
  correlation_id UUID
);

CREATE INDEX idx_audit_entity ON audit_events(entity, entity_id, occurred_at DESC);
CREATE INDEX idx_audit_action_time ON audit_events(action, occurred_at DESC);
```

8.4 Particionamento & Performance

- `partner_checkins` e `partner_feedbacks`: particionar por **mês** (range em `timestamp`).
- `score_snapshots`: particionar por **mês** em `computed_at`.
- **Hot tables** com índices **covering** para queries mais frequentes (local + tempo).
- **Autovacuum** ajustado (work_mem, maintenance_work_mem) para tabelas de alto churn.

Exemplo de particionamento:

```
ALTER TABLE partner_checkins PARTITION BY RANGE (timestamp);
CREATE TABLE partner_checkins_2025_09 PARTITION OF partner_checkins
  FOR VALUES FROM ('2025-09-01') TO ('2025-10-01');
```

8.5 Views & Materialized Views

8.5.1 View Pública (p/ app) — dados não sensíveis

```
CREATE VIEW v_public_locations AS
SELECT
  id, name, category, intention,
  frequency_score, vibrational_state,
  ST_Y(geo::geometry) AS lat,
  ST_X(geo::geometry) AS lng
FROM partner_locations
WHERE status = 'ACTIVE';
```

8.5.2 MV — KPI de Engajamento por Local (últimos 30 dias)

```
CREATE MATERIALIZED VIEW mv_location_kpis_30d AS
SELECT
  pl.id AS location_id,
  COUNT(DISTINCT pc.id) AS checkins_30d,
  AVG(pf.vibration_feedback)::NUMERIC(4,2) AS avg_feedback_30d
FROM partner_locations pl
LEFT JOIN partner_checkins pc
  ON pc.partner_location_id = pl.id
  AND pc.timestamp >= now() - interval '30 days'
LEFT JOIN partner_feedbacks pf
  ON pf.partner_location_id = pl.id
  AND pf.created_at >= now() - interval '30 days'
  AND pf.quarantined = FALSE
GROUP BY pl.id;

-- Agendamento de refresh (via cron/pg_cron):
-- SELECT cron.schedule('refresh_kpis_30d', '*/15 * * * *', 'REFRESH MATERIALIZED VIEW CONCURRENTLY mv
_location_kpis_30d');
```

8.6 Triggers Importantes

- `partner_locations.updated_at` → trigger `BEFORE UPDATE` para manter timestamp.
- `partner_feedbacks` → trigger que atualiza campo `quarantined` conforme regras rápidas (ex.: blacklist device_hash).
- `partner_events` → trigger validação `end_time > start_time` (já há CHECK, trigger registra auditoria).

Exemplo:

```
CREATE OR REPLACE FUNCTION set_updated_at()
RETURNS TRIGGER AS $$
BEGIN
  NEW.updated_at = now();
  RETURN NEW;
END; $$ LANGUAGE plpgsql;

CREATE TRIGGER trg_locations_updated_at
BEFORE UPDATE ON partner_locations
```

```
FOR EACH ROW EXECUTE FUNCTION set_updated_at();
```

8.7 CDC (Change Data Capture)

- **WAL/Logical replication** para data lake/BI.
- Tabelas com alto valor analítico: `partner_checkins`, `partner_feedbacks`, `score_snapshots`, `partner_plans`, `audit_events`.
- **Debezium**/Kafka como ponte → alimenta **Aurah Kosmos** para modelos offline.

8.8 Backup & Restore

- **Backups diários** (base + WAL) → retenção 30 dias.
- **Provas de restore** semanais (staging).
- **Scripts** para restore direcionado de partições mensais (check-ins/feedbacks).

8.9 Firestore (estrutura sugerida)

- `checkins/{checkin_id}`
  - `user_id`, `location_id`, `geo`, `timestamp`, `valid`
- `feedbacks/{feedback_id}`
  - `checkin_id`, `location_id`, `score`, `keywords`, `created_at`, `quarantined`
- `panels/{location_id}`
  - `frequency_score`, `vibrational_state`, `kpis` (espelho do Redis/PG)

> Sincronização: workers lêem Firestore → aplicam antifraude → consolidam em PG + atualizam Redis.

8.10 Redis (chaves e TTL)

- `loc:score:{location_id}` → `{score, state, alg_ver}` (TTL 15 min)
- `loc:kpi:{location_id}` → `{checkins_7d, avg_fb}` (TTL 15 min)
- `lim:checkin:{user_id}:{location_id}:{date}` → rate-limit (TTL 24h)
- `idem:{route}:{idem_key}` → idempotência (TTL 24h)

8.11 Queries Comuns (exemplos executáveis)

Locais ativos por raio (3km) ordenados por score

```
SELECT id, name, frequency_score,
      ST_Distance(geo, ST_MakePoint(:lng,:lat)::geography) AS dist_m
FROM partner_locations
WHERE status='ACTIVE'
  AND ST_DWithin(geo, ST_MakePoint(:lng,:lat)::geography, 3000)
ORDER BY frequency_score DESC
LIMIT 50;
```

KPI de um local (últimos 7 dias)

```
SELECT
  COUNT(*) FILTER (WHERE pc.timestamp >= now() - interval '7 days') AS checkins_7d,
  AVG(pf.vibration_feedback) FILTER (WHERE pf.created_at >= now() - interval '7 days' AND pf.quarantined = FA
```

```
LSE) AS avg_fb_7d
FROM partner_locations pl
LEFT JOIN partner_checkins pc ON pc.partner_location_id = pl.id
LEFT JOIN partner_feedbacks pf ON pf.partner_location_id = pl.id
WHERE pl.id = :location_id;
```

Histórico de score (para gráficos)

```
SELECT computed_at, score, state
FROM score_snapshots
WHERE location_id = :location_id
  AND computed_at >= now() - interval '30 days'
ORDER BY computed_at;
```

8.12 Privacidade & Pseudonimização

- Views públicas NUNCA expõem `user_id`.
- Métricas exibidas apenas com **K-anonymity (≥10 contribuições)**.
- PII sensível segregada em `pii.*` com **RBAC** (somente `role:compliance`).
- **Right-to-be-forgotten:** rotinas para excluir/anonymizar dados de `user_id` em tabelas de alto volume (substituição por hash irreversível em históricos).

8.13 Migrações & Versionamento de Schema

- **Expand → Migrate → Contract** para alterações de colunas.
- **Flyway/Liquibase** com pastas `V001__init.sql`, `V002__...`.
- Cada alteração relevante do algoritmo grava `algorithm_version` nos snapshots.

8.14 Testes de Banco

- **Unitarios** (functions/triggers): `pgTAP`.
- **Carga**: cenários com 1k check-ins/min (partições + índices validados).
- **Integridade**: FK ON DELETE/UPDATE, constraints CHECK/UNIQUE.
- **Restauração**: testes semanais de restore com verificação de checksums.

8.15 Playbooks

- **Índice bloat↑** → REINDEX CONCURRENTLY fora de horário de pico.
- **Fila de recomputação atrasada** → escalar workers + verificar locks em snapshots.
- **Explosão de quarentenas** → inspecionar antifraude/telemetria (ASN/IP, device_hash).

🔚 Fechamento da Camada 8

Esta camada entrega **o esqueleto completo de dados** para o Sistema de Locais Parceiros: **safe-by-design, auditável, performático e escalável**. Com esse DB, os módulos de API (Cam. 9), IA (Cam. 6), Painel (Cam. 7) e Moderação/Antifraude (Cam. 10/20) trabalham sobre **bases confiáveis** e prontas para crescimento global.

📘 CAMADA 9 — APIs & ENDPOINTS (CONTRATO TÉCNICO)

> Objetivo: definir todas as rotas do módulo de Locais Parceiros, seus métodos, payloads, respostas, erros, autenticação, rate-limits, idempotência e observabilidade. Este contrato é a base para dev front-end, mobile e back-end integrarem sem ambiguidade.

9.1 Convenções

- **Base URL (prod):** `https://api.friendapp.com/v1/partners`
- **Headers padrão:**
  - `Authorization: Bearer <jwt>`
  - `Idempotency-Key: <uuid>` em POST críticos
  - `X-Client-Version: <semver>`
- **Escopos JWT:**
  - `user` → check-ins, feedbacks
  - `partner` → perfil, eventos, benefícios
  - `admin` → moderação
- **Rate-limits:**
  - 60 req/min por `user`
  - 200 req/min por `partner`
  - 1000 req/min por `admin/system`

9.2 Autenticação & Sessão

POST `/auth/login`

Body

```
{ "email":"owner@local.com", "password":"Str0ng!Pass" }
```

200

```
{ "access_token":"jwt", "refresh_token":"jwt", "owner_id":"uuid" }
```

Erros: `401 INVALID_CREDENTIALS`, `423 ACCOUNT_LOCKED`

POST `/auth/refresh`

Body

```
{ "refresh_token":"jwt" }
```

200

```
{ "access_token":"jwt" }
```

9.3 Locais

POST `/locations`

Cria local parceiro (tier default = essencial).

Body

```
{
  "name":"Café Alma Zen",
  "category":"cafe",
  "description":"Ambiente acolhedor",
  "address_full":"Rua X, 123 - SP",
  "geo":{"lat":-23.55,"lng":-46.63},
  "images":["https://cdn/img1.jpg"],
  "intention":"cura"
}
```

201

```
{ "location_id":"uuid", "status":"DRAFT" }
```

GET `/locations/{id}`

Retorna dados completos do local.

200

```
{
  "id":"uuid-local",
  "name":"Café Alma Zen",
  "tier":"premium",
  "frequency_score":82.5,
  "vibrational_state":"EXPANSAO",
  "benefits":[...],
  "events":[...]
}
```

PUT `/locations/{id}`

Atualiza perfil (exige `Idempotency-Key`).

200

```
{ "status":"updated", "triggered_vibrational_scan":true }
```

DELETE `/locations/{id}`

Soft delete (`status=BLOCKED`).

204

9.4 Verificação

POST `/locations/{id}/submit-verification`

Submete documentos.

Body

```
{
  "cnpj":"12.345.678/0001-90",
  "doc_owner_type":"CNH",
  "doc_owner_number":"99999999999",
  "proof_address_url":"https://cdn/docs.pdf"
}
```

202

```
{ "verification_id":"uuid", "status":"PENDING" }
```

GET `/locations/{id}/status`

200

```
{
  "status":"PENDING_VERIFICATION",
  "verification_status":"PENDING",
  "vibrational_state_preview":"TRANSICAO"
}
```

9.5 Check-ins & Feedbacks (Usuários)

POST `/checkins`

Body

```
{ "location_id":"uuid", "geo":{"lat":-23.55,"lng":-46.63} }
```

201

```
{ "checkin_id":"uuid", "status":"recorded" }
```

POST `/checkins/{id}/feedback`

Body

```
{
  "vibration_feedback":4,
  "emojis":["✨","🌿"],
  "keywords":["leve","acolhedor"],
  "comment":"Ambiente inspirador"
}
```

201

```
{ "status":"saved", "impact_on_score":true }
```

9.6 Eventos

POST /events

Cria evento.

Body

```
{
  "partner_location_id":"uuid-local",
  "title":"Círculo de Som",
  "start_time":"2025-09-20T19:00:00Z",
  "end_time":"2025-09-20T22:00:00Z"
}
```

201

```
{ "event_id":"uuid", "status":"created" }
```

GET /events/{id}

Retorna evento.

9.7 Benefícios

POST /benefits

Body

```
{
  "partner_location_id":"uuid-local",
  "type":"desconto",
  "description":"10% em todo cardápio",
  "valid_until":"2025-09-30T23:59:59Z"
}
```

201

```
{ "benefit_id":"uuid", "status":"active" }
```

9.8 Assinaturas & Planos

POST /subscribe

Assinar ou upgrade.

Body

```
{
  "tier":"visionario",
  "billing_cycle":"annual",
  "payment_token":"tok_abc123"
}
```

200

```
{
  "status":"active",
  "subscription_end":"2026-09-14T00:00:00Z",
  "bonus_event_boosts":1
}
```

9.9 Impulsionamento

POST `/events/{id}/boost`

- Se `bonus_event_boosts > 0`: aplica sem cobrança.
- Caso contrário: processa pagamento.

200

```
{ "status":"boost_applied", "remaining_bonus":0 }
```

9.10 Erros Padronizados

```
{ "code":"VALIDATION_ERROR", "message":"Latitude obrigatória", "fields":{"geo.lat":"required"} }
```

- **401**: `UNAUTHORIZED`
- **403**: `TIER_NOT_ALLOWED`
- **404**: `NOT_FOUND`
- **409**: `DUPLICATE_REQUEST`
- **422**: `VALIDATION_ERROR`
- **429**: `RATE_LIMITED`
- **500**: `INTERNAL_ERROR`

9.11 Observabilidade

- **Logs**: `event_type`, `actor`, `entity`, `action`, `payload`, `correlation_id`.
- **Métricas**: `requests_total`, `errors_total{code}`, `latency_p95`, `idempotency_reuse_count`.
- **Alertas**:
  - erro 5xx > 1%/5min
  - p95 POST `/checkins` > 800ms
  - falhas webhook pagamento > 5%

⬅ END Fechamento da Camada 9

O contrato de APIs garante **clareza, segurança e robustez**. Cada rota foi definida com exemplos de payload, erros, autenticação e observabilidade. Isso permite desenvolvimento paralelo entre back e front, reduzindo ambiguidade e aumentando a confiança dos parceiros.

📘 CAMADA 10 — SEGURANÇA, MODERAÇÃO E GOVERNANÇA

> Objetivo: especificar as políticas, algoritmos e mecanismos de segurança, moderação e governança que protegem o ecossistema de Locais Parceiros contra fraudes, incoerências vibracionais, abusos e ataques coordenados. Esta camada garante justiça, transparência e confiança para parceiros e usuários.

10.1 Dimensões de Segurança

1. **Segurança Técnica**: autenticação, rate-limits, antifraude.
2. **Segurança Vibracional**: monitoramento de estados de colapso e coerência energética.
3. **Moderação de Conteúdo**: denúncias, auditoria e bloqueios.
4. **Governança de Algoritmo**: transparência, explicabilidade e rollback.

10.2 Segurança Técnica

- **Autenticação**: JWT curto (15min) + refresh (7 dias).
- **RBAC**: permissões por escopo (`user`, `partner`, `admin`).
- **Rate-limit**: Redis leaky-bucket, por IP + user_id.
- **Idempotência**: `Idempotency-Key` obrigatório em POST críticos.
- **Webhooks**: assinados com HMAC + replay protection (timestamp window 5min).
- **Geo-fencing**: check-ins válidos apenas se `dist ≤ 100m`.
- **PII** segregada em schema `pii`, acesso via role compliance.

10.3 Moderação Automática

Regras (SE → ENTÃO)

SE...	ENTÃO...	Módulo
Score < 40 por 7 dias	Ativar Missão de Harmonização	IA Aurah
Ataque de feedbacks detectado	Quarentena de feedbacks	Antifraude
Conteúdo impróprio em descrição	Ocultar até revisão	Moderação
Evento criado sem geo válido	Bloquear publicação	API
Benefício incoerente (spam)	Suspender benefício	Moderação

10.4 Denúncias de Usuário

Fluxo

1. Usuário clica em *"Denunciar Local"* no app.
2. Escolhe motivo:
   - Informações falsas
   - Energia incoerente
   - Local inexistente
   - Discurso de ódio / abuso
3. Denúncia salva em `partner_reports`.
4. Pipeline híbrido: IA → revisão humana (se necessário).

API

`POST /api/partners/locations/{id}/report`

```
{ "reason":"Local inexistente", "comment":"Não encontrei no endereço informado" }
```

10.5 Antifraude

- **Sinais**: device_id repetido, IPs datacenter, entropia baixa em feedbacks, padrões coordenados.
- **Ações**:
  - Peso reduzido em score.
  - Quarentena automática.
  - Shadow-ban para reincidentes.
- **Tabela** `anti_fraud_flags` registra flags com severidade e status.

10.6 Governança de Algoritmo

- Cada snapshot salva `algorithm_version`.
- Mudanças de algoritmo publicadas em changelog para parceiros.
- Parceiros podem ver no painel o breakdown do score.
- IA sempre retorna explicabilidade (`top-3 fatores que impactaram seu score`).
- Rollback automático se mudança gerar anomalias (ex: p95 cálculo > 5s).

10.7 Observabilidade

Logs

- `event_type`: `fraud_detected`, `report_submitted`, `location_blocked`.
- `actor`: user_id / partner_id / system.
- `payload`: motivo, severidade, ação tomada.

Métricas

- `reports_submitted_total`
- `fraud_flags_total`
- `locations_in_harmonization_total`
- `state_distribution{colapso,transicao,expansao,pico}`

Alertas

- 10 denúncias em 1h em mesmo local.
- taxa de fraude >5% em check-ins do dia.
- score médio global caindo >20% em 24h.

10.8 UX de Governança (Parceiro)

- Painel exibe:
  - Score atual + breakdown.
  - Avisos de incoerência: *“Parte dos feedbacks está sob revisão antifraude”*.
  - Missões de harmonização em caso de colapso.
  - Canal de contestação com SLA 48h.

🔚 Fechamento da Camada 10

A camada de Segurança, Moderação e Governança garante que o sistema seja **justo, confiável e transparente**. Protege contra manipulações, acolhe denúncias, educa parceiros e mantém a integridade da Economia da Consciência. A IA Aurah Kosmos atua não só como avaliadora, mas como **curadora vibracional e guardiã ética**.

📘 CAMADA 11 — INTEGRAÇÕES E DEPENDÊNCIAS CÍCLICAS DO ECOSSISTEMA

> Objetivo: mapear todas as integrações do Sistema de Locais Parceiros com os outros módulos do FriendApp, descrevendo entradas, saídas, APIs envolvidas, triggers de IA, dependências de banco e fluxos cíclicos de dados. Esta camada garante que os devs entendam onde este módulo consome e onde entrega dados, evitando silos e promovendo a interdependência do ecossistema.

11.1 Integrações de Entrada (dados que chegam ao módulo)

Origem	Tipo	Descrição
Cadastro Consciente & Perfil Vibracional	Usuário → Local	Usuários já chegam classificados por intenção vibracional; IA usa esse dado para sugerir locais.
Aurah Kosmos (IA central)	IA → Locais	Fornece matching vibracional, análise de sentimento e detecção de fraude.
Sistema de Pagamentos	Billing → Locais	Dados de assinatura, upgrades/downgrades de planos e créditos de impulsionamento.
Eventos	Eventos → Locais	Criação e associação de eventos a um local.
Feedbacks de Usuário	Usuário → Locais	Check-ins energéticos e avaliações anônimas.

11.2 Integrações de Saída (dados que este módulo fornece)

Destino	Tipo	Descrição
Mapa de Frequência Vibracional	Locais → Mapa	Envia aura + estado energético atualizado.
Feed Sensorial	Locais → Feed	Envia cards de locais, eventos, promoções e benefícios ativos.
Modo Bora / Resenhas	Locais → Matching	Sugere pontos de encontro compatíveis vibracionalmente.
Aurah Kosmos (IA)	Locais → IA	Envia feedbacks, check-ins e estados vibracionais para treinar e ajustar recomendações.
Painel do Parceiro (B2B)	Locais → Painel	Exibe métricas, score vibracional e sugestões.
Sistema de Selos & Gamificação	Locais → Gamificação	Gera badges e missões com base em performance vibracional.

11.3 Fluxos Técnicos (cíclicos)

1. **Check-in Energético**

   Usuário faz check-in → feedback processado → IA recalcula score → novo snapshot enviado para Mapa/Feed → exibido no app → usuários reagem novamente.

2. **Eventos**

   Parceiro cria evento → evento aparece no Feed/Mapa → usuários interagem (check-ins) → dados retornam para IA → IA ajusta sugestões futuras.

3. **Aurah Advisor**

IA analisa score + feedbacks → envia sugestões ao parceiro no painel → parceiro ajusta perfil/eventos → novos dados voltam para IA → ciclo se repete.

4. **Assinaturas Premium/Vis.**

   Pagamento ativo → libera recursos (benefícios, boosts, IA) → uso desses recursos gera métricas extras → métricas alimentam o sistema de valor → aumenta percepção de benefício → reforça upgrades.

11.4 APIs Compartilhadas

- **Com Sistema de Usuários**

  `GET /users/{id}/profile` → fornece intenção vibracional e preferências.

- **Com Sistema de Pagamentos**

  `POST /payments/webhooks` → atualiza `partner_plans`.

- **Com Feed Sensorial**

  `GET /feed/locals` → recebe lista curada pela IA, incluindo locais parceiros.

- **Com Mapa de Frequência**

  `GET /map/locations` → fornece snapshot vibracional.

11.5 Dependências de Banco

- **PostgreSQL**: integra com módulos de billing e auditoria.
- **Firestore**: integra com feed, mapa e painel mobile.
- **Neo4j**: integra com IA para analisar conexões locais-eventos-usuários.
- **Redis**: fornece cache para leitura rápida em todos os módulos (feed, mapa, painel).

11.6 Observabilidade Cíclica

- Cada integração registra logs com:
  - `source_module`
  - `target_module`
  - `event_type`
  - `payload_hash`
  - `latency_ms`

Métricas monitoradas:

- `integration_latency_p95{source,target}`
- `integration_failures_total{source,target}`
- `feedback_loop_time_avg` (tempo entre check-in e reflexo no feed).

🔚 Fechamento da Camada 11

O módulo de Locais Parceiros é um dos **mais interdependentes** do FriendApp. Ele não apenas recebe dados (usuários, billing, IA), mas devolve em forma de **insights, eventos e estados vibracionais**. Essa ciclicidade é o que transforma o FriendApp em uma rede viva e inteligente, onde o físico e o digital se alimentam mutuamente.

📘 CAMADA 12 — UX/UI SENSORIAL DO PAINEL DO PARCEIRO

> Objetivo: definir a experiência visual e sensorial do painel B2B dos parceiros, detalhando telas, estados, interações, feedbacks visuais e regras de design. Essa camada garante que o painel seja moderno, intuitivo e vibracionalmente coerente, equilibrando clareza de gestão com estética energética.

12.1 Princípios de Design

1. **Sensorialidade:** cores, luz e movimento transmitem o estado vibracional do local.
2. **Clareza Operacional:** nenhuma ação deve levar mais de 3 cliques.
3. **Gamificação:** missões, badges e progressos visuais incentivam engajamento.
4. **Consistência:** UI alinhada com paleta e tipografia definidas no manual de identidade FriendApp.
5. **Acessibilidade:** contraste AA, suporte a screen readers e labels claras.

12.2 Estrutura do Painel (Navegação)

- **Menu Lateral:**
  - Dashboard
  - Perfil do Local
  - Eventos
  - Benefícios
  - Feedbacks
  - Aurah Advisor (Visionário)
  - Meu Plano
  - Configurações
- **Área Central:** cards dinâmicos, gráficos, listas, formulários.
- **Header:** nome do local + estado vibracional atual (score + selo).

12.3 Dashboard Inicial

Cards principais

- **Estado Vibracional:** aura animada + score (0–100).
- **Check-ins últimos 7 dias:** número + variação % vs semana anterior.
- **Feedback médio:** nota (0–5) + nuvem de palavras positivas/negativas.
- **Missões Pendentes:** checklist de ações sugeridas.

Visualização

- Aura pulsante em dourado se o local estiver em *Pico*.
- Aura azul suave em *Expansão*.
- Aura instável (piscando) em *Transição*.
- Aura esmaecida em *Colapso*.

12.4 Perfil do Local (Edição)

- Campos editáveis: nome, descrição, categoria, intenção vibracional, galeria de fotos.
- Visualização imediata da aura simulada após alterações (prévia IA).
- Botão **“Salvar & Recalcular Frequência”** → dispara trigger para IA.

12.5 Eventos

- Lista de eventos futuros/passados.
- Botão **“Criar Evento”** → formulário com título, descrição, data, frequência.
- Eventos ativos podem ser **impulsionados** (se Visionário ou Premium com créditos).
- KPIs exibidos em cards: visualizações, confirmações, check-ins no evento.

12.6 Benefícios & Promoções

- Cards de benefícios ativos (ex.: “10% desconto”).
- CTA para adicionar novos benefícios.
- Status de validade exibido em cores (verde = ativo, laranja = perto de expirar, vermelho = expirado).

12.7 Feedbacks

- Lista de avaliações anônimas recentes.
- Filtros: positivas, negativas, por data.
- Resposta rápida → campo de 140 caracteres (com IA sugerindo tom cordial).
- Gráfico de linha mostrando evolução da frequência média.

12.8 Aurah Advisor (Visionário)

- Feed de sugestões inteligentes da IA (cards horizontais):
  - Ex.: “75% do seu público recente tem perfil criativo → Crie um evento de arte imersiva.”
- Botões: *“Aplicar Agora”* ou *“Descartar”*.
- Histórico de sugestões aceitas/rejeitadas.

12.9 Meu Plano

- Exibe tier atual, ciclo (mensal/anual), status de pagamento.
- Bônus disponíveis (ex.: “1 Boost Gratuito restante este mês”).
- CTA de **Upgrade** com comparativo de tiers lado a lado.

12.10 Estados Vazios e Mensagens de Apoio

- Sem fotos → *“Adicione imagens para ativar sua aura no mapa ✨”*.
- Sem eventos → *“Crie seu primeiro evento vibracional e atraia visitantes.”*
- Sem feedbacks → *“Convide seus visitantes a compartilhar energia.”*

12.11 Observabilidade de UI

- Eventos de tracking enviados para analytics:
  - `panel_load`
  - `event_created`
  - `benefit_created`
  - `advisor_suggestion_applied`

Payload inclui `partner_id`, `module`, `latency_ui_ms`.

🔚 Fechamento da Camada 12

O Painel do Parceiro não é apenas um admin: é uma **extensão vibracional do local**, onde cada clique reverbera em dados energéticos. A UI une **estética sensorial + clareza funcional**, garantindo engajamento, confiança e alinhamento com a malha vibracional global do FriendApp.

📘 CAMADA 13 — MÉTRICAS DE SUCESSO E KPIs VIBRACIONAIS

> Objetivo: definir as métricas que medem o sucesso do Sistema de Locais Parceiros. Esta camada garante que o FriendApp não dependa apenas de métricas tradicionais de negócio, mas também de indicadores vibracionais, traduzindo energia em dados acionáveis para parceiros, usuários e investidores.

13.1 Categorias de KPIs

1. **Operacionais** → uso técnico do sistema.
2. **Comerciais** → impacto financeiro e retenção.
3. **Vibracionais** → impacto energético no ecossistema.
4. **Gamificação** → engajamento em missões e selos.

13.2 KPIs Operacionais

Indicador	Definição	Fonte
Nº de Check-ins	Quantidade total por local no período	`partner_checkins`
Feedbacks Coletados	Nº de feedbacks anônimos válidos	`partner_feedbacks`
Eventos Criados	Total de eventos criados por tier	`partner_events`
Missões Completas	Nº de missões gamificadas concluídas	`partner_missions`
Acesso ao Painel	Nº de logins únicos no painel	Logs de sessão

13.3 KPIs Comerciais

Indicador	Definição	Fonte
Parceiros Ativos	Nº de locais com status `ACTIVE`	`partner_locations`
Assinaturas Pagas	Contagem de Premium + Visionário ativos	`partner_plans`
Taxa de Upgrade	% de Essenciais → Premium/Vis.	`audit_events`
Retenção de Parceiros (MRR churn)	% de parceiros que mantêm assinatura	Stripe / FriendPay
Uso de Boosts	Nº de impulsionamentos por mês	`partner_events` + `partner_plans`

13.4 KPIs Vibracionais

Indicador	Definição	Fonte
Score Médio Global	Média ponderada de `frequency_score`	`score_snapshots`
Distribuição de Estados	% em Colapso, Transição, Expansão, Pico	`score_snapshots`
Curva de Recuperação	Tempo médio de recuperação de Colapso → Expansão	Logs IA
Palavras-chave Positivas	Top keywords de feedbacks (NLP)	`partner_feedbacks`
Feedback Positivo (%)	% de avaliações ≥ 4	`partner_feedbacks`

13.5 KPIs de Gamificação

Indicador	Definição	Fonte
Missões de Harmonização Concluídas	Nº de locais que saíram de Colapso via missões	`partner_missions`
Selos Atribuídos	Quantidade de badges entregues a parceiros	Gamificação
Pontos Acumulados	Pontos ganhos em interações vibracionais	Logs Painel
Sugestões Aurah Aplicadas	% de sugestões aceitas	IA Advisor

13.6 Visualização no Painel (Parceiro)

Dashboard inclui:

- **Score Atual** + estado (com explicação de fatores).
- **Check-ins últimos 30d** (gráfico de linha).
- **Feedback Médio** (card + nuvem de palavras).
- **Missões Pendentes / Concluídas** (checklist).
- **Eventos Criados & Impacto** (gráfico barras).

13.7 Observabilidade & Alertas

Métricas monitoradas globalmente:

- `checkins_total{location_id}`
- `feedback_avg{location_id}`
- `frequency_score_avg{city}`
- `partners_active_total`
- `upgrades_total`

Alertas:

- score médio < 50 em uma região.
- churn de parceiros > 10% mês.
- explosão de feedbacks negativos em 1h.

13.8 Exemplo de Query (BI/Analytics)

```
SELECT
  pl.id,
  pl.name,
  COUNT(DISTINCT pc.id) AS checkins_30d,
  AVG(pf.vibration_feedback) AS avg_fb,
  ss.score AS last_score
FROM partner_locations pl
LEFT JOIN partner_checkins pc ON pc.partner_location_id = pl.id
  AND pc.timestamp >= now() - interval '30 days'
LEFT JOIN partner_feedbacks pf ON pf.partner_location_id = pl.id
  AND pf.created_at >= now() - interval '30 days'
LEFT JOIN LATERAL (
  SELECT score FROM score_snapshots s
  WHERE s.location_id = pl.id
  ORDER BY s.computed_at DESC
  LIMIT 1
) ss ON true
WHERE pl.status='ACTIVE'
```

```
GROUP BY pl.id, pl.name, ss.score;
```

🔚 Fechamento da Camada 13

As **Métricas de Sucesso e KPIs Vibracionais** traduzem a missão do FriendApp em números claros e acionáveis. Parceiros sabem como melhorar, investidores entendem impacto e os devs têm visibilidade para ajustar algoritmos. O resultado é um sistema **orientado por dados vibracionais e de negócio ao mesmo tempo**.

📘 CAMADA 14 — ESTRATÉGIA DE GO-TO-MARKET B2B (PARCEIROS LOCAIS)

> Objetivo: definir a estratégia técnica e operacional para atrair, converter e reter locais parceiros nas primeiras ondas de lançamento do FriendApp. Essa camada conecta marketing, onboarding B2B, métricas e tecnologia, garantindo que o sistema seja adotado e percebido como valioso desde o dia 1.

14.1 Princípios da Estratégia

1. **Velocidade com Qualidade** → trazer rápido massa crítica de locais, mas mantendo filtros vibracionais.
2. **Valor Tangível para o Parceiro** → métricas, eventos e visibilidade desde o primeiro uso.
3. **Gamificação Comercial** → onboarding com missões e badges de fundadores.
4. **Modelo Escalável** → replicável de cidade para cidade.

14.2 Fases de Go-to-Market

Fase	Período	Objetivo
Piloto	3 meses	Ativar 50–100 locais em 2 cidades iniciais (SP/RJ).
Expansão Regional	6–12 meses	Escalar para 1000 locais em 5 capitais.
Escala Nacional	18–24 meses	10k locais ativos, integração com grandes redes.

14.3 Aquisição de Parceiros

Estratégias

- **Onboarding Pessoal (Piloto):** time do FriendApp faz cadastro guiado.
- **Campanhas de Fundadores:** primeiros parceiros recebem selo “Fundador”.
- **Eventos de Lançamento:** convites exclusivos para parceiros premium/visionários.
- **Parcerias Estratégicas:** fechar com associações de bares, cafés, coworkings.

APIs de suporte

- `POST /api/partners/leads` → registrar leads captados.
- `POST /api/partners/onboarding` → criar fluxo de entrada assistido.

14.4 Conversão (Free → Pago)

- **Trial Premium**: 30 dias grátis ao completar cadastro 100%.
- **IA Advisor Demo**: Visionário tem prévia de sugestões da Aurah.
- **Comparativo de Tiers no Painel**: mostrar claramente diferença de recursos.

- **Gamificação**: upgrade desbloqueia missões e impulsionamentos extras.

14.5 Retenção

- **Dashboard de Valor**: mostrar impacto vibracional (check-ins, feedbacks, eventos).
- **Relatórios Mensais**: enviados por e-mail com resumo de métricas.
- **Missões de Expansão**: desafios recorrentes que mantêm engajamento.
- **Suporte Prioritário**: canais exclusivos para Visionários.

14.6 Métricas de Go-to-Market

KPI	Meta Piloto	Meta Regional	Meta Nacional
Parceiros Ativos	100	1.000	10.000
% Upgrade p/ Pago	20%	35%	50%
Churn Mensal	< 5%	< 3%	< 2%
Check-ins Médios / Local	50/mês	100/mês	200/mês

14.7 Ferramentas de Suporte Técnico

- **Painel Interno (Admin B2B):** gestão de leads, status de onboarding, tiers.
- **Webhooks CRM:** integração com ferramentas externas de marketing.
- **Tracking:** `onboarding_completed_total`, `trial_started_total`, `trial_to_paid_conversion_rate`.

14.8 Observabilidade

Logs:

- `event_type`: `lead_created`, `trial_started`, `plan_upgraded`.
- `actor`: partner_id / system.

Métricas:

- `active_partners_total`
- `conversion_trial_to_paid_rate`
- `avg_time_to_upgrade_days`

Alertas:

- taxa de upgrade < 10% (em piloto).
- churn > 5% mês.

🔚 Fechamento da Camada 14

A Estratégia de Go-to-Market garante que o FriendApp conquiste **massa crítica de locais parceiros** sem perder essência vibracional. Ela conecta marketing, onboarding gamificado, IA Advisor e relatórios de valor, garantindo que o parceiro veja benefícios imediatos e escolha evoluir para tiers pagos.

📘 CAMADA 15 — AURAH ADVISOR FOR PARTNERS + INTELIGÊNCIA PREDITIVA

> Objetivo: descrever a inteligência consultiva exclusiva para parceiros Visionários, a Aurah Advisor, que combina análise de dados, predição vibracional e sugestões acionáveis. É a camada onde a IA deixa de ser apenas

> avaliadora e passa a ser mentora vibracional, transformando dados em estratégias práticas para o parceiro.

15.1 Função Estratégica

1. **Orientar** → traduzir dados vibracionais em recomendações concretas.
2. **Predizer** → identificar tendências futuras no score e no público.
3. **Educar** → sugerir missões e boas práticas de harmonização.
4. **Comercializar** → ajudar o parceiro a criar eventos e benefícios alinhados com seu público energético.

15.2 Escopo de Acesso

Tier	Acesso ao Advisor
Essencial	❌ Nenhum
Premium	❌ Nenhum
Visionário	✅ Acesso total

15.3 Fontes de Dados Processadas

- **Check-ins energéticos**: volume, frequência, consistência geográfica.
- **Feedbacks vibracionais**: análise semântica, emojis, keywords.
- **Eventos**: presença, feedback pós-evento, curva de impacto.
- **Benefícios**: adesão dos usuários às promoções.
- **Histórico de score**: evolução temporal, quedas e picos.
- **Perfis de usuários visitantes**: clusterização por intenção vibracional.

15.4 Sugestões Geradas

Exemplos de insights que a IA gera:

Tipo	Exemplo de Insight
Horários	“Seu local atinge maior pico vibracional às sextas, entre 18h–22h.”
Eventos	“75% dos seus visitantes recentes têm perfil ‘Alquimista Sensorial’. Sugestão: evento de música introspectiva.”
Harmonização	“Feedbacks negativos sobre iluminação aumentaram. Sugestão: ajuste do ambiente com luz âmbar.”
Comparativo	“Você está 20% acima da média de locais similares em sua região.”

15.5 APIs

GET `/api/partners/advisor/suggestions`

200


```
{
  "location_id":"uuid-local",
  "suggestions":[
    {
      "id":"sug-123",
      "type":"event",
      "message":"Crie um evento de arte imersiva",
      "confidence":0.87,
      "expected_impact":"+15% check-ins"
    }
```

```
  ]
}
```

POST `/api/partners/advisor/accept`

Body

```
{ "suggestion_id":"sug-123" }
```

200

```
{ "status":"applied", "mission_created":true }
```

15.6 Pipeline Técnico

```
[Dados Brutos]
  → [Normalização]
  → [Antifraude]
  → [Clusterização de usuários]
  → [Predição (ML - regressão temporal, NLP)]
  → [Geração de Sugestões com confiança + impacto esperado]
  → [Persistência em Firestore/PG]
  → [Exibição no Painel Visionário]
```

15.7 Observabilidade

Logs:

- `event_type`: `advisor_suggestion_generated`, `advisor_suggestion_applied`.
- `actor`: partner_id.
- `payload`: tipo, impacto esperado, confiança.

Métricas:

- `advisor_suggestions_total`
- `advisor_acceptance_rate`
- `advisor_suggestion_latency_p95`

Alertas:

- taxa de aceitação < 10% (sugestões pouco úteis).
- geração de sugestões > 5s (latência).

15.8 UX no Painel

- Sugestões exibidas em **cards horizontais**, com:
  - Mensagem clara.
  - Confiança (%).
  - Impacto esperado.
  - Botões: **Aplicar Agora** ou **Descartar**.

- Histórico de sugestões aceitas/rejeitadas.
- Badge extra quando uma sugestão aplicada gera resultado positivo (ex.: aumento de score > 10%).

🔚 Fechamento da Camada 15

A **Aurah Advisor for Partners** transforma o parceiro Visionário em um **cocriador estratégico**, usando dados vibracionais para orientar decisões reais. É a ponte entre tecnologia, energia e negócio, garantindo que cada local se mantenha **em expansão e relevância no ecossistema**.

📘 CAMADA 16 — PAINEL DE OBSERVABILIDADE ENERGÉTICA + MÉTRICAS SENSORIAIS

> Objetivo: definir o painel exclusivo (Tier Visionário) que revela leituras energéticas avançadas, histórico de estados vibracionais e métricas sensoriais profundas do local. É o módulo que transforma dados ocultos em visualizações intuitivas, permitindo que o parceiro acompanhe, compare e ajuste sua vibração em tempo real.

16.1 Princípios

1. **Transparência Total:** mostrar a composição do score e seus fatores.
2. **Predição:** exibir tendências e alertas de possíveis quedas/picos.
3. **Granularidade:** métricas diárias e horárias, não só médias.
4. **Comparação:** benchmarking com locais similares na região.
5. **Explicabilidade:** IA sempre indica *"por que"* algo mudou.

16.2 Estrutura do Painel

Seções

- **Resumo Energético Atual**
- **Histórico de Frequência** (gráficos de linha)
- **Mapa de Calor Emocional** (emoções dominantes por horário/dia)
- **Radar de Alinhamento Coletivo** (quanto o público está alinhado à intenção do espaço)
- **Benchmarking Vibracional** (comparativo com locais similares)
- **Alertas & Predições** (detecção de risco de colapso ou oportunidade de pico)

16.3 Métricas Sensoriais Disponíveis

Métrica	Definição	Visualização
Curva de Score	evolução do `frequency_score` (últimos 30/90 dias)	gráfico de linha
Distribuição de Estados	% de tempo em cada estado (Pico/Expansão/Transição/Colapso)	gráfico de pizza
Check-ins por Hora	fluxo de visitantes vibracionais	heatmap
Feedback Médio Diário	nota média dos feedbacks/dia	gráfico de barras
Palavras-Chave Dominantes	termos mais citados em feedbacks	nuvem de palavras
Tempo de Recuperação	dias para sair de Colapso → Expansão	indicador numérico
Radar de Emoções	categorias emocionais mapeadas por NLP	gráfico radial
Afinidade de Público	% de visitantes compatíveis com intenção vibracional	card + gráfico radial

---

16.4 Modelagem de Dados

PostgreSQL

- `score_snapshots` → histórico (score, state, algorithm_version).
- `partner_feedbacks` → keywords + sentimentos.
- `partner_checkins` → timestamps, horários de maior fluxo.

Firestore

- `panels/{location_id}` → snapshots em tempo real para UI.

Redis

- cache de métricas de curto prazo (últimas 24h).

16.5 APIs

GET `/api/partners/panel/observability`

200


```
{
  "score_history":[{"ts":"2025-09-10T00:00Z","score":78,"state":"EXPANSAO"}],
  "state_distribution":{"PICO":20,"EXPANSAO":50,"TRANSICAO":25,"COLAPSO":5},
  "checkins_hourly":[10,22,34,15,...],
  "avg_feedback_daily":[4.2,4.5,4.1,3.8],
  "keywords":["leve","acolhedor","introspectivo"],
  "emotions_radar":{"alegria":40,"calma":30,"caos":10,"inspiração":20},
  "benchmark":{"local_avg":75,"region_avg":68,"percentile":82}
}
```

16.6 Observabilidade Técnica

Logs

- `event_type`: `observability_viewed`, `score_trend_predicted`, `alert_generated`.
- `actor`: partner_id.
- `payload`: métricas exibidas.

Métricas

- `observability_requests_total`
- `observability_latency_p95`
- `alerts_triggered_total`

Alertas

- falha na geração de score trend > 5%
- latência > 1s na API `/observability`

16.7 UX no Painel

- **Aura animada** exibida junto ao gráfico de score.
- **Predições da IA** aparecem como mensagens:
  - *"Sua frequência tende a cair nas próximas 48h devido a aumento de feedbacks negativos."*

- **Benchmarking** exibido com barras comparativas.
- **Heatmap de Check-ins** colorido (verde = alto fluxo positivo, vermelho = baixo fluxo/negativo).

🔚 Fechamento da Camada 16

O **Painel de Observabilidade Energética** é um **microscópio vibracional** para os parceiros Visionários. Ele revela padrões invisíveis, dá contexto sobre a evolução energética e permite decisões baseadas em dados reais. É a ferramenta que transforma o parceiro em um **gestor consciente da vibração do seu espaço**.

📘 CAMADA 17 — CENÁRIOS DE USO E FLUXOS EM CIDADES PILOTO

> Objetivo: demonstrar o funcionamento real do Sistema de Locais Parceiros em cidades piloto, conectando estratégia de implantação, onboarding de parceiros e experiência dos usuários. Essa camada garante que os devs, equipe de growth e operação entendam como a teoria se traduz em prática viva.

17.1 Escolha das Cidades Piloto

Critérios Técnicos

- Alta densidade de bares, cafés, coworkings e espaços culturais.
- Comunidades abertas a inovação e espiritualidade urbana.
- Ecossistemas de eventos frequentes (música, wellness, networking).
- Hubs estratégicos para mídia e growth.

Exemplo de Fase 1

- **São Paulo (SP):** Vila Madalena, Pinheiros, Liberdade.
- **Rio de Janeiro (RJ):** Lapa, Botafogo, Jardim Botânico.
- **Florianópolis (SC):** Lagoa da Conceição, Campeche.

17.2 Fluxo de Onboarding em Cidades Piloto

1. **Curadoria de Locais Fundadores** → seleção de 50–100 espaços vibracionais.
2. **Onboarding Guiado** → cadastro presencial ou remoto assistido pelo time FriendApp.
3. **Ativação de Missões Iniciais** → completar perfil, criar 1º evento, responder feedbacks.
4. **Eventos Inaugurais** → criação de experiências vibracionais com check-ins incentivados.
5. **Feedback Coletado** → IA Aurah ajusta curva vibracional inicial.
6. **Relatórios & Gamificação** → locais recebem primeiros insights + selo de Fundadores.

17.3 Jornada do Usuário (Cidade Piloto)

```
[Usuário instala FriendApp]
  → [Cadastro + Perfil Vibracional]
  → [Exploração do Mapa Vibracional da Cidade]
  → [Descobre locais fundadores em destaque]
  → [Faz check-in em evento inaugural]
  → [Deixa feedback vibracional]
```

→ [IA Aurah recomenda novos locais compatíveis]

17.4 APIs Ativadas

- `GET /map/locations` → puxa apenas locais ativos em cidade piloto.
- `GET /feed/locals` → feed prioriza "locais fundadores".
- `POST /checkins` + `POST /feedbacks` → dados entram em pipeline vibracional.
- `GET /panel/observability` → painel avançado para Visionários fundadores.

17.5 Métricas de Sucesso em Pilotos

KPI	Meta
Nº de Locais Ativos	100 por cidade
Check-ins médios	50/dia nos primeiros 30 dias
Feedbacks coletados	≥ 70% dos check-ins com feedback
Eventos criados	≥ 2 por local/mês
Upgrade para Premium/Vis.	≥ 25% em 3 meses

17.6 Fluxo Cíclico (Cidade Ativa)

1. Usuários chegam → exploram mapa/feed.
2. Locais parceiros recebem check-ins → dados viram score vibracional.
3. IA Aurah gera insights → parceiros aplicam melhorias.
4. Eventos e benefícios atraem mais visitas → novo ciclo.
5. Cidade inteira aparece no Mapa de Frequência Global → gera efeito rede.

17.7 Observabilidade em Pilotos

Logs

- `event_type`: `pilot_location_onboarded`, `pilot_event_created`, `pilot_feedback_submitted`.

Métricas

- `pilot_city_active_users_total`
- `pilot_city_checkins_total`
- `pilot_city_feedback_rate`

Alertas

- cidade com < 30% dos locais ativos após 2 meses.
- churn de parceiros > 10% em piloto.

🔚 Fechamento da Camada 17

Os **cenários de uso em cidades piloto** demonstram como o FriendApp conecta estratégia de growth, dados vibracionais e experiência real. Cada local ativado é um nó de energia, cada check-in é um pulso e cada evento é um catalisador. Essa camada garante que a implantação inicial seja **estratégica, mensurável e replicável** em escala global.

📘 CAMADA 18 — FUNCIONALIDADES OCULTAS PARA PARCEIROS VISIONÁRIOS

> Objetivo: definir os recursos exclusivos e secretos disponíveis apenas para parceiros Tier Visionário, que oferecem vantagens estratégicas, insights especiais e experimentos beta. Esta camada diferencia o plano Visionário, criando valor único e alimentando a percepção de elite vibracional.

18.1 Princípios

1. **Exclusividade:** apenas Visionários têm acesso, aumentando percepção de status.
2. **Valor Oculto:** funções que não aparecem para tiers inferiores.
3. **Experimentação:** acesso a features em beta antes do público geral.
4. **Vantagem Estratégica:** insights e boosts invisíveis que elevam performance.

18.2 Funcionalidades Secretas

🔮 1. Aurah Advisor Avançado

- Sugestões preditivas personalizadas com impacto esperado.
- Insights comparativos regionais (benchmarking de frequência).
- Alertas vibracionais exclusivos (queda de energia iminente).

⚡ 2. Boost Oculto de Visibilidade

- Eventos de Visionários recebem **peso extra no algoritmo do feed**.
- Local aparece para usuários até fora da zona geográfica imediata (expansão orgânica).

🌀 3. Radar de Afinidade

- Visualização anônima de clusters de público mais compatíveis.
- Ex.: "Seu espaço atrai 60% de usuários com perfil criativo e 25% com perfil introspectivo."

🧪 4. Acesso Beta

- Funcionalidades experimentais liberadas antes dos demais tiers:
  - Check-in Espelhado (conexão entre dois locais).
  - Ritual Sensorial (efeitos especiais no app durante eventos).
  - Portal de Feedback Expandido (relatos longos com processamento semântico avançado).

🌟 5. Selos Especiais

- Selos exclusivos (Guardião da Frequência, Portal de Expansão).
- Exibidos apenas em Visionários, atraindo mais visitantes.

18.3 Modelagem de Dados

PostgreSQL

- `partner_visionary_features`

```
CREATE TABLE partner_visionary_features (
  partner_id UUID PRIMARY KEY REFERENCES partners(id),
  boost_multiplier NUMERIC(3,2) DEFAULT 1.2,
  beta_access BOOLEAN DEFAULT true,
```

```
  radar_enabled BOOLEAN DEFAULT true,
  created_at TIMESTAMPTZ DEFAULT now()
);
```

18.4 APIs

GET `/api/partners/visionary/features`

200

```
{
  "boost_multiplier":1.2,
  "beta_features":["checkin_espelhado","ritual_sensorial"],
  "radar_enabled":true,
  "exclusive_badges":["guardiao_frequencia","portal_expansao"]
}
```

18.5 Observabilidade

Logs

- `event_type`: `visionary_feature_used`, `beta_feature_enabled`.
- `actor`: partner_id.
- `payload`: tipo da feature usada.

Métricas

- `visionary_boosts_total`
- `radar_queries_total`
- `beta_features_usage_total`

Alertas

- uso anômalo de boosts > 10x/dia.
- falhas em APIs beta > 5% requests.

18.6 UX

- Menu oculto no painel → visível só em Visionários.
- Design diferenciado (aura dourada, selos especiais).
- Cards exclusivos de sugestões da Aurah Advisor.
- Gamificação: conquistas especiais só para Visionários.

🔚 Fechamento da Camada 18

As **funcionalidades ocultas** transformam o parceiro Visionário em **um agente privilegiado dentro do ecossistema**. São recursos que não apenas ampliam a performance comercial, mas reforçam o valor de status, exclusividade e inovação do plano Visionário, mantendo o FriendApp competitivo e diferenciado.

📘 CAMADA 19 — MÓDULO DE EDUCAÇÃO E CONSCIÊNCIA PARA PARCEIROS

> Objetivo: especificar o sistema de onboarding educativo, conteúdos vibracionais e microcursos que capacitam os locais parceiros a operar no ecossistema FriendApp de forma alinhada, consciente e evolutiva. Essa camada garante que cada parceiro não apenas use o sistema, mas entenda seu papel como guardião vibracional.

19.1 Princípios

1. **Educação como Valor:** o parceiro aprende enquanto opera.
2. **Gamificação de Aprendizado:** missões educativas se traduzem em pontos e selos.
3. **Curadoria Vibracional:** conteúdos são adaptados pela IA Aurah Kosmos conforme estado do local.
4. **Escalabilidade:** módulos em formato micro (2–5 min), replicáveis globalmente.

19.2 Fluxo Educativo

```
[Cadastro Concluído]
  → [Onboarding Educativo Inicial]
  → [Microcursos Disponíveis no Painel]
  → [Missões Educacionais Gamificadas]
  → [Avaliação IA sobre aplicação do conhecimento]
  → [Selo de Local Consciente desbloqueado]
```

19.3 Conteúdos Oferecidos

1. Onboarding Inicial (obrigatório)

- O que é a Economia da Consciência.
- Como funcionam check-ins e feedbacks.
- Importância de manter a aura do local.
- Papel do parceiro na malha vibracional.

2. Microcursos

- **Hospitalidade Vibracional:** como a recepção influencia energia.
- **Ambiência Sensorial:** luz, som e cheiro como variáveis vibracionais.
- **Eventos que Curam:** como criar experiências que elevam.
- **Feedback Energético:** interpretar avaliações como sinais de campo.
- **Ciclo de Harmonização:** transformar colapsos em oportunidades.

19.4 Modelagem de Dados

PostgreSQL

- `partner_courses`

```
CREATE TABLE partner_courses (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  title TEXT NOT NULL,
  description TEXT,
  duration_minutes INT,
  is_mandatory BOOLEAN DEFAULT false
```

```
);
```

- `partner_course_progress`

```
CREATE TABLE partner_course_progress (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  partner_id UUID NOT NULL REFERENCES partners(id),
  course_id UUID NOT NULL REFERENCES partner_courses(id),
  completed BOOLEAN DEFAULT false,
  completed_at TIMESTAMPTZ
);
```

19.5 APIs

GET `/api/partners/courses`

200

```
[
  { "id":"uuid1","title":"Hospitalidade Vibracional","mandatory":true },
  { "id":"uuid2","title":"Ambiência Sensorial","mandatory":false }
]
```

POST `/api/partners/courses/{id}/complete`

200

```
{ "status":"completed","points_awarded":20,"badge":"Local Consciente" }
```

19.6 Gamificação

- Completar curso = pontos + badges.
- Badge principal: **“Local Consciente”** → desbloqueado após concluir todos cursos obrigatórios.
- Missões Educativas: ex.: *“Responda 3 feedbacks com cordialidade”* → recompensa extra.

19.7 Observabilidade

Logs

- `event_type`: `course_started`, `course_completed`.
- `actor`: partner_id.
- `payload`: título do curso, duração, pontos.

Métricas

- `courses_completed_total`
- `avg_completion_time_minutes`
- `mandatory_completion_rate`

Alertas

- taxa de conclusão obrigatória < 60% em cidade piloto.

- parceiros ativos sem curso concluído > 30 dias.

19.8 UX

- Aba “Academia Vibracional” no painel.
- Cursos exibidos como **cards interativos** com vídeo + resumo.
- Barra de progresso com % concluído.
- Recompensas aparecem imediatamente após conclusão.

🔚 Fechamento da Camada 19

O **Módulo de Educação e Consciência** garante que o FriendApp não seja só uma ferramenta, mas também uma **escola vibracional**. Parceiros são guiados, treinados e reconhecidos como agentes de transformação. Isso aumenta engajamento, reduz churn e fortalece a integridade da rede.

📘 CAMADA 20 — LÓGICA DE MODERAÇÃO DE LOCAIS, EVENTOS E AÇÕES COMERCIAIS

> Objetivo: especificar como o sistema protege o ecossistema contra uso inadequado, incoerências vibracionais e abusos comerciais. Inclui IA de moderação, regras de negócio, fluxo de denúncias e governança híbrida (IA + revisão humana).

20.1 Princípios

1. **Proteção da Rede:** só locais autênticos e coerentes permanecem ativos.
2. **Justiça:** moderação clara, com explicações e direito a recurso.
3. **Automação Inteligente:** IA resolve 80% dos casos; 20% vão para revisão humana.
4. **Não-Punitivo:** colapso vibracional gera **missões de harmonização**, não apenas bloqueios.

20.2 Itens Moderados

- Perfil de Locais → descrição, imagens, intenção vibracional.
- Eventos → título, descrição, temática, frequência.
- Benefícios Comerciais → coerência com propósito do app (nada abusivo).
- Feedbacks → monitorados para detectar manipulação ou ataques coordenados.

20.3 Regras (SE → ENTÃO)

SE...	ENTÃO...	Módulo
Local em **Colapso ≥ 7 dias**	Ativar missão de harmonização	IA Aurah
Local com descrições ofensivas	Ocultar até revisão	Moderação
Evento com spam ou incoerente	Suspender evento	Eventos
Benefício suspeito (ex.: clickbait)	Bloquear promoção	Benefícios
Ataque coordenado de feedbacks	Quarentena + peso reduzido	Antifraude
Denúncia confirmada por usuários	Local → `UNDER_REVIEW`	Moderador humano

---

20.4 Fluxo de Denúncia

1. Usuário denuncia → `POST /api/partners/locations/{id}/report`.
2. Denúncia classificada: conteúdo, vibração ou fraude.
3. IA analisa histórico + contexto:
   - Alta confiança → ação imediata.
   - Baixa confiança → `status=UNDER_REVIEW`.
4. Moderador humano revisa casos críticos.
5. Parceiro recebe notificação no painel.

20.5 APIs

POST `/locations/{id}/report`

Body

```
{
  "reason":"energia incoerente",
  "comment":"O ambiente estava agressivo, diferente da proposta."
}
```

201

```
{ "status":"received","report_id":"uuid" }
```

GET `/locations/{id}/moderation-status`

200

```
{
  "status":"UNDER_REVIEW",
  "pending_actions":["enviar novas fotos","corrigir descrição"]
}
```

20.6 Antifraude na Moderação

- **Clusterização de feedbacks**: detecta padrões iguais vindos de usuários novos.
- **Peso dinâmico**: feedbacks suspeitos recebem peso reduzido até revisão.
- **Shadow-ban**: usuários reincidentes podem enviar feedbacks, mas eles não impactam score.

20.7 Governança de Algoritmo

- Algoritmo versionado (`algorithm_version`).
- Painel mostra explicação: *"Seu score caiu por feedbacks negativos relacionados à iluminação"*.
- Parceiro pode **recorrer** → canal de contestação.
- SLA de revisão: até 48h úteis.

20.8 Observabilidade

Logs

- `event_type`: `report_submitted`, `moderation_action_taken`, `fraud_flagged`.
- `actor`: user_id / partner_id / system.

Métricas

- `reports_total`
- `moderation_actions_total`
- `locations_under_review_total`
- `avg_review_time_hours`

Alertas

- explosão de denúncias (> 20/h em um local).
- % de locais em `UNDER_REVIEW` > 10% em uma cidade.
- falha na fila de moderação > 5 min.

20.9 UX no Painel (Parceiro)

- Banner de aviso: *"Seu local está sob revisão vibracional."*
- Lista de ações recomendadas: ex. "Adicione novas fotos do ambiente."
- Botão para **enviar contestação** com comentários.
- Status atualizado em tempo real no painel.

🔚 Fechamento da Camada 20

A moderação de locais, eventos e ações comerciais garante que o FriendApp mantenha sua integridade vibracional e comercial. O sistema não apenas bloqueia, mas **educa e orienta parceiros**, transformando incoerências em oportunidades de evolução, e protegendo usuários de experiências negativas.

📘 CAMADA 21 — PAINEL DE ANÁLISE VIBRACIONAL DOS LOCAIS (USUÁRIOS)

> Objetivo: definir como os usuários comuns visualizam e interagem com as leituras vibracionais de um local parceiro. Este painel conecta feedbacks anônimos, check-ins, score vibracional e insights da IA Aurah Kosmos, traduzindo dados complexos em experiência visual e sensorial acessível.

21.1 Princípios

1. **Clareza:** exibir informações de forma simples, sem jargão técnico.
2. **Anonimato:** feedbacks individuais nunca revelam identidade do usuário.
3. **Explicabilidade:** IA deve traduzir dados vibracionais em frases claras.
4. **Gamificação:** tornar o ato de consultar e interagir divertido e recompensador.

21.2 Fluxo do Usuário

```
[Mapa / Feed]
  → [Clica em Local]
  → [Abre Painel Vibracional]
  → [Visualiza Score, Aura, Emoções]
  → [Explora Benefícios/Eventos]
```

→ [Decide fazer Check-in ou Salvar como Favorito]

21.3 Estrutura do Painel

Blocos exibidos ao usuário:

1. **Aura Ativa** → animação em cores (colapso, transição, expansão, pico).
2. **Score Vibracional Atual** → 0–100 com selo do estado.
3. **Resumo da IA** → frase explicativa:

   "O espaço está em Expansão, com muitos relatos de acolhimento e inspiração."
4. **Curva de Energia (7 dias)** → gráfico de linha.
5. **Feedback Coletivo** → nuvem de palavras + emojis dominantes.
6. **Eventos Ativos** → cards clicáveis.
7. **Benefícios Ativos** → promoções e descontos disponíveis.

21.4 Modelagem de Dados

Firestore

- `public_panels/{location_id}`

```
{
  "score":82,
  "state":"EXPANSAO",
  "keywords":["leve","acolhedor"],
  "emojis":["✨","🌿"],
  "feedback_avg":4.3,
  "trend":[70,72,75,80,82],
  "last_updated":"2025-09-14T18:00:00Z"
}
```

PostgreSQL

- `score_snapshots` → histórico detalhado.
- `partner_feedbacks` → agregados (apenas dados não sensíveis).

21.5 APIs

GET `/api/public/locations/{id}/vibration`

200

```
{
  "score":82,
  "state":"EXPANSAO",
  "summary":"O espaço está em Expansão, com energia acolhedora.",
  "feedback_avg":4.3,
  "keywords":["leve","acolhedor"],
  "trend":[70,72,75,80,82],
  "events":[ { "id":"uuid","title":"Roda de Som" } ],
  "benefits":[ { "type":"desconto","desc":"10% cardápio" } ]
```

```
}
```

21.6 Observabilidade

Logs

- `event_type`: `panel_opened`, `trend_viewed`, `event_clicked`.
- `actor`: user_id (hash).
- `entity`: partner_location_id.

Métricas

- `panels_opened_total`
- `avg_time_on_panel`
- `event_click_rate`
- `benefit_redeem_rate`

Alertas

- latência GET `/vibration` > 600ms.
- divergência entre Firestore e PostgreSQL > 5%.

21.7 UX

- Aura animada no topo (cores conforme estado vibracional).
- Cards interativos com animações suaves.
- Estados vazios tratados:
  - Sem feedbacks → *"Seja o primeiro a compartilhar sua percepção ✨"*.
  - Sem eventos → *"Nenhum evento ativo agora, siga o local para novidades."*
  - Sem benefícios → *"Este espaço ainda não oferece benefícios ativos."*

🔚 Fechamento da Camada 21

O Painel de Análise Vibracional dos Locais transforma dados invisíveis em **experiência sensorial**. É aqui que o usuário sente confiança para visitar, interagir e se conectar, sabendo que está entrando em um espaço **alinhado e transparente** dentro do ecossistema FriendApp.

📘 CAMADA 22 — SISTEMA DE FEEDBACK ENERGÉTICO ANÔNIMO + INSIGHTS COLETIVOS

> Objetivo: especificar como o FriendApp coleta, processa e exibe feedbacks vibracionais dos usuários após check-ins. Esta camada garante anonimato por design, transforma avaliações em dados coletivos e alimenta a IA Aurah Kosmos para recalcular scores e gerar insights preditivos.

22.1 Princípios

1. **Anonimato Total**: nenhum feedback revela identidade do usuário.
2. **Curadoria Sensorial**: inputs não são textões, mas sensações guiadas.
3. **Coleta Massiva + Inteligência**: cada feedback é pequeno, mas juntos revelam padrões.
4. **Explicabilidade**: insights exibidos para parceiros e usuários em linguagem clara.

22.2 Fluxo do Feedback

```
[Check-in Energético]
  → [Convite para Feedback]
  → [Usuário seleciona: slider + emojis + palavras-chave + opcional comentário]
  → [Envio anônimo para Firestore]
  → [Pipeline antifraude + NLP]
  → [Recalcular Score Vibracional]
  → [Gerar Insights Coletivos]
  → [Exibir nuvem de palavras + curva de emoções no painel/usuários]
```

22.3 Inputs Coletados

- **Slider Vibracional**: 0–5.
- **Emojis Energéticos**: ex.: ✨ 🌿 🔥 🌀.
- **Palavras-Chave**: sugeridas pela IA (*acolhedor, caótico, leve*).
- **Comentário Livre**: até 200 chars, opcional.

22.4 Modelagem de Dados

Firestore

Coleção: `feedbacks/{feedback_id}`

```
{
  "checkin_id":"uuid",
  "location_id":"uuid",
  "score":4,
  "emojis":["✨","🌿"],
  "keywords":["leve","acolhedor"],
  "comment":"Ambiente inspirador",
  "created_at":"2025-09-14T21:00:00Z",
  "quarantined":false}
```

PostgreSQL (consolidado)

Tabela: `partner_feedbacks` (ver Camada 8).

22.5 Antifraude

- **Peso por VTS**: usuários confiáveis têm peso maior.
- **Quarentena**: feedbacks suspeitos (ex.: explosão negativa coordenada).
- **Shadow-ban**: reincidentes não impactam score.
- **Clusterização Semântica**: detecta repetição massiva de termos.

22.6 Processamento e Insights

1. Normalizar dados.
2. Aplicar antifraude.
3. Rodar NLP em keywords + comentários.

4. Gerar **sentiment_score (-1 a +1)**.
5. Atualizar score vibracional.
6. Agregar para exibição coletiva.

22.7 APIs

POST `/api/partners/checkins/{id}/feedback`

Body

```
{
  "vibration_feedback":4,
  "emojis":["✨","🌿"],
  "keywords":["leve","acolhedor"],
  "comment":"Ambiente inspirador"
}
```

201

```
{ "status":"saved","impact_on_score":true }
```

GET `/api/partners/locations/{id}/feedback-summary`

200

```
{
  "avg_score":4.3,
  "keywords":["leve","acolhedor","criativo"],
  "emojis":["✨","🌿","😊"],
  "sentiment_trend":[0.7,0.8,0.6],
  "feedback_count":250
}
```

22.8 Observabilidade

Logs

- `event_type`: `feedback_submitted`, `feedback_quarantined`.
- `actor`: user_id (hash).
- `entity`: partner_location_id.

Métricas

- `feedbacks_total`
- `feedbacks_avg_score`
- `feedbacks_quarantined_total`

Alertas

- explosão de feedbacks (> 50 em 10 min).
- 20% feedbacks quarentenados.

22.9 UX

- Feedback enviado → animação de partículas douradas unindo-se ao campo.
- Usuário recebe mensagem: *"Sua ressonância foi registrada ✨"*.
- Parceiro vê no painel apenas agregados (palavras, emojis, médias).

🔚 Fechamento da Camada 22

O **Sistema de Feedback Energético Anônimo** garante anonimato e coleta massiva, transforma percepções individuais em **inteligência coletiva** e gera insights explicáveis. É um dos pilares da **Economia da Consciência**, equilibrando confiança do usuário, valor para o parceiro e justiça vibracional.

📘 CAMADA 23 — SISTEMA DE RECOMPENSAS E GAMIFICAÇÃO PARA PARCEIROS

> Objetivo: definir o mecanismo de gamificação B2B, onde parceiros acumulam pontos, desbloqueiam selos e recebem benefícios ao interagir de forma positiva com o ecossistema. Esta camada cria engajamento recorrente, reduz churn e incentiva boas práticas vibracionais.

23.1 Princípios

1. **Engajamento Contínuo:** cada ação gera pontos e progresso.
2. **Mérito Vibracional:** qualidade > quantidade (feedbacks positivos, missões de harmonização).
3. **Transparência:** regras claras e painel mostrando evolução.
4. **Recompensas Visíveis:** badges, boosts, acesso antecipado.

23.2 Pontuação de Ações

Ação	Pontos	Observações
Check-in positivo recebido	+3	score ≥ 4
Evento criado e realizado	+10	evento concluído sem fraudes
Feedback respondido cordialmente	+2	NLP verifica tom positivo
Missão de Harmonização concluída	+20	local saiu de Colapso
Adição de fotos/atualização de perfil	+1	até 5x/mês
Sugestão da Aurah Advisor aplicada	+5	aceita e executada

23.3 Níveis de Recompensa

Nível	Nome	Requisitos	Benefícios
1	**Aliado**	0–49 pts	Badge "Aliado Vibracional"
2	**Guardião**	50–149 pts	Destaque no feed + 1 boost grátis
3	**Mestre da Ressonância**	150–299 pts	Acesso antecipado a betas
4	**Oráculo**	300+ pts	Badge dourado + boosts extras mensais

23.4 Modelagem de Dados

PostgreSQL

- `partner_rewards`

```
CREATE TABLE partner_rewards (
  partner_id UUID PRIMARY KEY REFERENCES partners(id),
  points INT DEFAULT 0,
  level INT DEFAULT 1,
  badges TEXT[],
  updated_at TIMESTAMPTZ DEFAULT now()
);
```

- `partner_actions_log`

```
CREATE TABLE partner_actions_log (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  partner_id UUID NOT NULL,
  action TEXT NOT NULL,
  points_awarded INT NOT NULL,
  created_at TIMESTAMPTZ DEFAULT now()
);
```

23.5 APIs

GET `/api/partners/rewards`

200

```
{
  "points":120,
  "level":2,
  "badges":["Guardião da Frequência"],
  "next_level_in":30
}
```

POST `/api/partners/rewards/apply-action`

Body

```
{
  "action":"feedback_responded",
  "points":2
}
```

200

```
{ "status":"ok","new_total":122,"level_up":false }
```

23.6 Observabilidade

Logs

- `event_type`: `reward_points_awarded`, `level_up`, `badge_unlocked`.
- `actor`: partner_id.

- `payload`: ação, pontos, total.

Métricas

- `rewards_points_total`
- `level_distribution{1..4}`
- `badges_unlocked_total`

Alertas

- parceiros sem progresso > 30 dias.
- 10% ações revertidas por fraude.

23.7 UX no Painel

- Aba “Recompensas”: exibe barra de progresso.
- Animações ao ganhar pontos/badges.
- Notificação: *“Você subiu para Guardião da Frequência ✨”*.
- Histórico de ações → transparente.

🔚 Fechamento da Camada 23

O **Sistema de Recompensas e Gamificação** transforma parceiros em jogadores ativos da malha vibracional. Cada ação vira um passo em direção a status, benefícios e reconhecimento. Isso mantém engajamento constante, reduz churn e reforça a **cultura de mérito vibracional** do FriendApp.

📘 CAMADA 24 — SISTEMA DE HARMONIZAÇÃO PARA ESPAÇOS EM COLAPSO

> Objetivo: definir o protocolo automático e gamificado para locais em estado de Colapso. Esta camada garante que, em vez de punição, o sistema ofereça missões de recuperação vibracional, criando um ciclo de regeneração consciente e engajando parceiros e usuários no processo de cura do espaço.

24.1 Princípios

1. **Não-punitivo:** colapso não é banimento, mas convite à regeneração.
2. **Gamificado:** recuperação vira missão com recompensas.
3. **Coletivo:** usuários também podem apoiar o processo.
4. **Auditável:** logs e métricas registram todo o processo.

24.2 Gatilhos para Ativação

- `frequency_score < 40` por **≥ 7 dias consecutivos**.
- Explosão de feedbacks negativos não justificada.
- Evento com impacto vibracional crítico (queda de score > 30 pts).

24.3 Fluxo da Harmonização

```
[IA detecta Colapso]
  → [Ativa Missão no Painel do Parceiro]
  → [Sugestões da Aurah Advisor para recuperação]
```

```
→ [Parceiro executa ações práticas]
→ [Usuários convidados a apoiar com check-ins positivos]
→ [IA recalcula score e atualiza estado]
→ [Se recuperação → recompensa + selo]
```

24.4 Missões de Harmonização

Exemplos de missões propostas ao parceiro:

Missão	Ação	Impacto
Renovar Visual	Adicionar 3 novas fotos	+5 pts score
Reconexão com Público	Responder 5 feedbacks recentes	+7 pts score
Ritual Vibracional	Criar evento temático de cura	+15 pts score
Engajamento Coletivo	Receber 30 check-ins positivos em 7 dias	+20 pts score

24.5 Participação do Usuário

- Usuários frequentes recebem **convite push**:

  "Ajude a elevar o Café Alma Zen: faça um check-in vibracional positivo ✨."

- Check-ins positivos durante missão têm peso dobrado.

24.6 Modelagem de Dados

PostgreSQL

- `partner_harmonizations`

```
CREATE TABLE partner_harmonizations (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  location_id UUID NOT NULL REFERENCES partner_locations(id),
  mission TEXT NOT NULL,
  status TEXT NOT NULL DEFAULT 'active',
  started_at TIMESTAMPTZ DEFAULT now(),
  completed_at TIMESTAMPTZ
);
```

24.7 APIs

GET `/api/partners/harmonization/{location_id}`

200

```
{
  "status":"active",
  "missions":[
    {"id":"uuid1","title":"Adicionar novas fotos","progress":0.5},
    {"id":"uuid2","title":"Responder feedbacks","progress":0.0}
  ],
  "time_remaining":"5d"
}
```

POST `/api/partners/harmonization/{id}/complete`

200

```
{ "status":"completed","points_awarded":20,"badge":"Espaço Restaurado" }
```

24.8 Recompensas

- **Boost Gratuito** (não cumulativo).
- **Selo Temporário:** *"Espaço Restaurado"* (7 dias no perfil).
- **Pontuação Extra:** +20 pts no sistema de gamificação (Camada 23).

24.9 Observabilidade

Logs

- `event_type`: `harmonization_started`, `mission_completed`, `harmonization_success`.
- `actor`: partner_id / system.

Métricas

- `locations_in_harmonization_total`
- `harmonization_success_rate`
- `avg_recovery_time_days`

Alertas

- local em colapso > 30 dias.
- taxa de insucesso > 40%.

24.10 UX no Painel

- Banner vermelho: *"Seu espaço entrou em Colapso. Missões de Harmonização foram ativadas."*
- Lista de missões com progresso visual.
- Recompensas visíveis (boost, selo).
- Mensagem positiva da IA:

  "Colapso é oportunidade de renascimento. Complete as missões e veja sua frequência florescer novamente 🌱."

🔚 Fechamento da Camada 24

O **Sistema de Harmonização** transforma momentos de queda em **jornadas de cura e engajamento**. Parceiros são incentivados a agir, usuários a apoiar, e a IA acompanha todo o ciclo. É o equilíbrio entre **disciplina, regeneração e gamificação vibracional**.

📘 CAMADA 25 — CAMPO DE SUGESTÕES DA AURAH KOSMOS PARA PARCEIROS

> Objetivo: detalhar o módulo de sugestões proativas da IA Aurah Kosmos voltado a parceiros. Esse campo é o canal direto entre a inteligência vibracional do FriendApp e os locais, oferecendo aconselhamento contextual, acionável e explicável.

25.1 Princípios

1. **Proatividade:** IA não espera consulta; sugere em tempo real.
2. **Data-driven:** recomendações baseadas em métricas e feedbacks reais.
3. **Explicabilidade:** cada sugestão vem com os fatores que a geraram.
4. **Gamificação:** sugestões aceitas contam pontos no sistema de recompensas (Camada 23).

25.2 Fontes de Dados

- Feedbacks vibracionais (médias, keywords, sentimento).
- Check-ins recentes (picos/quedas de fluxo).
- Eventos passados (impacto no score pós-evento).
- Score vibracional histórico (tendências).
- Benchmarking (comparação com locais similares).
- Perfis de público predominante (Aurah clustering).

25.3 Tipos de Sugestões

Tipo	Exemplo
Ambiente	“Feedbacks apontam iluminação fraca. Sugestão: fotos novas com luz mais clara.”
Evento	“70% do seu público recente é criativo. Sugestão: evento de arte coletiva.”
Benefícios	“Usuários estão buscando descontos em sucos naturais. Sugestão: 10% off.”
Harmonização	“Sua frequência caiu para Transição. Missão: adicione 2 novas fotos + responda feedbacks.”
Comparativo	“Seu score está 15% abaixo da média de locais similares no bairro.”

25.4 APIs

GET `/api/partners/advisor/suggestions`

200

```
{
  "location_id":"uuid-local",
  "suggestions":[
    {
      "id":"sug-001",
      "type":"event",
      "message":"Crie um evento de meditação sonora",
      "confidence":0.84,
      "expected_impact":"+12% check-ins",
      "factors":["perfil do público","queda de score"]
    }
  ]
}
```

POST `/api/partners/advisor/accept`

Body

```
{ "suggestion_id":"sug-001" }
```

200

```
{ "status":"applied","points_awarded":5 }
```

25.5 Modelagem de Dados

PostgreSQL

- `advisor_suggestions`

```
CREATE TABLE advisor_suggestions (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  location_id UUID NOT NULL REFERENCES partner_locations(id),
  type TEXT NOT NULL,
  message TEXT NOT NULL,
  confidence NUMERIC(4,3),
  expected_impact TEXT,
  factors TEXT[],
  created_at TIMESTAMPTZ DEFAULT now()
);
```

- `advisor_actions`

```
CREATE TABLE advisor_actions (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  suggestion_id UUID REFERENCES advisor_suggestions(id),
  partner_id UUID,
  status TEXT CHECK (status IN ('accepted','rejected')),
  acted_at TIMESTAMPTZ DEFAULT now()
);
```

25.6 Observabilidade

Logs

- `event_type`: `advisor_suggestion_generated`, `advisor_suggestion_accepted`.
- `actor`: partner_id.
- `payload`: tipo, impacto esperado, fatores.

Métricas

- `advisor_suggestions_total`
- `advisor_acceptance_rate`
- `avg_time_to_accept_suggestion`

Alertas

- taxa de aceitação < 15% → sugestões pouco úteis.
- tempo médio para gerar sugestão > 5s.

25.7 UX no Painel

- Sugestões exibidas em cards com:

- Mensagem clara.
- Confiança (%).
- Impacto esperado.
- Lista dos fatores que motivaram.

- Botões: **Aplicar Agora** ou **Rejeitar**.
- Histórico de sugestões aceitas/rejeitadas.
- Badge extra quando uma sugestão aplicada resulta em aumento de score.

🔚 Fechamento da Camada 25

O **Campo de Sugestões da Aurah Kosmos** é o **GPS vibracional dos parceiros**, conectando inteligência coletiva em tempo real a ações práticas. Com ele, cada local evolui continuamente, transformando dados brutos em crescimento vibracional e comercial, sempre guiado pela IA guardiã do FriendApp.