

➡ MANUAL TÉCNICO DEFINITIVO — TESTE DE PERSONALIDADE ENERGÉTICA (FRIENDAPP)

✅ CAMADA 01 — ARQUITETURA GERAL DO SISTEMA E OBJETIVO PRIMÁRIO

🎯 Objetivo da Camada

Definir o núcleo da arquitetura técnica e vibracional do **Teste de Personalidade Energética**, que funciona como **módulo central de leitura energética** do FriendApp.

Ele coleta estímulos sensoriais e emocionais, processa via IA híbrida (Neural + Semântica + Vibracional), gera vetores energéticos e distribui os resultados para todo o ecossistema do app.

🧩 Componentes Arquiteturais

Camada	Função	Tecnologias
Frontend Sensorial	Coleta de respostas visuais, auditivas e intuitivas	Flutter + WebGL + FriendFX
Backend Core	Orquestra lógica do teste, APIs e microserviços	Node.js + Go + Python
IA Híbrida	Interpretação de padrões energéticos	TensorFlow, PyTorch, NLP (BERT/Spacy), Algoritmos proprietários
Banco de Dados	Persistência, histórico e vetores energéticos	PostgreSQL + Firestore + Redis
Fila Assíncrona	Processamento não bloqueante	Kafka ou Redis Streams
Infraestrutura	Escalabilidade e resiliência	Kubernetes + Istio + Multi-cloud (AWS, GCP, Azure)

📡 Fluxo Alto Nível

flowchart TD

```
A[Entrada Sensorial: imagens, sons, frases] → B[Backend API / Test Service]
B → C[Armazenamento Temporário Redis]
B → D[Fila de Mensagens Kafka]
D → E[Microserviço de Processamento IA]
E → F[PostgreSQL / Firestore (persistência)]
E → G[IA Aurah Kosmos]
```

G → H[Feed / Jogo / Conexões / Mapa de Frequência]

Segurança

- Sessão única por teste (`token_teste` , expira em 30min).
- Dados trafegam via TLS 1.3.
- Resultados salvos com criptografia AES-256.
- Proteção contra manipulação → IA de consistência valida padrões de tempo e respostas incoerentes.

Resultados Gerados (Outputs principais)

Saída	Formato	Destino
Perfil Dominante	Label ID + JSON	IA Aurah + Feed
Perfil Secundário	Label ID + JSON	Matching
Frequência Atual (Hz)	Float	Mapa de Frequência
Tendência Energética	Enum (ascendente, descendente, colapso, regeneração)	Jogo da Transmutação
UUID de Ressonância	Hash único	Logs e IA

Fechamento da Camada

Essa arquitetura garante que o **Teste de Personalidade Energética** seja **modular, escalável e seguro**, permitindo leituras profundas e ao mesmo tempo entregas rápidas para UX.

Ele serve como **alicerce de personalização de toda a experiência FriendApp**.

CAMADA 02 — FLUXO COMPLETO DO USUÁRIO (UX + BACKEND + DECISÕES)

Objetivo da Camada

Descrever, ponta a ponta, **como o usuário percorre o Teste de Personalidade Energética**, o que o **frontend** faz em cada etapa, **quais APIs** são chamadas, **quais eventos assíncronos** são disparados, e **quais decisões** o sistema toma até exibir o resultado e ativar o ecossistema (Feed, Jogo, Conexões, Mapa).

1) Estados de UX (frontend)

Estado	Descrição	Ações do App
<code>consent_gate</code>	Termos de leitura + privacidade	Exibir 3 checkboxes; habilitar “Começar”

Estado	Descrição	Ações do App
<code>session_start</code>	Criação/validação de sessão de teste	Chamar <code>POST /teste/sessao</code>
<code>questioning</code>	Perguntas sensoriais (1..N)	Render modular (imagem/som/frase); registrar tempo
<code>cooldown_breath</code>	Microtransição/respiração	Exibe animação leve 3–5s
<code>submit_end</code>	Envio de finalização	<code>POST /teste/finalizar</code> → resposta 202
<code>processing_info</code>	Espera elegante	Tela “sua frequência está sendo sintonizada...”
<code>result_ready</code>	Resultado pronto	Chamar <code>GET /teste/resultado/:test_id</code>
<code>activation</code>	Ativa jornada	Disparar triggers locais (mapa, feed, jogo)

UX Nota: Botão “voltar” desabilitado durante as perguntas. Indicação de progresso (p.ex. 8/12) sempre visível.

2) Sequência Técnica (end-to-end)

```
sequenceDiagram
```

```
autonumber
```

```
participant U as Usuária (App)
```

```
participant API as Backend API
```

```
participant Q as Queue (Kafka/Redis)
```

```
participant P as Worker Teste-Processor
```

```
participant DB as DB (Postgres/Firestore)
```

```
participant IA as IA Aurah
```

```
U->>API: POST /teste/sessao (consentimentos, deviceId)
```

```
API->>DB: cria sessão (status=started, token_teste)
```

```
API->>U: 201 {test_id, token_teste}
```

```
loop Para cada pergunta
```

```
U->>API: POST /teste/resposta {test_id, pergunta_id, choice_id, tempo_ms}
```

```
API->>DB: persiste resposta (stream/flush)
```

```
API->>U: 200 {ok, proxima_pergunta_id}
```

```
end
```

```
U->>API: POST /teste/finalizar {test_id}
```

```
API->>DB: marca status=submitted
```

```
API->>Q: publish event teste_finalizado {test_id}
```

```
API->>U: 202 Accepted (processing)
```

```
P->>DB: fetch respostas + metadados
```

```
P->>P: compõe vetores via Matriz de Ponderação (Camada V2 dedicada)
```

```
P->>IA: gerar frase-código + ajustes
```

```
IA->>P: perfil + score + flags (sombra/transição)
```

P→>DB: grava score_vector + perfil + status=completed
P→>U: (push/notify) resultado pronto

U→>API: GET /teste/resultado/:test_id
API→>DB: lê resultado + desbloqueios
API→>U: 200 resultado + triggers

3) Contratos de API (resumo operacional)

3.1 **POST /teste/sessao**

Body

```
{
  "consent_leitura": true,
  "consent_privacidade": true,
  "consent_emocional": true,
  "device_id": "XYZ",
  "app_version": "1.0.0"
}
```

Response 201

```
{
  "test_id": "T-9841",
  "token_teste": "jwt.session",
  "expires_in": 1800
}
```

3.2 **POST /teste/resposta**

Body

```
{
  "test_id": "T-9841",
  "pergunta_id": "Q-07",
  "choice_id": "IMG-014-AZURE",
  "tempo_ms": 3200
}
```

Response 200

```
{ "ok": true, "proxima_pergunta": "Q-08" }
```

3.3 **POST /teste/finalizar** → **assíncrono**

Body

```
{ "test_id": "T-9841" }
```

Response 202

```
{ "status": "processing", "message": "Sua frequência está sendo sintonizada." }
```

3.4 **GET /teste/resultado/:test_id**

Response 200

```
{
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Analítico Introspectivo",
  "frequencia_hz": 9.28,
  "tendencia": "expansão_emocional",
  "score_vector": {
    "foco": 0.74,
    "visao": 0.68,
    "presenca": 0.82,
    "forca": 0.61,
    "fluxo": 0.47,
    "sensibilidade": 0.59,
    "sombra": 0.37
  },
  "flags": {
    "sombra_ativa": false,
    "transicao": false
  },
  "ativacoes": {
    "feed": true,
    "jogo": true,
    "mapa": true,
    "conexoes": true
  }
}
```

4) Lógica de Decisão (backend)

4.1 Finalização (assíncrona, no worker)

```
on teste_finalizado(test_id):
    respostas = repo.getRespostas(test_id)
    matriz = repo.getMatrizPonderacao(version="v2")
    vetor = calcular_vetor(respostas, matriz) # soma ponderada por pergunta.weight
    perfil, conf, flags = IA.atribuirPerfil(vetor) # usa Camada 16 + heurísticas

    if flags.transicao == true:
        ativacoes = modoTransicao() # feed exploratório + trilha autodescoberta
    else:
        ativacoes = ativarPadrao(perfil, vetor) # feed/jogo/mapa/conexões

    repo.saveResultado(test_id, vetor, perfil, flags, ativacoes)
    push.notify(user, "Seu resultado está pronto")
```

4.2 Modo Transição (sem perfil dominante)

- Feed → curadoria **exploratória**, não-arquetípica
- Jogo → trilha **Autodescoberta e Clareza**
- Conexões → perfis de **alta plasticidade**
- Reavaliação → **micro-calibração** (2–3 perguntas) após 24–72h

5) Tratamento de Erros e Retentativas

Situação	Resposta/Comportamento
Perda de conexão ao enviar resposta	Retry exponencial (3 tentativas) + cache local
<code>POST /teste/finalizar</code> falha	Retorna 500; frontend exibe fallback e sugere reenviar; worker é idempotente
Worker caiu durante processamento	Mensagem permanece na fila (ack only on success)
Resultado não pronto em 60s	Frontend mantém tela elegante + oferece “notificar quando pronto”
Dados inconsistentes (anti-fraude)	Marcar sessão como suspeita; exigir reexecução guiada

6) Telemetria e Observabilidade

- **Eventos front:** `view_consent`, `start_test`, `answer_submitted`, `finalize_clicked`, `result_viewed`
- **Métricas back:** latência por endpoint; tempo de processamento do worker; taxas de erro

- **Tracing distribuído:** OpenTelemetry (API ↔ worker ↔ IA ↔ DB)
 - **Alertas:** filas acima de N msgs; retries > limiar; tempo médio de processamento > SLO
-

7) Performance e UX

- `POST /finalizar` sempre **202** → sem travar o usuário
 - Tela “sintonizando” com microanimação e 2 CTAs:
 - “Continuar explorando o app”
 - “Me avise quando estiver pronto” (push)
 - Pré-carregar assets das telas de resultado (Lottie/SVG) para entrada suave
-

8) Segurança e Privacidade

- `token_teste` com escopo **somente** para endpoints do teste
 - Criptografia em repouso (AES-256) e em trânsito (TLS 1.3)
 - Campos sensíveis (vetores, sombra) em JSONB **com GIN index e masking** nos logs
 - **Consentimentos** registrados com `timestamp`, IP e `hash_vibe`
-

9) Aceite (QA Checklist)

- ☐ `POST /finalizar` responde 202 e publica evento na fila
 - ☐ Worker processa e persiste `score_vector` + `perfil` + `ativacoes`
 - ☐ `GET /resultado/:test_id` retorna 200 com dados completos
 - ☐ Fluxo transição ativado quando não há perfil dominante
 - ☐ Retentativas robustas nas respostas e finalização
 - ☐ Telemetria (front/back) visível no dashboard
 - ☐ Segurança: tokens de sessão, consentimentos salvos, logs sem dados brutos
-

10) Pseudocódigo de Frontend (resumido)

```
async function startTest() {
  await api.post('/teste/sessao', consents)
  while (hasNextQuestion) {
    const choice = await renderQuestionAndCollectChoice()
    await api.post('/teste/resposta', { test_id, ...choice })
  }
  await api.post('/teste/finalizar', { test_id }) // 202
  showTuningScreen()
  await waitForPushOrPoll(() => api.get(`/teste/resultado/${test_id}`))
  renderResultAndActivations()
```

```
}
```

← **Fechamento da Camada**

Este fluxo garante **UX suave (sem travar)**, **backend escalável (event-driven)** e **decisão clara** para todos os casos (perfil dominante, transição, sombra).

Está pronto para implementação **sem ambiguidade**.

✅ **CAMADA 03 — MATRIZ DE PONDERAÇÃO ENERGÉTICA (DEFINIÇÃO, VERSIONAMENTO E EXEMPLOS)**

🎯 **Objetivo da Camada**

Definir com **clareza matemática e técnica** como cada escolha do usuário (imagem, som, frase, decisão rápida) se traduz em impactos nos **vetores energéticos**.

Essa matriz é a base do cálculo do **perfil dominante** e garante que os devs saibam exatamente **como implementar a lógica de atribuição**.

1) Estrutura da Matriz

A matriz é um **arquivo JSON versionado** que contém todos os **choice_id** do teste, cada um com o seu impacto vetorial.

Estrutura JSON

```
{
  "choice_id": "IMG-014-AZURE",
  "vector_impact": {
    "foco": -0.1,
    "fluxo": 0.2,
    "sensibilidade": 0.15,
    "sombra": 0.0
  },
  "weight": 1.0,
  "metadata": {
    "categoria": "visual",
    "descricao": "Mar azul profundo"
  }
}
```


2) Vetores Considerados

Vetor	Significado
foco	Clareza mental, direção
fluxo	Capacidade de soltar e seguir o curso
sensibilidade	Receptividade emocional
força	Determinação, ação
presenca	Estabilidade, ancoragem
visao	Capacidade de perceber além
sombra	Resistência ou bloqueio inconsciente

3) Exemplos Práticos

3.1 Escolha Visual

```
{
  "choice_id": "IMG-014-AZURE",
  "vector_impact": { "foco": -0.1, "fluxo": 0.2, "sensibilidade": 0.15, "sombra": 0.0 },
  "weight": 1.0
}
```

👉 Representa calma, aumenta fluxo e sensibilidade, reduz foco.

3.2 Escolha Sonora

```
{
  "choice_id": "FREQ-528-PULSE",
  "vector_impact": { "força": 0.2, "foco": 0.1, "fluxo": -0.05, "sombra": 0.0 },
  "weight": 1.2
}
```

👉 Frequência 528Hz é de cura: aumenta força e foco, mas reduz um pouco o fluxo.

3.3 Escolha Frasal

```
{
  "choice_id": "FRASE_CORAGEM",
  "vector_impact": { "força": 0.3, "sombra": -0.1 },
  "weight": 1.0
}
```

👉 Frase de coragem fortalece a ação e reduz levemente a sombra.

4) Lógica de Cálculo do Vetor Final

```
para cada resposta do usuário:  
  vetor_total += resposta.vector_impact * resposta.weight  
  
vetor_final = normalizar(vetor_total)
```

- Cada resposta soma seus impactos.
- O `weight` da pergunta multiplica o impacto.
- Resultado final é normalizado (0.0 a 1.0) para cada vetor.

5) Versionamento da Matriz

- A matriz é mantida em repositório Git como `matriz_vibracional_v2.json`.
- Cada mudança gera versão (`v2.1`, `v2.2` ...), garantindo rastreabilidade.
- O worker de IA sempre usa a **última versão estável**.

6) Fallback e Calibração

- Se uma escolha não tiver `vector_impact` definido → fallback = vetor neutro (`0.0` em todos os eixos).
- Calibração contínua: IA coleta resultados reais, compara com feedback dos usuários e ajusta pesos em releases futuras.



Fechamento da Camada

A **Matriz de Ponderação Energética** é a base matemática do Teste.

Com ela, cada escolha é convertida em impacto vetorial explícito → garantindo que a atribuição de perfil seja **implementável, auditável e calibrável**.



CAMADA 04 — SISTEMA DE COLETA SENSORIAL (IMAGENS, SONS E ESTÍMULOS)



Objetivo da Camada

Definir como o sistema apresenta **estímulos visuais, auditivos e textuais** ao usuário durante o teste, registrando **escolha, tempo de resposta e coerência**.

Essa coleta é a **fonte bruta de dados** para os cálculos da matriz de ponderação energética.

1) Estrutura Modular de Coleta

Módulo	Função	Tipos de Estímulo
VisualEngine	Renderiza imagens/arquetipos	Natureza, geometria, símbolos
AudioEngine	Toca sons/frequências	Binaurais, pulsos, sons naturais
FraseEngine	Apresenta frases curtas	Emoção, coragem, silêncio
TimingEngine	Mede tempo de reação	Impulsivo, reflexivo, hesitante
ConsistencyCheck	Valida coerência	Detecta repetições ou manipulação

2) Dados Coletados

Cada estímulo gera um registro no banco (tabela `energy_test_answers`):

```
{
  "user_id": "U123",
  "test_id": "T456",
  "step": "visual_2",
  "choice_id": "IMG-014-AZURE",
  "reaction_time_ms": 3200,
  "previous_choice": "FREQ-528-PULSE",
  "coherence_score": 0.82,
  "timestamp": "2025-09-04T20:15:23Z"
}
```

3) Exemplo de Estímulos

3.1 Imagem

- ID: `IMG-014-AZURE`
- Categoria: `visual`
- Descrição: "Mar azul profundo, horizonte aberto"

3.2 Som

- ID: `FREQ-528-PULSE`
- Categoria: `auditivo`
- Descrição: "Pulso binaural 528Hz com eco leve"

3.3 Frase

- ID: `FRASE_CORAGEM`
- Categoria: `frase`

- Texto: "O que mais preciso neste momento é coragem."

4) Lógica de Registro

```
onEscolha(user_id, test_id, choice_id, tempo_ms):  
    impacto = matriz.vector_impact(choice_id)  
    coerencia = calcular_coerencia(choice_id, previous_choice)  
    salvarResposta(user_id, test_id, choice_id, tempo_ms, coerencia, impacto)
```

5) Regras de Qualidade e Anti-Manipulação

- **Tempo mínimo:** respostas abaixo de 300ms → descartadas.
- **Repetição excessiva:** mais de 2x a mesma escolha → marcado como inconsistente.
- **Aleatoriedade total:** respostas incoerentes em sequência → IA ativa modo de validação.
- **Fallback:** se um estímulo falhar no frontend, apresenta outro do mesmo grupo.

6) Integração com a Matriz de Ponderação (Camada 03)

- Cada `choice_id` já tem seu **vector_impact** definido na matriz.
- O `TimingEngine` multiplica o impacto pela curva de tempo:
 - Rápido demais → peso reduzido.
 - Muito lento → marca hesitação.
 - Dentro do esperado → mantém peso normal.

← END Fechamento da Camada

O **Sistema de Coleta Sensorial** é a ponte entre a experiência do usuário e os cálculos da IA.

Ele garante que cada escolha (imagem, som, frase) seja registrada com precisão, validada contra manipulação e convertida em dados vetoriais de impacto energético.

✓ CAMADA 05 — PROCESSAMENTO INTELIGENTE (IA SEMÂNTICA + NEURAL VIBRACIONAL)

🎯 Objetivo da Camada

Traduzir as respostas do usuário em **vetores energéticos consolidados** e em um **perfil final**, usando uma arquitetura híbrida de IA.

Essa é a etapa em que os dados brutos coletados (imagens, sons, frases, tempos de resposta) são processados, interpretados e transformados em significado energético.

1) Arquitetura da IA

Módulo	Tipo	Função
Aurah.NLP	IA Semântica	Lê frases, palavras-chave, intenção emocional
Aurah.Cluster	Agrupador Inteligente	Classifica padrões em blocos vibracionais
Aurah.Resonance	Neural Vibracional	Calcula frequência, polaridade e sombra
Aurah.DecisionMap	Heurístico	Define perfil dominante, secundário e tendência

2) Pipeline de Processamento

flowchart TD

A[Respostas Brutas] → B[Aurah.NLP]

B → C[Aurah.Cluster]

C → D[Aurah.Resonance]

D → E[Aurah.DecisionMap]

E → F[Perfil + Frequência + Frase-Código]

F → G[IA Aurah Kosmos / Feed / Jogo / Conexões]

3) Aurah.NLP (Semântico)

- Extrai tokens emocionais: *"coragem"*, *"silêncio"*, *"liberdade"*
- Classifica em **categorias vibracionais**: expansão, introspecção, cura, sombra
- Exemplo de saída:

```
{
  "tokens": ["coragem"],
  "categoria": "forca",
  "impacto": { "forca": 0.3, "sombra": -0.1 }
}
```

4) Aurah.Cluster (Agrupamento)

- Usa **clustering não-linear (DBSCAN + lógica difusa)**
- Agrupa combinações de imagens, sons e frases em **arquétipos vibracionais**
- Exemplo:

```
{
  "cluster": "Guardião Solar",
  "score": 0.87
}
```

5) Aurah.Resonance (Neural Vibracional)

- CNN + algoritmos proprietários para mapear vetores em frequência (Hz)
- Gera polaridade e nível de sombra
- Exemplo:

```
{
  "frequencia": 9.28,
  "polaridade": "positiva",
  "sombra": 0.37
}
```

6) Aurah.DecisionMap (Decisão Final)

- Junta todos os resultados:
 - Vetores consolidados
 - Perfis com maior score
 - Tendência vibracional
- Decide **perfil dominante, secundário e frase-código**

Exemplo de saída final:

```
{
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Analítico Introspectivo",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "resonance_uuid": "a8f1c2d9-31e4"
}
```

7) Pseudocódigo do Worker

```

respostas = repo.getRespostas(test_id)
vetores = aplicarMatriz(respostas)

tokens = AurahNLP.analisar(respostas.frases)
clusters = AurahCluster.classificar(vetores, tokens)
resonancia = AurahResonance.calcular(vetores)

perfil, tendencia = AurahDecisionMap.definir(clusters, resonancia)

salvarResultado(user, vetores, perfil, tendencia)

```

8) Segurança e Transparência

- **Score mínimo de confiança:** 0.75 para perfil dominante
- **Audit trail:** salvar vetores + perfil atribuído + versão da matriz usada
- **Fallback:** se nenhum perfil > 0.75 → entra em *modo transição* (Camada 09)

← END Fechamento da Camada

Essa camada garante que o FriendApp não entregue apenas um “resultado simbólico”, mas sim uma **tradução matemática, lógica e auditável da energia do usuário**.

Aqui a IA converte dados brutos em **perfil dominante, secundário, frequência e frase-código**, alimentando todo o ecossistema.

✅ CAMADA 06 — RESULTADO FINAL (EXIBIÇÃO SENSORIAL + ENTREGA AO USUÁRIO + API DE RETORNO)

🎯 Objetivo da Camada

Transformar o processamento da IA em uma **experiência sensorial única para o usuário** e, ao mesmo tempo, disponibilizar o resultado em **formato técnico via API**, alimentando o ecossistema (Feed, Jogo, Conexões, Mapa).

1) Estrutura da Tela de Resultado

Elemento	Comportamento
Nome do Perfil	Animado em destaque (ex.: <i>Guardião Solar</i>)
Frase-Código	Texto vibracional gerado pela IA, ex.: <i>“Você é chama que aquece, mas precisa descansar”</i>




Elemento	Comportamento
Frequência Hz	Valor em destaque com visual fractal
Polaridade	Exibida em cor/ícone (ex.: ☀️ solar, 🌙 lunar, ⚖️ neutra)
Mapa Energético	Gráfico circular dinâmico dos vetores
Símbolo Visual	Ícone/Fractal animado relacionado ao arquétipo
CTA Botões	"Ativar minha Jornada" → desbloqueios no app

2) Feedback Emocional do Usuário

Após exibir o resultado, o app pergunta:

“Como você se sentiu com esse resultado?”

Opções:

-  Conectado(a)
-  Em dúvida
-  Não me representou

Uso dos feedbacks:

- **Conectado** → resultado confirmado.
- **Em dúvida** → IA agenda *micro-recalibração* em até 48h (2–3 perguntas extras).
- **Não representou** → sistema ativa *Modo Refinamento* → sugere repetir teste ou rodar fluxo de “transição”.

3) API de Retorno (GET Resultado)

Endpoint

GET /api/teste/resultado/:test_id

Response

```
{
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Analítico Introspectivo",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "score_vector": {
    "foco": 0.74,
    "visao": 0.68,
    "presenca": 0.82,
    "forca": 0.61,
    "fluxo": 0.47,
```



```
"sensibilidade": 0.59,  
"sombra": 0.37  
},  
"frase_codigo": "Você é a luz que guia e expande.",  
"flags": {  
  "sombra_ativa": false,  
  "transicao": false},  
"ativacoes": {  
  "feed": true,  
  "jogo": true,  
  "mapa": true,  
  "conexoes": true}  
}
```

4) Fluxo Técnico Pós-Entrega

```
ao_exibir_resultado(test_id):  
  render_visuals(perfil, frequencia, mapa)  
  perguntar_feedback()  
  if feedback == "duvida" or "nao":  
    IA.agendar_micro_calibracao(user_id)  
  triggers = repo.getAtivacoes(test_id)  
  ativar_sistemas(triggers)
```

5) Integrações Ativadas

- **Feed Sensorial** → recebe tags do perfil dominante.
- **Jogo da Transmutação** → inicia trilha inicial conforme tendência.
- **Mapa de Frequência** → plota coordenada vibracional.
- **Conexões** → IA sugere perfis compatíveis.
- **Aurah Kosmos** → registra assinatura para acompanhamento contínuo.

6) Experiência de UX

- **Animações suaves** → fractal se expandindo ao revelar o perfil.
- **Trilha sonora adaptativa** → ajustada à frequência do resultado.
- **Opção de compartilhamento** → imagem vibracional + frase-código (Premium).
- **Modo Transição** → caso não haja perfil dominante, mensagem especial:

“Você está em um momento de transição. Sua energia ainda está se reorganizando. Vamos caminhar juntos para clarear seu campo.”

← END Fechamento da Camada

Essa camada garante que o **resultado não seja apenas dado técnico**, mas uma **experiência sensorial completa**, ao mesmo tempo que expõe via API os dados necessários para alimentar todo o ecossistema FriendApp.

✓ CAMADA 07 — SISTEMA DE CONSENTIMENTO, ARMAZENAMENTO E PERSISTÊNCIA DOS RESULTADOS (CACHE + CRIPTOGRAFIA + REVOGAÇÃO)

🎯 Objetivo da Camada

Garantir que o Teste de Personalidade Energética seja **seguro, transparente e ético**, com foco em:

1. Consentimento explícito do usuário.
2. Armazenamento criptografado.
3. Persistência controlada (cache temporário x banco permanente).
4. Opção clara de revogação/exclusão de dados.

1) Consentimento Explícito

Antes de iniciar o teste, o usuário deve aceitar **3 checkboxes obrigatórios**:

- 📖 **Consentimento de Leitura** → “Aceito que a IA interprete minhas respostas e emoções durante este teste.”
- 🛡️ **Consentimento de Privacidade** → “Aceito que meus dados sejam criptografados e tratados de acordo com LGPD/GDPR.”
- 💎 **Consentimento Emocional** → “Estou pronto(a) para participar de um processo de autoconhecimento vibracional.”

Todos os consentimentos são armazenados com:

- `timestamp`
- `IP`
- `hash_vibe` (assinatura criptografada única do teste)

2) Estratégias de Armazenamento

Tipo	Condição	Tempo de Retenção	Criptografia
Cache Temporário	Usuário não confirma salvar	72h	AES-256
Banco Permanente	Usuário aceita salvar resultado	Indefinido (com opção de exclusão)	AES-256 + RSA
Backup Fragmentado	Segurança e redundância global	Até 30 dias	Fragmentação + múltiplas chaves

3) Estrutura Técnica de Persistência

Tabela: **energy_test_results**

Campo	Tipo	Proteção
user_id	UUID	FK + mascaramento
test_id	UUID	PK
score_vector	JSONB	AES-256 + índice GIN
perfil_dominante	String	AES-256
frequencia_hz	Float	AES-256
frase_codigo	String	AES-256
consentimentos	JSON	visível apenas em painel técnico
revogado	Boolean	default false

4) Revogação de Dados

O usuário pode excluir seu resultado a qualquer momento:

- **No App** → Configurações → Privacidade → “Excluir minha leitura vibracional”.
- **Processo:**
 - Marca **revogado = true** no banco.
 - Move os dados para área de *quarentena criptografada* (mantida só para logs de auditoria).
 - Notificação confirmando exclusão:

“Sua frequência foi liberada. Você pode refazer o teste quando desejar.”

5) Fallbacks Técnicos

- Se o usuário **não responder ao consentimento** → bloqueia acesso ao teste.
- Se o app estiver offline → respostas salvas localmente (storage criptografado) e enviadas quando online.
- Se houver erro no salvamento em banco permanente → mantém em cache até reprocesso.

6) Logs e Observabilidade

Cada ação gera log criptografado:

- Consentimento aceito ou recusado.
- Resultado salvo em cache ou banco.
- Exclusão ou revogação realizada.
- Falhas técnicas (ex: erro de sincronização).

Logs visíveis apenas para equipe autorizada de DevOps / Compliance.



Fechamento da Camada

Essa camada garante que o **FriendApp seja confiável** e respeite profundamente a jornada do usuário: nada é armazenado sem consentimento, tudo é criptografado, e o usuário tem sempre controle para excluir sua frequência.

✓ CAMADA 08 — ATIVAÇÃO DE FUNCIONALIDADES A PARTIR DO RESULTADO (FEED, JOGO, CONEXÕES, MAPA DE FREQUÊNCIA, RA, IA AURAH)



Objetivo da Camada

Garantir que, assim que o resultado do Teste de Personalidade Energética é gerado, todo o ecossistema do FriendApp seja **personalizado dinamicamente**.

Essa camada define **quem recebe o quê**, em qual momento, e com qual intensidade.

1) Funcionalidades Ativadas Imediatamente

Funcionalidade	Dados Recebidos	Ação
Feed Sensorial	Perfil dominante + vetores	Exibir conteúdos compatíveis com o arquétipo do usuário
Jogo da Transmutação	Frequência Hz + tendência	Definir trilha inicial e totem vibracional desbloqueado
Mapa de Frequência	Frequência Hz + polaridade	Posicionar usuário em cluster vibracional
Conexões Autênticas	Vetores compatíveis + sombra	Sugerir conexões energéticas complementares
Aurah Kosmos (IA)	Assinatura vibracional + feedback	Aprendizado contínuo e acompanhamento evolutivo
RA (Realidade Aumentada)	resonance_uuid + perfil	Exibir símbolos energéticos na câmera (Premium)

2) Pipeline de Ativação Técnica

flowchart TD

A[Resultado Teste] → B[Feed Sensorial]

A → C[Jogo da Transmutação]

A → D[Mapa de Frequência]

A → E[Conexões Autênticas]

A → F[IA Aurah Kosmos]

A → G[RA e Experiências Imersivas]

3) API de Ativação

Endpoint: `POST /teste/ativar-funcionalidades`

Body

```
{
  "user_id": "U123",
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Analítico Introspectivo",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "resonance_uuid": "abc-xyz-999"
}
```

Response

```
{
  "feed": "ativado",
  "jogo": "iniciado",
  "mapa": "atualizado",
  "conexoes": "reconstruidas",
  "aurah": "sincronizada",
  "ra": "habilitada"
}
```

4) Lógica de Encadeamento

```
onResultadoGerado(user_id, resultado):
  FeedSensorial.injetar(resultado.perfil_dominante, resultado.vetores)
  JogoTransmutacao.iniciar(resultado.frequencia, resultado.tendencia)
  MapaFrequencia.atualizar(user_id, resultado.frequencia)
```

```
Conexoes.rebuild(user_id, resultado.perfil, resultado.sombra)
AurahKosmos.sync(user_id, resultado)
if user.premium:
    RA.habilitar(user_id, resultado.resonance_uuid)
```

5) Regras Técnicas

- Feed atualizado em **tempo real**.
- Jogo só inicia após confirmação de resultado.
- Conexões são recriadas a cada teste, sobrescrevendo anteriores.
- RA só disponível para Premium.
- Logs de ativação são salvos na tabela `test_activations`.

6) Painel de Diagnóstico Técnico

Campos monitorados:

- Última frequência registrada.
- Perfil dominante atribuído.
- Funcionalidades ativadas (true/false).
- Tempo de propagação do resultado (meta: < 500ms).

Fechamento da Camada

Essa camada garante que o **Teste de Personalidade Energética não é isolado**, mas sim o **gatilho central** que personaliza o Feed, inicia o Jogo, posiciona no Mapa, sugere Conexões e sincroniza a IA Aurah Kosmos.

CAMADA 09 — REAVALIAÇÃO ENERGÉTICA, NOVO TESTE E HISTÓRICO DE FREQUÊNCIAS

Objetivo da Camada

Permitir que o usuário **refaça o Teste de Personalidade Energética** em ciclos de tempo definidos ou em momentos de transição, garantindo que sua evolução vibracional seja acompanhada ao longo da jornada.

Essa camada cria o **histórico de frequências** e define regras claras para **novo teste, microcalibrações e reavaliações automáticas**.

1) Regras de Reavaliação

Tipo	Gatilho	Frequência Permitida	Observação
Novo Teste Completo	Solicitação manual do usuário	1x a cada 60 dias	Gera novo perfil e substitui versão ativa
Micro-Calibração	Feedback "não me representou" ou inconsistência IA	Até 1x por semana	2–3 perguntas adicionais
Reavaliação Automática	Ciclo vibracional completo (90 dias)	Automática	Usuário notificado para novo teste
Revisão Emergencial	Evento detectado pela IA (colapso/expansão abrupta)	Imediata	IA sugere nova leitura

2) Estrutura do Histórico

Tabela: `energy_profile_history`

Campo	Tipo	Descrição
<code>history_id</code>	UUID	Identificador único
<code>user_id</code>	UUID	FK
<code>test_id</code>	UUID	FK
<code>perfil_dominante</code>	String	Nome do perfil
<code>frequencia_hz</code>	Float	Valor vibracional
<code>tendencia</code>	Enum	expansão, transição, colapso
<code>resonance_uuid</code>	String	Assinatura única
<code>created_at</code>	Timestamp	Data da leitura
<code>ativa</code>	Boolean	Se é o perfil atual

3) Fluxo de Reavaliação

```
ao_finalizar_teste(user_id, resultado):
    salvarResultado(resultado, ativa=true)
    desativarVersaoAnterior(user_id)
    registrarHistorico(user_id, resultado)

se user.feedback == "nao":
    agendarMicroCalibracao(user_id, prazo=48h)

se tempoDesdeUltimoTeste > 90 dias:
    notificar(user_id, "Está na hora de refazer sua leitura energética")
```

4) Experiência de Usuário (UX)

- **Tela de histórico** → gráfico de evolução vibracional (linha ou curva) mostrando picos, quedas e transições.
- **Notificação push** → “Sua energia mudou? Refaça seu teste para atualizar sua frequência.”
- **Badge de evolução** → usuários Premium recebem selos: “Recalibrado”, “Em Transição”, “Nova Frequência”.

5) Casos Especiais

- **Perfil em transição** → se nenhum arquétipo dominante for detectado, IA agenda reteste automático em 7 dias.
- **Mudança abrupta** → se a frequência cair ou subir > 20% em menos de 30 dias, IA recomenda reteste.
- **Feedback negativo** → ativa *Modo Refinamento* (micro-calibração de 2–3 perguntas).

Fechamento da Camada

A reavaliação garante que o **perfil energético seja dinâmico, vivo e fiel ao momento do usuário**.

Essa camada conecta o FriendApp à realidade: energias mudam, e o app acompanha essas mudanças com inteligência, ética e profundidade.

CAMADA 10 — ARQUITETURA DE IA: COLETA, TREINAMENTO E PERSONALIZAÇÃO DINÂMICA (AURAH KOSMOS)

Objetivo da Camada

Descrever como a **IA Aurah Kosmos** coleta dados durante o teste, processa em tempo real, aprende continuamente e personaliza a jornada vibracional de cada usuário.

Aqui, a IA não apenas interpreta, mas também **evolui junto com o usuário**, garantindo precisão e adaptação.

1) Níveis de Coleta da IA

Nível	Tipo de Dado	Finalidade
N1	Escolha literal (imagem, som, frase)	Mapeamento inicial de intenção
N2	Tempo de resposta (ms)	Detectar hesitação ou impulsividade
N3	Ordem das escolhas	Entender polaridade energética

Nível	Tipo de Dado	Finalidade
N4	Palavras/frases abertas	Interpretar significado e emoção
N5	Histórico vibracional	Ajustar perfil com base em ciclos

2) Pipeline de Processamento da IA

flowchart TD

A[Respostas do Usuário] → B[Aurah.Collector]

B → C[Aurah.Mapper]

C → D[Aurah.MindsetEngine]

D → E[Aurah.Resonance]

E → F[Aurah.DecisionMap]

F → G[Aurah Kosmos Core]

- **Aurah.Collector** → coleta respostas, tempos e eventos.
- **Aurah.Mapper** → converte em vetores usando a Matriz de Ponderação.
- **Aurah.MindsetEngine** → avalia tendências e padrões emocionais.
- **Aurah.Resonance** → calcula frequência, sombra e polaridade.
- **Aurah.DecisionMap** → define perfil dominante, secundário e tendência.

3) Estrutura de Vetorização

Cada escolha vira um vetor energético multidimensional:

```
{
  "choice_id": "IMG-014-AZURE",
  "impacto": {
    "foco": -0.1,
    "fluxo": 0.2,
    "sensibilidade": 0.15,
    "sombra": 0.0
  },
  "tempo_resposta": 3200,
  "peso_final": 1.0
}
```

4) Aprendizado Contínuo

- A IA salva os vetores anonimizados para **treinamento coletivo**.
- Reajusta pesos da matriz conforme feedback dos usuários.

- Detecta incoerências repetidas → ativa modo de calibração.
- Sugere **micro-calibrações** quando o usuário não se identifica com o resultado.

5) Personalização Dinâmica

Com base no perfil ativo, a IA Aurah Kosmos:

Sistema	Ajuste Personalizado
Feed Sensorial	Sugestões de posts compatíveis com energia
Jogo da Transmutação	Desafios calibrados para arquétipo dominante
Mapa de Frequência	Posicionamento em clusters corretos
Conexões	Matching por afinidade vibracional
RA	Símbolos visuais relacionados ao arquétipo

6) Regras Técnicas de IA

- **Confiança mínima:** perfil só é atribuído se `score > 0.75`.
- **Logs de decisão** → salvar versão da matriz usada, score, flags.
- **Fallback de transição** → se não atingir confiança mínima, usuário entra em *modo transição*.
- **Ética** → IA nunca força um resultado fixo, apenas interpreta.

7) Painel Técnico de IA

Campos monitorados para devs:

- Última versão da matriz usada.
- Score médio por perfil.
- Perfis mais comuns atribuídos.
- Taxa de usuários em transição.
- Tempo médio de resposta IA.



END Fechamento da Camada

Essa camada mostra que a IA Aurah Kosmos é o **cérebro vivo do Teste de Personalidade Energética**, transformando dados brutos em insights, adaptando o ecossistema e aprendendo continuamente.



CAMADA 11 — SISTEMA DE TIPAGEM ENERGÉTICA (CATEGORIAS, NOMES,

POLARIDADES, LEGADO E EXPANSÃO)

Objetivo da Camada

Definir o **sistema de tipagem energética** usado pelo FriendApp para classificar os usuários em arquétipos vibracionais, garantindo **clareza técnica** e **expansibilidade futura**.

Esse sistema é a base para os resultados do teste e para a personalização do app.

1) Estrutura da Tipagem

Cada perfil energético contém os seguintes elementos:

Campo	Descrição
Nome	Arquétipo que simboliza a energia (ex.: Guardiã Solar)
Slug	Identificador técnico único (ex.: <code>guardiao_solar</code>)
Frequência Base (Hz)	Média vibracional associada
Polaridade	Solar (ativa), Lunar (receptiva), Neutra (equilibrada)
Elemento Arquétipo	Conexão simbólica (fogo, água, ar, terra, éter)
Tendência Natural	Caminho predominante da energia
Sombra Associada	Bloqueio ou limitação mais comum
Legado Vibracional	Potencial de contribuição coletiva
Frase de Retorno	Mensagem gerada pela IA para o usuário

2) Exemplos de Perfis

Guardião Solar

```
{
  "nome": "Guardião Solar",
  "slug": "guardiao_solar",
  "frequencia_base": 9.28,
  "polaridade": "solar",
  "elemento": "fogo",
  "tendencia": "expansao_guiada",
  "sombra": "controle_excessivo",
  "legado": "despertar_coletivo",
  "frase_retorno": "Você é chama que aquece e guia, mas precisa também repousar."
}
```

Oráculo Lunar

```
{
  "nome": "Oráculo Lunar",
  "slug": "oraculo_lunar",
  "frequencia_base": 8.43,
  "polaridade": "lunar",
  "elemento": "água",
  "tendencia": "introspecao",
  "sombra": "evasao",
  "legado": "sabedoria_coletiva",
  "frase_retorno": "Você é silêncio que escuta e revela, mas deve evitar se esconder."
}
```

3) Regra de Atribuição

- O perfil dominante é aquele cujo **score vetorial** > **0.75**.
- Se houver empate entre dois perfis → usuário recebe **perfil híbrido** (dominante + secundário).
- Se nenhum atingir 0.75 → usuário entra em **modo transição** (Camada 09).

4) Integração com o Ecossistema

Sistema	Uso da Tipagem
Jogo da Transmutação	Define trilha inicial, missões e totems
Feed Sensorial	Seleção de temas e conteúdos compatíveis
Mapa de Frequência	Agrupamento em clusters por arquétipo
Conexões Autênticas	Compatibilidade energética entre usuários
RA (Realidade Aumentada)	Exibição de símbolos e fractais relacionados

5) Banco de Dados

Tabela: `energy_profiles`

Campo	Tipo
<code>id</code>	UUID
<code>nome</code>	String
<code>slug</code>	String (único)
<code>frequencia_base</code>	Float
<code>polaridade</code>	Enum (solar, lunar, neutra)
<code>elemento</code>	String
<code>tendencia</code>	String

Campo	Tipo
sombra	String
legado	String
frase_retorno	String

6) Expansibilidade

- O sistema deve suportar a criação de **novos perfis** em releases futuras.
- Cada perfil é tratado como **objeto de classe independente** para permitir evolução sem impactar os existentes.
- Arquitetura permite até **144 perfis ativos**, organizados em **famílias arquetípicas**.

← END Fechamento da Camada

O **Sistema de Tipagem Energética** é o núcleo simbólico e técnico do Teste.

Ele garante que cada resultado seja **claro, rastreável e expansível**, servindo de base para a personalização de toda a experiência no FriendApp.

✓ CAMADA 12 — CAMADA UX SENSORIAL (FLUXO, NAVEGAÇÃO, IMAGENS E MICROANIMAÇÕES DO TESTE)

🎯 Objetivo da Camada

Definir a **experiência sensorial do usuário** durante o Teste de Personalidade Energética, detalhando **fluxo, telas, estímulos, microanimações e transições**.

O objetivo é transformar o teste em uma **jornada imersiva**, mantendo consistência técnica para devs e clareza estética para designers.

1) Estrutura do Fluxo UX

Etapa	Comportamento	Elementos
Boas-vindas Vibracional	Tela introdutória suave com fractal animado + som 432Hz	CTA "Começar"
Consentimento	3 checkboxes obrigatórios	Privacidade, leitura, emocional
Início do Teste	Primeira escolha sensorial	Imagem/som/frase
Perguntas Sequenciais	Entre 10–12 rodadas	Render modular, timer invisível
Micro-Transição	Pausas com respiração guiada (3s)	Animação leve

Etapa	Comportamento	Elementos
Finalização	Envio assíncrono (<code>POST /finalizar</code>)	Tela "Sua frequência está sendo sintonizada"
Resultado	Exibição vibracional personalizada	Perfil + frequência + frase
Ativações	Feed, Jogo, Conexões	Botão "Ativar Minha Jornada"

2) Navegação

- **Progresso:** barra discreta (ex.: 4/12).
- **Botão voltar:** desabilitado (experiência linear).
- **Fallback offline:** respostas armazenadas em cache criptografado, enviadas quando online.
- **Tempo por pergunta:** invisível para o usuário, mas registrado para IA.

3) Estímulos Visuais

- Imagens carregadas via CDN + cache local.
- Categoria: natureza, fractais, símbolos, arquétipos.
- Cada imagem com **ID único** (mapeado na matriz).
- Microanimações:
 - Escolha → efeito de pulso energético.
 - Transição → fade suave + mudança de cor.

4) Estímulos Auditivos

- Sons binaurais (174Hz, 285Hz, 432Hz, 528Hz, 963Hz).
- Carregados em buffer de baixa latência.
- Sincronizados com transições visuais.
- Regras:
 - Usuário só pode ouvir uma vez por pergunta.
 - Se áudio falhar → fallback com frequência padrão 432Hz.

5) Microanimações e Feedback Tátil

- **Feedback tátil** (vibração leve) ao tocar na escolha.
- **Microanimações** baseadas na frequência:
 - Tons escuros → animações lentas e densas.
 - Tons claros → animações rápidas e expansivas.
- **Fractais dinâmicos** → mudam levemente a cada rodada para manter frescor.

6) Performance e Otimização

- Peso máximo de assets por pergunta: < **1MB**.
- Pré-carregamento das 2 próximas perguntas em background.
- Modo offline: fallback com assets mínimos locais.
- Tempo médio esperado do teste: **4 a 6 minutos**.

7) Pseudocódigo de Renderização UX

```
renderPergunta(pergunta):  
  mostrarImagem(pergunta.imagem_id)  
  tocarAudio(pergunta.audio_id)  
  mostrarFrase(pergunta.frase_id)  
  iniciarTimer()  
  
onEscolha(choice_id):  
  aplicarAnimacaoPulso(choice_id)  
  vibrarLeve()  
  enviarResposta(choice_id, tempo_resposta)  
  carregarProximaPergunta()
```

8) Observabilidade UX

- Eventos front:
 - `view_consent` , `start_question` , `choice_selected` , `choice_time` , `finish_test` , `view_result` .
- Monitoramento:
 - Tempo médio por pergunta.
 - Taxa de abandono.
 - Latência de carregamento de imagens/sons.

Fechamento da Camada

A **Camada UX Sensorial** transforma o teste em uma **experiência viva e coerente**, com microanimações, sons e navegação guiada.

Aqui a tecnologia garante que o usuário **sinta, escolha e avance com fluidez**, enquanto a IA registra e interpreta tudo em tempo real.

✓ CAMADA 13 — SISTEMA DE RESULTADO SENSORIAL (EXIBIÇÃO, FRASES, DADOS TÉCNICOS E TRIGGERS DE ATIVAÇÃO)

🎯 Objetivo da Camada

Definir como o **resultado final do Teste** é exibido ao usuário de forma **sensorial e imersiva**, ao mesmo tempo em que dispara **triggers técnicos** para ativar o ecossistema (Feed, Jogo, Conexões, Mapa, IA).

1) Estrutura da Tela de Resultado

Elemento	Descrição
Título Arquétipo	Nome do perfil dominante (ex.: <i>Guardião Solar</i>)
Frase-Código Vibracional	Mensagem curta gerada pela IA (motivacional e reflexiva)
Mapa Energético Circular	Gráfico visual dos vetores em 7 eixos
Frequência Hz	Valor numérico + representação fractal animada
Polaridade	Ícone Solar ☀️, Lunar 🌙 ou Neutra ⚖️
Símbolo Sensorial	Ícone/Fractal único do perfil atribuído
Feedback CTA	Pergunta: <i>"Como você se sentiu com esse resultado?"</i>

2) Frases-Código (Exemplos)

- Guardiã Solar → *"Você é a chama que protege, mas lembre-se de se aquecer também."*
- Oráculo Lunar → *"Você é silêncio que revela, mas precisa evitar se esconder."*
- Fluxo Selvagem → *"Você dança com o vento, mas precisa manter raízes."*

⌋ Todas as frases são geradas dinamicamente pela IA a partir do `score_vector`.

3) Triggers Pós-Resultado

Assim que o resultado é confirmado:

Sistema	Trigger	Ação
Feed Sensorial	<code>feed.injectTags(perfil)</code>	Ajusta curadoria de posts
Jogo da Transmutação	<code>jogo.start(perfil, tendencia)</code>	Inicia trilha personalizada
Mapa de Frequência	<code>mapa.update(frequencia_hz, polaridade)</code>	Atualiza posição vibracional
Conexões Autênticas	<code>conexoes.rebuild(perfil, sombra)</code>	Sugere conexões compatíveis
Aurah Kosmos (IA)	<code>aurah.sync(vetor, perfil)</code>	Salva assinatura vibracional

4) API Técnica

Endpoint

POST /teste/triggers

Body

```
{
  "user_id": "U123",
  "perfil": "Guardião Solar",
  "frequencia_hz": 9.28,
  "polaridade": "solar",
  "resonance_uuid": "xyz-abc-789"
}
```

Response

```
{
  "feed": "ok",
  "jogo": "ok",
  "mapa": "ok",
  "conexoes": "ok",
  "aurah": "ok"
}
```

5) Feedback do Usuário (Acionável)

- **Se conectado** → resultado confirmado e ativação completa.
- **Se em dúvida** → IA agenda micro-recalibração (2-3 perguntas).
- **Se não representou** → resultado marcado para revisão e IA sugere refazer teste em até 48h.

6) Visual e UX

- **Transições suaves** (fade, fractal crescendo).
- **Som adaptativo** à frequência final.
- **Mapa energético** renderizado em tempo real.
- **Botão Premium extra**: *"Compartilhar minha energia"* (gera imagem vibracional única).

← END Fechamento da Camada

O **Sistema de Resultado Sensorial** garante que o usuário receba uma **revelação vibracional imersiva** e que o FriendApp ative automaticamente todas as funcionalidades necessárias para

personalizar a jornada.

✓ CAMADA 14 — CAMADA DE ATIVAÇÃO: MODO JORNADA, FEED SENSORIAL, JOGO, CONEXÕES E MAPA

🎯 Objetivo da Camada

Definir como o resultado do teste **ativa imediatamente o modo jornada** e sincroniza os principais sistemas do FriendApp: Feed Sensorial, Jogo da Transmutação, Conexões Autênticas e Mapa de Frequência.

Essa é a etapa onde o usuário **sai do teste e entra na vivência prática da sua energia no app**.

1) Ativações Automáticas Pós-Teste

Sistema	Tipo de Ativação	Dados Utilizados
Feed Sensorial	Injeção de tags vibracionais	Perfil dominante + vetores
Jogo da Transmutação	Definição da trilha inicial	Frequência + tendência
Mapa de Frequência	Posicionamento e clusterização	Hz + polaridade
Conexões Autênticas	Reconstrução das sugestões	Perfil dominante + sombra
Aurah Kosmos (IA)	Sincronização e aprendizado	Vetores + assinatura vibracional

2) Fluxo UX de Ativação (Modo Jornada)

1. Usuário vê tela de resultado.
2. CTA → **“Ativar minha Jornada”**.
3. Ao clicar, app dispara triggers automáticos.
4. Microanimação: fractal se expande e conecta ícones de Feed, Jogo, Mapa e Conexões.
5. Usuário é guiado a explorar cada área com destaque sensorial.

3) Triggers Técnicos

```
ao_ativar_jornada(user_id, resultado):  
    FeedSensorial.injectTags(resultado.perfil_dominante, resultado.vetores)  
    JogoTransmutacao.start(resultado.frequencia_hz, resultado.tendencia)  
    MapaFrequencia.update(user_id, resultado.frequencia_hz, resultado.polaridade)  
    Conexoes.rebuild(user_id, resultado.perfil_dominante, resultado.sombra)  
    AurahKosmos.sync(user_id, resultado)
```

4) API Técnica

Endpoint: `POST /teste/ativar-jornada`

Body

```
{
  "user_id": "U123",
  "perfil_dominante": "Guardião Solar",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "polaridade": "solar",
  "sombra": false}
```

Response

```
{
  "feed": "ativado",
  "jogo": "iniciado",
  "mapa": "atualizado",
  "conexoes": "reconstruidas",
  "aurah": "sincronizada"
}
```

5) Regras de Encadeamento

- Feed é atualizado primeiro (conteúdo imediato).
- Jogo só inicia após Feed confirmado.
- Conexões são recriadas em último lugar (pois dependem do novo perfil).
- Aurah Kosmos sincroniza em paralelo com tudo.

6) Painel de Observabilidade

Campos monitorados por DevOps:

- Última ativação realizada.
- Tempo médio de propagação (< 500ms).
- Percentual de ativação concluída com sucesso.
- Logs vibracionais por sistema ativado.



Fechamento da Camada

A **Camada de Ativação** garante que o usuário não receba apenas um diagnóstico, mas **entre em ação imediatamente**, com Feed ajustado, Jogo iniciado, Conexões recriadas e Mapa atualizado.

Ela é a **ponte entre o insight e a vivência prática no FriendApp**.

✓ CAMADA 15 — BANCO DE DADOS, OBJETOS, RELACIONAMENTOS E NORMALIZAÇÃO DO TESTE

🎯 Objetivo da Camada

Definir a modelagem de dados que sustenta o Teste de Personalidade Energética, garantindo:

- Estrutura relacional clara.
- Normalização até 3FN (sem redundâncias).
- Suporte a consultas rápidas para IA e matching.
- Escalabilidade para milhões de execuções.

1) Tabelas Principais

energy_tests

- Armazena informações gerais de cada teste.

Campo	Tipo	Observações
id	UUID (PK)	Identificador único
user_id	UUID (FK → users.id)	Usuário dono do teste
status	Enum(started, submitted, completed, revoked)	Estado da sessão
perfil_id	UUID (FK → energy_profiles.id)	Perfil dominante atribuído
frequencia_hz	Float	Frequência final
score_vector	JSONB	Vetores energéticos (GIN index)
created_at	Timestamp	Início do teste
updated_at	Timestamp	Última atualização

energy_test_answers

- Registra cada resposta individual do teste.

Campo	Tipo	Observações
id	UUID (PK)	Identificador único
test_id	UUID (FK → energy_tests.id)	Referência ao teste

Campo	Tipo	Observações
question_id	UUID (FK → energy_questions.id)	Qual pergunta
choice_id	UUID (FK → energy_choices.id)	Qual opção foi escolhida
tempo_resposta_ms	Int	Tempo em milissegundos
coerencia_score	Float	Calculado pela IA
impacto_vetor	JSONB	Valores aplicados ao vetor final
timestamp	Timestamp	Hora da resposta

energy_profiles

- Catálogo dos arquétipos energéticos.

Campo	Tipo
id	UUID
nome	String
slug	String (único)
frequencia_base	Float
polaridade	Enum(solar, lunar, neutra)
elemento	String
sombra	String
legado	String
frase_retorno	String

test_activations

- Controla as ativações pós-teste (feed, jogo, mapa etc).

Campo	Tipo
id	UUID
test_id	UUID
feed_ativado	Bool
jogo_iniciado	Bool
mapa_atualizado	Bool
conexoes_reconstruidas	Bool
aurah_sincronizada	Bool
created_at	Timestamp

2) Relacionamentos ER

```
erDiagram
    users ||--o{ energy_tests : has
```

```
energy_tests ||--o{ energy_test_answers : contains
energy_tests }|--|| energy_profiles : maps
energy_tests ||--o{ test_activations : triggers
energy_test_answers }|--|| energy_questions : references
energy_test_answers }|--|| energy_choices : selects
```

3) Índices e Performance

- **GIN** index nos campos JSONB (`score_vector` , `impacto_vetor`).
- Índices compostos: (`user_id` , `created_at`) em `energy_tests` .
- Particionamento por data em `energy_test_answers` para reduzir volume.
- **ETL para Data Warehouse** (ex.: BigQuery) → dados históricos migrados periodicamente para análises complexas, mantendo Postgres/Firestore leves.

4) Normalização e Regras

- Cada resposta pertence a um teste (`test_id`).
- Cada teste pertence a um usuário (`user_id`).
- Vetores salvos no JSONB garantem flexibilidade sem quebrar modelo relacional.
- Sessões não concluídas expiram em 24h e são marcadas como `revoked` .

Fechamento da Camada

Essa camada garante que a base de dados seja **leve, consultável e expansível**, sem riscos de inconsistência.

Os devs têm agora um **modelo relacional + JSON híbrido** com suporte para IA, auditoria e analytics.

CAMADA 16 — LÓGICA DE ATRIBUIÇÃO DO PERFIL ENERGÉTICO (CÁLCULO, CONFIANÇA, DESEMPATE E MODO TRANSIÇÃO)

Objetivo da Camada

Definir com precisão **como** as respostas do teste viram um **vetor energético consolidado** e, a partir dele, **como** é escolhido o **perfil dominante/ secundário**, com **confiança, desempate e fallback** padronizados.

1) Entradas do Cálculo

- `answers[]` : lista de respostas com `choice_id` , `tempo_ms` , `question_weight` , `impacto_vetor` (da **Matriz V2**, Camada 03).
- `timing_curve(tempo_ms)` : fator $\in [0.7, 1.2]$ que corrige peso pelo tempo (impulsivo/hesitante).
- `coerencia_score` $\in [0,1]$: qualidade local da resposta (Camada 04).
- `matriz.version` : versão aplicada (auditável).

2) Cálculo do Vetor Consolidado

2.1 Acúmulo bruto por eixo (7 eixos: foco, visao, presenca, forca, fluxo, sensibilidade, sombra)

Para cada resposta `r`:

```
peso_r = r.question_weight * timing_curve(r.tempo_ms) * clamp(0.8 + 0.4*r.coerencia_score, 0.8, 1.2)
acumulado[eixo] += r.impacto_vetor[eixo] * peso_r
```

2.2 Normalização (por robustez)

Usamos **tanh** para saturar extremos e mapear para [0,1]:

```
v_norm[eixo] = 0.5 * (tanh(acumulado[eixo]) + 1.0)
```

| Garante estabilidade contra outliers.

2.3 Ajuste de Sombra

A sombra entra tanto como eixo quanto como **penalizador global**:

```
sombra = v_norm["sombra"]
penalizador_sombra = 1 - 0.15 * sombra // até -15% no score luminoso
```

3) Score de Perfil (Matching Vetorial)

Cada perfil `p` possui um **centro vetorial** (prototipo) definido em `energy_profiles_proto`:

```
{
  "guardiao_solar": { "foco":0.7, "visao":0.6, "presenca":0.8, "forca":0.7, "fluxo":0.5, "sensibilidade":0.5, "sombra":0.3 },
  "oraculo_lunar": { ... }
```

```
}
```

3.1 Similaridade

Usar **cosine similarity** com pesos por eixo (opcionalmente diferentes por família):

```
cos = dot(v_norm, proto_p) / (||v_norm||*||proto_p|| + ε)
```

3.2 Ajustes de contexto

- Penalizar se `polaridade` do perfil conflitar com a inferida (Camada 05):

```
ajuste_polaridade = (polaridade_inferida == polaridade_p) ? 1.0 : 0.95
```

- Penalizador de sombra (do §2.3):

```
score_p = cos * ajuste_polaridade * penalizador_sombra
```

4) Seleção, Confiança e Desempate

1. **Dominante preliminar:** `p* = argmax_p(score_p)`

2. **Confiança:**

```
conf_p* = softmax(scores)[p*] // ou margem: score_p* - segundo_maior
```

Critério de aceite:

- `conf_p* >= 0.75` e `score_p* >= 0.62` → **perfil dominante aceito.**

1. **Híbrido (dominante + secundário)** se:

- `conf_p* ∈ [0.65, 0.75)` ou
- `|score_p* - score_p2| < 0.04` (empate técnico)
→ retornar `p*` (dominante) e `p2` (secundário), com `peso_sec = normalize(score_p2/score_p*)`.

1. **Modo Transição** se:

- `conf_p* < 0.65` ou
- `score_p* < 0.58`
→ ativar **Transição** (Camada 09) + micro-calibração em 24–72h.

Esses thresholds são parâmetros configuráveis (`matching.thresholds`) e versionados.

5) Polaridade e Tendência

5.1 Polaridade

Calcular pela média ponderada dos eixos luminosos vs. sombra:

```
lum = mean(foco, visao, presenca, forca, fluxo, sensibilidade)
polarity_score = lum - 0.8*sombra
polaridade =
  polarity_score > +0.08 → "positiva/solar"
  |polarity_score| <= 0.08 → "neutra"
  polarity_score < -0.08 → "negativa/lunar densa"
```

5.2 Tendência (dinâmica)

Compara o vetor atual com histórico (último ativo):

```
delta = ||v_norm_atual - v_norm_prev||_2
if delta > 0.22 → "transicao"
else if lum aumenta ≥ 10% e sombra cai ≥ 10% → "expansao"
else se lum cai e sombra sobe ≥ 10% → "colapso"
senão → "estavel"
```

6) Exemplos Rápidos

Ex. A — Dominante claro

- `score_guardiao_solar = 0.84` , `conf = 0.81` → **Guardião Solar** (aceito)
- Secundário a 0.68 → diferença > 0.04 → **não** híbrido.

Ex. B — Híbrido

- `p1=0.72` , `p2=0.70` , diferença 0.02 → **híbrido** (dominante+secundário)

Ex. C — Transição

- melhor score 0.59 e conf 0.61 → **Transição** + micro-calibração.

7) Anti-abuso e Qualidade

- **Tempo mínimo** por resposta: <300ms → peso cortado a 40%.
- **Padrão randômico** (baixa coerência média <0.4) → reduzir `penalizador_sombra` para 0.85 (mais conservador).

- **Sessão suspeita:** marcar `flag_suspeita=true` e solicitar reteste assistido.

8) Persistência e Auditoria

Salvar em `energy_tests`:

```
{
  "score_vector": { ... v_norm ... },
  "perfil_dominante": "guardiao_solar",
  "perfil_secundario": "oraculo_lunar",
  "conf_dominante": 0.78,
  "scores_perfil": { "guardiao_solar":0.84, "oraculo_lunar":0.69, ... },
  "polaridade": "positiva",
  "tendencia": "expansao",
  "matching_version": "v2.0.3",
  "matriz_version": "v2.1.0",
  "thresholds_version": "v2.0"
}
```

Índice GIN sobre score_vector e scores_perfil (JSONB) para consultas rápidas.

9) Pseudocódigo do Matching

```
function atribuirPerfil(respostas, matriz, protos, thresholds):
  v = consolidarVetor(respostas, matriz)    # §2
  sombra = v["sombra"]
  penal_sombra = 1 - 0.15 * sombra

  for p in protos:
    cos = cosineSimilarity(v, protos[p])
    adjPol = (inferirPolaridade(v) == polPerfil[p]) ? 1.0 : 0.95
    score[p] = cos * adjPol * penal_sombra

  p1 = argmax(score); p2 = secondBest(score)
  conf = softmax(score)[p1]

  if conf >= 0.75 and score[p1] >= 0.62:
    return Dominante(p1, conf, score, v)
  if conf >= 0.65 or abs(score[p1] - score[p2]) < 0.04:
    return Hibrido(p1, p2, conf, score, v)
  return Transicao(conf, score, v)
```

10) Testes e Métricas

- **QA unitário:** casos de borda (empate, sombra alta, tempos extremos).
- **A/B interno:** validar thresholds em coorte piloto.
- **Métricas:**
 - % em Transição (meta: 10–18%).
 - Concordância usuário ("Conectado") $\geq 75\%$.
 - Tempo médio de processamento do matching $< 150\text{ms}$ (worker).

Fechamento da Camada

Com esta lógica, os devs têm **todos os números, fórmulas e decisões** para implementar a atribuição de perfil **sem nenhuma dúvida**: cálculo vetorial, similaridade, confiança, desempate, polaridade, tendência, anti-abuso, persistência e auditoria.

CAMADA 17 — ENDPOINTS DO TESTE (CRUD + STREAM DE RESPOSTAS + FINALIZAÇÃO 202 + WORKER)

Objetivo da Camada

Definir todos os **endpoints REST** do Teste de Personalidade Energética, cobrindo:

- Criação e ciclo de vida da sessão.
- Registro das respostas em tempo real.
- Finalização assíncrona com retorno 202.
- Worker responsável pelo processamento.

1) Endpoints REST

1.1 Criar Sessão de Teste

POST /api/teste/sessao

Body

```
{
  "user_id": "U123",
  "device_id": "D456",
  "consents": {
    "leitura": true,
    "privacidade": true,
    "emocional": true
  }
}
```

```
}
```

Response 201

```
{  
  "test_id": "T-9841",  
  "token_teste": "jwt.session.token",  
  "expires_in": 1800  
}
```

1.2 Enviar Resposta

POST /api/teste/resposta

Body

```
{  
  "test_id": "T-9841",  
  "pergunta_id": "Q-07",  
  "choice_id": "IMG-014-AZURE",  
  "tempo_resposta_ms": 3200  
}
```

Response 200

```
{  
  "ok": true,  
  "proxima_pergunta": "Q-08"  
}
```

1.3 Finalizar Teste (assíncrono)

POST /api/teste/finalizar

Body

```
{ "test_id": "T-9841" }
```

Response 202

```
{  
  "status": "processing",  
}
```

```
"message": "Sua frequência está sendo sintonizada."
}
```

1.4 Consultar Resultado

GET /api/teste/resultado/:test_id

Response 200

```
{
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Oráculo Lunar",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "score_vector": { "foco":0.74, "visao":0.68, "sombra":0.37, "...": "..." },
  "frase_codigo": "Você é chama que aquece e guia.",
  "ativacoes": {
    "feed": true,
    "jogo": true,
    "mapa": true,
    "conexoes": true},
  "flags": {
    "sombra_ativa": false,
    "transicao": false}
}
```

2) Fluxo de Mensagens (Assíncrono)

sequenceDiagram

participant App

participant API

participant Kafka

participant Worker

participant DB

participant IA

App->>API: POST /teste/finalizar

API->>DB: marcar status=submitted

API->>Kafka: publish teste_finalizado

API->>App: 202 Accepted (processing)

Worker->>Kafka: consume teste_finalizado

Worker->>DB: fetch respostas

```
Worker→>IA: calcular vetores + perfis
IA→>Worker: resultado completo
Worker→>DB: salvar resultado + ativacoes
Worker→>App: push notify "Resultado pronto"
```

3) Worker de Processamento

- Escrita em **Go ou Python**.
- Consome mensagens da fila (`teste_finalizado`).
- Processa vetores via Matriz de Ponderação (Camada 03).
- Executa atribuição de perfil (Camada 16).
- Persiste resultado em `energy_tests` + triggers.
- Dispara push/notify.

4) Segurança dos Endpoints

- `token_teste` → válido apenas para sessão do teste, expira em 30min.
- Autenticação via JWT + OAuth2.
- Rate limiting: 10 req/s por usuário.
- Payloads com `hash_integridade` .
- Logs em `api_audit_logs` (rota, user_id, IP, latência).

5) Observabilidade

- Métricas coletadas em Prometheus/Grafana:
 - Latência por rota.
 - Tempo médio de processamento do worker.
 - Taxa de sucesso vs erro.
 - Qtd. testes processados/hora.
- Alertas:
 - Worker parado > 1min → alerta Slack.
 - Fila Kafka > 10k mensagens → escala pods.

Fechamento da Camada

Essa camada garante que os **endpoints sejam claros, estáveis e escaláveis**.

Com `finalizar` assíncrono (202 + fila + worker), a experiência é rápida para o usuário e resiliente no backend.

✓ CAMADA 18 — SISTEMA DE SOMBRAS E BLOQUEIOS ENERGÉTICOS (DETECÇÃO, ACIONAMENTO E DESBLOQUEIO)

🎯 Objetivo da Camada

Detectar **padrões de sombra/bloqueio** nas respostas do teste e no pós-teste, classificar o **grau de impacto**, e **acionar intervenções** (Jogo da Transmutação, Feed, IA em modo cuidadoso, micro-calibrações) até a **resolução ou estabilização**.

1) Sinais de Entrada (features)

Sinal	Fonte	Descrição
sombra_axis	Vetor final	Valor normalizado do eixo "sombra" (Camada 16)
tempo_var	Coleta (Camada 04)	Variância de tempo de resposta entre perguntas
coerencia_med	Coleta	Média de coerência entre escolhas consecutivas
nlp_tokens_neg	NLP (Camada 05)	Presença de léxico de autoproteção/negação (ex.: "tanto faz", "não sinto")
pattern_rand	Coleta	Padrão randômico (alta entropia sem correlação ao estímulo)
repeticoes	Coleta	Repetição de uma mesma opção sem relação com o estímulo
delta_hist	Histórico (Camada 09)	Mudança abrupta vs. último perfil ativo

2) Índice de Sombra (shadow_index)

```
shadow_index = w1*sombra_axis  
              + w2*zscore(tempo_var)  
              + w3*(1 - coerencia_med)  
              + w4*nlp_tokens_neg_score  
              + w5*pattern_rand  
              + w6*repeticoes_norm  
              + w7*delta_hist_norm  
  
# pesos padrão (versionados):  
w = {0.45, 0.10, 0.10, 0.10, 0.10, 0.05, 0.10}  
shadow_index = clamp(shadow_index, 0, 1)
```

Faixas

- **Leve:** 0.30–0.49
- **Moderada:** 0.50–0.69
- **Alta:** ≥ 0.70

Pesos e thresholds ficam em shadow_config.version = v2.0.

3) Tipologia de Sombra (taxonomia)

Código	Nome	Indicadores principais	Vetores afetados
S-CTRL	Controle Exagerado	sombra_axis↑ + coerencia_med↑ + tempo_var baixo (rigidez)	foco, força, presença
S-EVAS	Evasão/Desligamento	nlp_tokens_neg↑ + tempo_var↑ + pattern_rand↑	fluxo, sensibilidade
S-JUDG	Autojulgamento	nlp_tokens_neg↑ ("errado", "culpa")	sensibilidade, força
S-DISP	Dispersão	tempo_var↑ + coerencia_med↓	foco, fluxo
S-HSOC	Hiper-social	repeticoes em itens "aprovação/likes"	presença, foco
S-RETR	Retraimento	escolha de isolamento + aumento de sombra vs. histórico	presença, sensibilidade

Uma sessão pode ter 1 primária e 1 secundária (se score dentro de 85% da primária).

4) Ações por Grau (playbook)

Grau	IA Aurah	Feed	Jogo	UX	Reavaliação
Leve	Modo normal com tom acolhedor	Conteúdo de estabilização	Missão curta (5–10 min)	Banner sutil "respirar 1 min"	N/A
Moderada	Modo Cuidadoso (tom leve, evitar confronto)	Feed reduz estímulos altos	Trilha "Aterramento & Clareza"	Tela "pausa consciente" opcional	Micro-calibração em 48h
Alta	Modo Cuidadoso+ (limites)	Feed focado em autocuidado	Jogo limitado a 1 esfera/dia	Pop-up de cuidado + opção "falar com alguém"	Micro-calibração em 24h

5) Estado & Máquina de Sombra

Estados: none → light → moderate → high → resolving → resolved

Transições:

```
if shadow_index < 0.30: state = none
elif 0.30..0.49: state = light
```



```
elif 0.50..0.69: state = moderate
else: state = high
```

```
on mission_completed or feedback_conectado:
```

```
    state = resolving
```

```
on reavaliacao_ok (shadow_index < 0.30 for 7d):
```

```
    state = resolved
```

6) Persistência (banco de dados)

Tabela: **energy_shadow_state**

Campo	Tipo	Descrição
user_id	UUID (PK)	Usuário
state	Enum	none/light/moderate/high/resolving/resolved
primary_code	Enum	S-CTRL/S-EVAS/...
secondary_code	Enum	opcional
shadow_index	Float	0–1
updated_at	Timestamp	auditoria

Tabela: **energy_shadow_events**

Campo	Tipo	Descrição
event_id	UUID (PK)	
user_id	UUID	
test_id	UUID	origem
codes	JSONB	{primary, secondary, scores}
shadow_index	Float	
action_taken	JSONB	{ia_mode, feed_mode, mission}
created_at	Timestamp	

Tabela: **shadow_playbook**

Campo	Tipo	Descrição
code	Enum	S-CTRL/S-EVAS/...
level	Enum	light/moderate/high
ia_mode	Enum	normal/safe/safe+
feed_policy	Enum	normal/low_stim/care_only
mission_id	String	id da missão padrão
version	String	v2.*

Índices GIN em codes e action_taken.

7) Endpoints

7.1 Avaliar Sombra (worker)

POST /internal/shadow/evaluate

Body

```
{
  "user_id": "U123",
  "test_id": "T-9841",
  "signals": {
    "sombra_axis": 0.61,
    "tempo_var": 0.42,
    "coerencia_med": 0.58,
    "nlp_tokens_neg": 0.22,
    "pattern_rand": 0.11,
    "repeticoes": 0.05,
    "delta_hist": 0.09
  }
}
```

Response

```
{
  "shadow_index": 0.55,
  "state": "moderate",
  "primary_code": "S-EVAS",
  "secondary_code": "S-DISP",
  "actions": {
    "ia_mode": "safe",
    "feed_policy": "low_stim",
    "mission_id": "M-ATERRAMENTO-001",
    "reavaliacao": "48h"
  }
}
```

7.2 Aplicar Ações

POST /internal/shadow/apply

```
{
  "user_id": "U123",
```

```
"state":"moderate",
"primary_code":"S-EVAS",
"actions":{"ia_mode":"safe","feed_policy":"low_stim","mission_id":"M-ATERRAMENTO-001"}
}
```

Response `{ "ok": true }`

7.3 Consultar Estado (app/painel)

`GET /api/shadow/state`

Response

```
{
  "state":"moderate",
  "primary_code":"S-EVAS",
  "shadow_index":0.55,
  "last_update":"2025-09-04T18:15:00Z"
}
```

8) Integrações

- **IA Aurah** → muda para `ia_mode = safe/safe+` (tom + cadência).
- **Feed** → política `low_stim/care_only` reduz estímulos e prioriza autocuidado.
- **Jogo** → missão específica do playbook; limita progressos em `high`.
- **Notificações** → convites gentis (sem pressão), micro-calibração agendada.
- **Conexões** → prioriza matches acolhedores; evita confrontos.

9) UX & Mensagens (exemplos)

- **Moderada:**

“Seu campo pede cuidado. Vamos ancorar sua presença com 1 exercício de aterramento (5 min)?”

- **Alta:**

“Pausa consciente: respire com a gente por 3 minutos. Depois seguimos com a sua jornada.”

- **Resolving → Resolved:**

“Você estabilizou lindamente. Vamos retomar sua trilha com suavidade.”

10) Observabilidade & Métricas

- % de usuários por estado (none/light/moderate/high/resolving/resolved)
- Tempo médio em cada estado
- Conclusão de missão vs. queda do `shadow_index`
- Aderência ao `feed_policy` e impacto em engajamento
- Alerta se `high` > 8% por 7 dias (rever matriz/pesos)

11) Testes (QA)

- **Unit:** cálculo do `shadow_index` por cenários (tempo alto, coerência baixa, NLP negativo).
- **Contract:** endpoints `evaluate` / `apply` e idempotência.
- **E2E:** fluxo "teste → detecta S-EVAS moderada → aplica missão → reavalia 48h → resolving → resolved".
- **A/B:** validar pesos `w` em coorte piloto.

Fechamento da Camada

Esta camada transforma sinais brutos em **diagnóstico objetivo de sombra**, aplica **intervenções graduais**, protege a UX com **IA em modo cuidadoso**, e conduz o usuário de volta à **estabilidade** com métricas e auditoria.

CAMADA 19 — GAMIFICAÇÃO: JOGO DA TRANSMUTAÇÃO (TRILHAS, MISSÕES, XP, LIMITES E TELEMETRIA)

Objetivo da Camada

Transformar o resultado do teste em **ação prática e gamificada**, conectando o usuário ao **Jogo da Transmutação** com trilhas, missões e recompensas adaptadas ao seu perfil e estado vibracional.

O objetivo é **engajar o usuário em microdesafios que auxiliem na transmutação energética real**.

1) Ativações do Jogo Pós-Teste

- Perfil dominante → define **trilha inicial**.
- Tendência (expansão, colapso, transição) → ajusta **nível de desafio**.
- Sombra detectada → ativa **missão de desbloqueio específica**.

- Polaridade → regula o **tom das interações**.

2) Estrutura de Trilhas

Cada trilha é uma sequência de **missões vibracionais**:

Trilha	Perfil de Entrada	Foco
Expansão Solar	Guardião, Guerreiro, Visionário	Liderança, coragem, ação
Introspecção Lunar	Oráculo, Curador, Guardião	Silêncio, cura, sabedoria
Equilíbrio Neutro	Perfis híbridos	Harmonia, observação
Transição	Usuários sem perfil dominante	Autodescoberta, clareza

3) Tipos de Missão

Tipo	Exemplo	Tempo médio
Respiração Guiada	3 min com som 432Hz	3–5 min
Escrita Reflexiva	“Escreva algo que gostaria de liberar hoje”	5–10 min
Ação Social	Mandar mensagem para 1 conexão vibracional	variável
Movimento Sensorial	Pequeno exercício físico/alongamento	5 min
Missão de Cura	Escutar frequência 528Hz + visualizar luz	7 min

4) Sistema de XP e Recompensas

- Cada missão concluída = **+10 XP**.
- Missão de desbloqueio de sombra = **+25 XP**.
- Missão de transição concluída = **+15 XP + selo “Clareza”**.
- XP acumula em **totens vibracionais** no Jogo da Transmutação.

5) Limites e Progressão

- Usuário pode concluir **máx. 3 missões/dia** (para evitar overload).
- Sombra em estado alto → limite de 1 missão/dia.
- Trilhas evoluem em **níveis** (Lv.1 → Lv.5).
- Cada nível desbloqueia **símbolos energéticos** exibidos no Mapa e no Perfil.

6) Telemetria e Métricas

Monitorar:

- % de usuários que concluem a primeira missão.
- Tempo médio gasto por missão.

- Taxa de abandono em missões de cura.
- Impacto no `shadow_index` antes/depois das missões.

7) Integração Técnica

Endpoint

POST `/api/jogo/missao/completar`

Body

```
{
  "user_id": "U123",
  "missao_id": "M-RESPIRACAO-432",
  "resultado": "concluida"
}
```

Response

```
{
  "xp_ganho": 10,
  "totem_atualizado": "T-SOLAR-01",
  "selo_desbloqueado": null
}
```

Persistência

Tabela: `game_missions_log`

Campo	Tipo
<code>log_id</code>	UUID
<code>user_id</code>	UUID
<code>missao_id</code>	String
<code>resultado</code>	Enum(concluida, desistiu)
<code>xp_ganho</code>	Int
<code>selo_desbloqueado</code>	String/null
<code>timestamp</code>	Timestamp

← END Fechamento da Camada

Essa camada conecta o Teste ao **Jogo da Transmutação**, garantindo que o resultado não seja só informativo, mas sim **um portal para ação prática**.

Cada missão é curta, simbólica e calibrada pelo perfil energético do usuário.

✓ CAMADA 20 — PERFORMANCE, ESCALABILIDADE E FILAS ASSÍNCRONAS (FINALIZAÇÃO DO TESTE E PROCESSAMENTO IA)

🎯 Objetivo da Camada

Definir a arquitetura que garanta **baixa latência na UX** e **alta resiliência no backend**.

O processamento da IA é feito em **fluxo assíncrono**, desacoplando a finalização do teste da geração do resultado.

1) Problema Resolvido

- **Antes (síncrono)** → `POST /finalizar` bloqueava o app até processar todas as respostas + IA (~3–5s).
- **Agora (assíncrono)** → `POST /finalizar` retorna **202 Accepted** em <200ms, enquanto o **worker** processa offline e envia push/notify quando pronto.

2) Fluxo de Finalização

```
sequenceDiagram
    participant App
    participant API
    participant Kafka/Redis
    participant Worker
    participant DB
    participant IA

    App->>API: POST /teste/finalizar
    API->>DB: marca status=submitted
    API->>Kafka/Redis: publish teste_finalizado
    API->>App: 202 Accepted {processing}

    Worker->>Kafka/Redis: consume teste_finalizado
    Worker->>DB: fetch respostas
    Worker->>IA: processar vetores + atribuir perfil
    IA->>Worker: resultado completo
    Worker->>DB: salvar resultado + ativacoes
    Worker->>App: push notify "Resultado pronto"
```

3) Arquitetura Técnica

- **Fila de Mensagens:** Kafka (prod) ou Redis Streams (MVP).

- **Worker Pool:**
 - Escritos em Go/Python.
 - Escalam horizontalmente (Kubernetes + HPA).
 - **Timeouts:** mensagens expiradas em 24h → reteste sugerido.
 - **Idempotência:** reprocessamento seguro se worker falhar.
-

4) SLOs de Performance

Métrica	Meta
Tempo de resposta POST /finalizar	< 200ms
Tempo médio de processamento IA	< 2s
Disponibilidade do worker	99.9%
Throughput suportado	10k testes/min
Latência de fila	< 1s (p95)

5) Escalabilidade

- **Kubernetes HPA:** escala workers baseado em lag da fila.
 - **Multi-região:** usuários roteados ao cluster mais próximo.
 - **Replicação de banco:** leituras via réplicas; gravações apenas no master.
 - **Fallback:** se fila indisponível → processar em modo síncrono simplificado (apenas perfis básicos).
-

6) Observabilidade

- **Métricas monitoradas:**
 - Tamanho da fila.
 - Tempo médio de processamento.
 - % de falhas por worker.
 - **Painéis (Grafana/Prometheus):**
 - Latência API vs Worker.
 - Volume por região.
 - Estados de sombra em tempo real.
 - **Alertas:**
 - Fila > 10k mensagens → escala automática.
 - Worker inativo > 60s → alerta Slack.
-

7) Persistência e Logs

- Logs de cada evento `teste_finalizado` armazenados em `processing_logs`.
- Campos: `event_id`, `test_id`, `status`, `worker_id`, `latencia_total`.
- Resultados salvos somente após **confirmação da IA** → estado = `completed`.

Fechamento da Camada

Com essa camada, o FriendApp garante que o teste seja **instantâneo para o usuário e infinitamente escalável no backend**.

O processamento assíncrono com filas elimina travamentos e mantém a UX fluida mesmo em picos globais.

CAMADA 21 — API PRINCIPAL: REQUESTS, RESPONSES, ERROS E BOAS PRÁTICAS

Objetivo da Camada

Padronizar **contratos HTTP** do módulo “Teste de Personalidade Energética”, cobrindo:

- **Autenticação, headers e versionamento**
- **Requests/Responses** (com schemas)
- **Idempotência, rate limiting e erros padronizados**
- Fluxo **assíncrono** com **202 Accepted** (fila + worker)

1) Convenções Gerais

- **Base URL:** `https://api.friendapp.com/v1`
- **Auth:** `Authorization: Bearer <JWT>` (OAuth2)
- **Escopo:** `scope=energy:test`
- **Conteúdo:** `Content-Type: application/json; charset=utf-8`
- **Versionamento:** prefixo `/v1` + `X-API-Version: 1`
- **TLS:** obrigatório (HSTS + TLS 1.3)
- **CORS:** `Origin` verificado (lista de apps oficiais)

Headers padrão

- `Authorization: Bearer <token>`
- `X-Test-Token: <jwt_sessao_do_teste>` (somente rotas do teste)
- `X-Request-Id: <uuid>` (correlação / tracing)

- **Idempotency-Key:** <uuid> (POST sensíveis: resposta/finalizar)

2) Endpoints (públicos)

2.1 Criar sessão do teste

POST /teste/sessao

Body

```
{
  "device_id": "D456",
  "consents": { "leitura": true, "privacidade": true, "emocional": true },
  "app_version": "1.0.0"
}
```

201

```
{ "test_id": "T-9841", "token_teste": "jwt.session.token", "expires_in": 1800 }
```

Erros: 400/422 (consent inválido), 401, 409 (sessão ativa).

2.2 Enviar resposta (stream)

POST /teste/resposta

Headers: X-Test-Token, Idempotency-Key (recomendado)

Body

```
{
  "test_id": "T-9841",
  "pergunta_id": "Q-07",
  "choice_id": "IMG-014-AZURE",
  "tempo_resposta_ms": 3200
}
```

200

```
{ "ok": true, "proxima_pergunta": "Q-08" }
```

Erros: 400 (payload), 401/403 (token), 410 (sessão expirada), 409 (duplicado sem idempotência).

2.3 Finalizar (assíncrono)

POST /teste/finalizar

Headers: X-Test-Token , Idempotency-Key

Body

```
{ "test_id": "T-9841" }
```

202

```
{ "status": "processing", "message": "Sua frequência está sendo sintonizada." }
```

Publica evento teste_finalizado (Kafka/Redis). Resultado será consultado via GET ou push.

Erros: 400 (sem test_id), 401/403 , 409 (já finalizado), 410 (revogado).

2.4 Buscar resultado

GET /teste/resultado/{test_id}

200

```
{
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Analítico Introspectivo",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "score_vector": { "foco":0.74, "visao":0.68, "presenca":0.82, "forca":0.61, "fluxo":0.47, "sensibilidade":0.59, "sombra":0.37 },
  "frase_codigo": "Você é a luz que guia e expande.",
  "flags": { "sombra_ativa": false, "transicao": false },
  "ativacoes": { "feed": true, "jogo": true, "mapa": true, "conexoes": true },
  "matching_meta": {
    "conf_dominante": 0.78,
    "scores_perfil": { "guardiao_solar":0.84, "oraculo_lunar":0.69 },
    "matriz_version": "v2.1.0",
    "matching_version": "v2.0.3"
  }
}
```

404 (não encontrado), 425 (resultado ainda processando — opcional), 401/403.

2.5 Ativar jornada (triggers encadeados)

POST /teste/ativar-jornada

Body

```
{
  "user_id": "U123",
  "perfil_dominante": "Guardião Solar",
  "frequencia_hz": 9.28,
  "tendencia": "expansao_emocional",
  "polaridade": "solar",
  "sombra": false}
```

200

```
{ "feed": "ativado", "jogo": "iniciado", "mapa": "atualizado", "conexoes": "reconstruidas", "aura": "sincronizada" }
```

Erros: 409 (estado inconsistente), 422 (dados faltando).

2.6 Estado de sombra (painel/app)

GET /shadow/state

200

```
{ "state": "moderate", "primary_code": "S-EVAS", "shadow_index": 0.55, "last_update": "2025-09-04T18:15:00Z" }
```

3) Endpoints (internos / worker)

- POST /internal/teste/processar → consume evento, recomputa (idempotente)
- POST /internal/shadow/evaluate → calcula shadow_index
- POST /internal/shadow/apply → aplica políticas (feed/jogo/ia_mode)

Autenticação interna por mTLS + service account (não expor publicamente).

4) Esquema de Erros Padronizado

Formato

```
{
  "error": {
    "code": "VALIDATION_ERROR",
    "message": "Campo 'choice_id' é obrigatório.",
    "details": [
      { "field": "choice_id", "rule": "required" }
    ]
  }
}
```

```
],  
  "request_id": "a3f7-..."  
}  
}
```

Códigos

- 400 VALIDATION_ERROR
- 401 UNAUTHORIZED
- 403 FORBIDDEN
- 404 NOT_FOUND
- 409 CONFLICT (duplicidade/idempotência, sessão já finalizada)
- 410 GONE (sessão expirada/revogada)
- 422 UNPROCESSABLE_ENTITY (regra de negócio)
- 425 TOO_EARLY (resultado ainda em processamento – opcional)
- 429 RATE_LIMITED
- 500 INTERNAL_ERROR
- 503 SERVICE_UNAVAILABLE

5) Idempotência, Retries e Timeouts

- **Idempotency-Key** obrigatório em:
 - POST /teste/resposta
 - POST /teste/finalizar
- O servidor registra a **primeira resposta** por chave e repete o **mesmo retorno** nas subsequentes (24h).
- **Retries:**
 - Cliente pode re-tentar **GETs**.
 - Para **POSTs** usar sempre **Idempotency-Key**.
- **Timeouts:**
 - API timeout: **30s**
 - Worker timeout por job: **60s** (requeue automático)
- **Conexão:** keep-alive; **X-Request-Id** para correlação (OpenTelemetry).

6) Rate Limiting e Segurança

- **Limites** (default): **60 req/min/user** ; estouro controlado via token bucket.

- **Respostas:** 429 + headers `Retry-After` .
- **Proteções:**
 - Validação de `X-Test-Token` (vinculado ao `test_id` , expira em 30min).
 - Verificação de integridade (`hash_integridade` no payload, opcional).
 - Sanitização/escapes; logs **sem PII sensível**.
 - **JSON schema** validado antes do processamento.

7) Exemplos de cURL

Criar sessão

```
curl -X POST https://api.friendapp.com/v1/teste/sessao \  
-H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \  
-d '{ "device_id":"D456", "consents":{"leitura":true,"privacidade":true,"emocional":true} }'
```

Enviar resposta (idempotente)

```
curl -X POST https://api.friendapp.com/v1/teste/resposta \  
-H "Authorization: Bearer $TOKEN" \  
-H "X-Test-Token: $TEST_TOKEN" \  
-H "Idempotency-Key: 3b3f3c7d-0e5c-4a" \  
-d '{ "test_id":"T-9841","pergunta_id":"Q-07","choice_id":"IMG-014-AZURE","tempo_respo  
sta_ms":3200 }'
```

Finalizar (assíncrono)

```
curl -X POST https://api.friendapp.com/v1/teste/finalizar \  
-H "Authorization: Bearer $TOKEN" \  
-H "X-Test-Token: $TEST_TOKEN" \  
-H "Idempotency-Key: 9812cce1-51a8-4b" \  
-d '{ "test_id":"T-9841" }'
```

Buscar resultado

```
curl -X GET https://api.friendapp.com/v1/teste/resultado/T-9841 \  
-H "Authorization: Bearer $TOKEN"
```

8) OpenAPI (trecho ilustrativo)

```
openapi: 3.0.3
info:
  title: FriendApp - Teste de Personalidade Energética
  version: "1.0.0"
servers:
  - url: https://api.friendapp.com/v1
components:
  securitySchemes:
    bearerAuth:
      type: http
      scheme: bearer
paths:
  /teste/sessao:
    post:
      security: [{ bearerAuth: [] }]
      summary: Cria sessão do teste
      requestBody:
        required: true
      responses:
        "201":
          description: Sessão criada
        "400": { description: Erro de validação }
        "401": { description: Não autorizado }
  /teste/resposta:
    post:
      security: [{ bearerAuth: [] }]
      summary: Registra resposta
      parameters:
        - in: header
          name: Idempotency-Key
          schema: { type: string }
      responses:
        "200": { description: OK }
        "409": { description: Conflito / duplicado }
  /teste/finalizar:
    post:
      security: [{ bearerAuth: [] }]
      summary: Finaliza teste (assíncrono)
      responses:
        "202": { description: Processando }
        "409": { description: Já finalizado }
  /teste/resultado/{test_id}:
    get:
      security: [{ bearerAuth: [] }]
      parameters:
```

```
- in: path
  name: test_id
  required: true
  schema: { type: string }
responses:
  "200": { description: Resultado pronto }
  "404": { description: Não encontrado }
```

9) Observabilidade e Auditoria

- **Audit Log** por chamada: rota, `user_id`, `request_id`, IP, status, latência.
- **Tracing distribuído** (OpenTelemetry): `X-Request-Id` propagado.
- **Métricas** (Prometheus): latência p95/p99 por rota; rate de 4xx/5xx; tempo médio do worker.

← **Fechamento da Camada**

A **Camada 21** sela os **contratos de integração**: rotas, headers, erros, idempotência, limites e fluxo assíncrono.

Com isso, o time consegue integrar frontend, backend e worker **sem ambiguidades** e com **observabilidade** desde o primeiro deploy.

✅ **CAMADA 22 — SEGURANÇA, PRIVACIDADE, CRIPTO (AES-256/RSA), MÁSCARA DE LOGS E LGPD/GDPR**

🎯 **Objetivo da Camada**

Garantir que o **Teste de Personalidade Energética** opere com **segurança zero-trust**, **privacidade por padrão** e **conformidade LGPD/GDPR**, cobrindo:

- Criptografia em trânsito e em repouso
- Gestão de chaves (KMS/HSM) e rotação
- Controle de acesso (RBAC/ABAC) e segregação por ambiente
- Retenção, descarte, portabilidade e auditoria
- Minimização de dados e mascaramento de logs

1) Criptografia

1.1 Em trânsito

- **TLS 1.3** obrigatório, HSTS + TLS pinning no app.
- Mutual TLS (**mTLS**) para tráfego **intra-serviços** (API ↔ Worker ↔ IA).

1.2 Em repouso

- **AES-256-GCM** para campos sensíveis:
 - `energy_tests.score_vector`
 - `perfil_dominante`, `frase_codigo`, `frequencia_hz`
 - consentimentos, IP, device_id
- **TDE** (Transparent Data Encryption) no cluster de banco.
- **Backups** criptografados (AES-256) com chaves separadas (envelope encryption).

1.3 Gestão de chaves

- **KMS/HSM** (AWS KMS/CloudHSM | GCP KMS) com **envelope encryption**:
 - DEK (Data Encryption Key) rotativa a cada 90 dias.
 - CMK (Customer Master Key) com dual control, rotation anual.
- **Acesso a KMS** via **least privilege** (IAM) e **just-in-time**.

2) Controle de Acesso e Isolamento

Medida	Detalhe
RBAC	Perfis: <code>app-user</code> , <code>support-read</code> , <code>analyst-agg</code> , <code>dev-readonly</code> , <code>dpo</code>
ABAC	Filtros por <code>tenant_id</code> , <code>region</code> , <code>purpose=energy_test</code>
Segregação	Ambientes <code>dev/stage/prod</code> isolados (VPCs, contas cloud distintas)
Tabelas	Leitura de <code>score_vector</code> somente por serviços autorizados (worker, IA, API resultado)
Acesso humano	Exclusivo via break-glass com aprovação dupla (DPO + Security), tudo auditado

3) Logs, Máscara e Observabilidade Segura

- **Não logar** dados brutos de respostas nem `score_vector`.
- **Mascarar**: IP parcial (`xxx.xxx`), `device_id` ofuscado, tokens truncados.
- **PII guard** no pipeline de logs (Data Loss Prevention).
- **Logs de auditoria** em storage WORM (Write Once Read Many) por 2 anos:
 - `api_audit_logs`: rota, user_id, request_id, status, latência
 - `security_events`: acesso negado, falha mTLS, erro KMS
 - `privacy_actions`: exportação/eliminação de dados

4) LGPD/GDPR — Conformidade

4.1 Bases legais e princípios

- **Base:** consentimento explícito (Camada 07) + legítimo interesse para telemetria agregada.
- **Minimização:** coletar apenas dados do teste e metadados estritamente necessários.
- **Finalidade:** personalização do app + evolução do usuário; **proibido** uso publicitário externo.

4.2 Direitos do titular

Endpoints públicos:

- **Portabilidade/Export**

POST /privacy/export

Body: { "user_id": "U123" }

Retorna pacote **JSON + CSV** criptografado (link tempo-limitado, 24h).

- **Eliminação (Right to Erasure)**

POST /privacy/delete

Body: { "user_id": "U123", "confirm": true }

Ações:

- Marca `revogado = true` em `energy_tests`
- Remove PII direto; persiste somente **metadados anonimizados** para métricas
- Gera recibo de eliminação (hash + timestamp)

- **Revogar consentimento**

POST /privacy/consent/revoke

Efeito: bloqueia novas coletas; oferece exclusão dos dados existentes.

4.3 Retenção e descarte

Dado	Retenção padrão	Ação ao vencer
Sessões incompletas	72h	Exclusão automática
Resultados (Free)	12 meses	Notificar para renovar/descartar
Resultados (Premium)	Indefinida	Revisão anual com opção de descarte
Backups	30 dias	Rotação + destruição segura

| Todos os descartes geram evento em `privacy_actions` (auditável).

5) Privacidade por Design (DPIA resumido)

- **Pseudonimização:** chaves internas usam `user_uuid`; evitar e-mail/telefone nas tabelas do teste.

- **Separação lógica:** PII mínima fica em `users` ; resultados do teste em `energy_*`.
- **Acesso condicionado por finalidade** (`purpose=energy_test`).
- **Avaliação de impacto (DPIA)** revisada a cada release maior (v2.x).

6) Segurança Aplicativa (AppSec)

- **Validação de schema** (OpenAPI/JSON Schema) em todas as rotas.
- **CSP** (Content Security Policy) estrita + **SRI** para assets da web.
- **Proteções:** WAF, rate limiting, proteção CSRF onde aplicável (web), anti-replay (`Idempotency-Key`).
- **Secrets:** armazenados em Secret Manager (rotacionar a cada 90 dias).
- **SAST/DAST** em CI/CD; **SBOM** (CycloneDX) por release.
- **Dependency pinning** + renovação semanal (Renovate/Dependabot).

7) Tabelas e Metadados de Privacidade

privacy_actions

Campo	Tipo
<code>action_id</code> (PK)	UUID
<code>user_id</code>	UUID
<code>type</code>	Enum(export, delete, consent_revoke)
<code>status</code>	Enum(queued, done, failed)
<code>hash_receipt</code>	String
<code>created_at</code>	Timestamp

consents_log

Campo	Tipo
<code>consent_id</code> (PK)	UUID
<code>user_id</code>	UUID
<code>leitura</code> / <code>privacidade</code> / <code>emocional</code>	Boolean
<code>ip_mask</code>	String
<code>hash_vibe</code>	String
<code>timestamp</code>	Timestamp

8) Incidentes e Resposta (IR)

- **Deteção:** SIEM + alerts (anomaly, spike 5xx, falha KMS/mTLS).
- **Classificação:** baixo/médio/alto (dados sensíveis expostos = alto).

- **Prazos:** notificação a titulares/autoridade **em até 72h** (GDPR) quando aplicável.
- **Runbooks:** isolar chaves, invalidar tokens, regenerar DEKs, post-mortem com lições.

9) Checklists de Aceite (QA/Segurança)

- ☐ TLS 1.3 verificado, HSTS ativo
- ☐ Criptografia AES-256-GCM nos campos sensíveis
- ☐ Rotação de DEK/CMK configurada (90d/1y)
- ☐ RBAC/ABAC com least privilege
- ☐ Logs sem PII; PII guard validando máscaras
- ☐ Endpoints de **export/delete/revoke** funcionando e auditados
- ☐ Retenções automatizadas e testadas
- ☐ DPIA e política de consentimento publicados
- ☐ Pentest concluído e findings críticos resolvidos

10) Pseudocódigo de Cripto (envelope encryption)

```
function encryptSensitive(payload):
  cmk = KMS.getCMK("energy-prod")
  dek = KMS.generateDataKey(cmk)      # returns {plaintext, ciphertext}
  ciphertext = AES256GCM.encrypt(payload, dek.plaintext, aad="energy_test")
  return { data: ciphertext, dek: dek.ciphertext }

function decryptSensitive(record):
  dek_plain = KMS.decrypt(record.dek)  # requires IAM + mTLS
  return AES256GCM.decrypt(record.data, dek_plain, aad="energy_test")
```

← END Fechamento da Camada

Com esta camada, o módulo do Teste opera sob **padrões de segurança enterprise**, com **privacidade real** (não cosmética) e **conformidade auditável**.

Os devs têm caminhos claros para cripto, acesso, logs, retenção e direitos do titular.

✅ CAMADA 23 — FRONTEND (ESTADO DO TESTE, CACHE OFFLINE, FALLBACKS E TELEMETRIA DE UX)

Objetivo da Camada

Padronizar a implementação de **estado do teste** no app (Flutter/React Native), com:

- **Máquina de estados** confiável,
- **Cache offline** criptografado,
- **Retries/idempotência**,
- **Push/polling** do resultado,
- **Telemetria** de UX desde a 1ª tela.

1) Máquina de Estados do Teste (FE)

Estados-chave (imutáveis, serializáveis):

```
IDLE → CONSENT → SESSION_READY → QUESTIONING  
→ COOLDOWN → SUBMITTING → PROCESSING (202)  
→ RESULT_READY | ERROR
```

Transições críticas:

- `CONSENT.accept()` → cria sessão (`POST /teste/sessao`) → `SESSION_READY`
- `QUESTIONING.answer()` → `POST /teste/resposta` (com Idempotency-Key) → próxima
- `SUBMITTING.finalize()` → `POST /teste/finalizar` → `PROCESSING`
- Notificação push ou polling OK → `RESULT_READY`

Regra: sem botão “voltar” dentro do teste; tudo dirigido pela state machine.

2) Store de Estado (exemplo TS/Flutter)

Shape do estado

```
type TestState = {  
  testId?: string  
  tokenTeste?: string  
  stepIndex: number  
  totalSteps: number  
  status: 'IDLE'|'CONSENT'|'SESSION_READY'|'QUESTIONING'|'COOLDOWN'|'SUBMITTING'|  
  'PROCESSING'|'RESULT_READY'|'ERROR'  
  queue: Array<ClientAnswer> // respostas pendentes/offline  
  lastError?: string  
}
```

Ações principais

```
startSession(consents)
submitAnswer(perguntaId, choiceId, tempoMs)
finalizeTest()
pollResult()
applyPush(payload)
resetTest()
```

Biblioteca sugerida: Zustand/MobX (RN) ou Riverpod (Flutter).

Persistência: storage criptografado (see §4).

3) Idempotência + Retries (cliente)

- Gerar **Idempotency-Key** por resposta (`uuid4()`).
- Retries com **exponential backoff**: 200ms, 500ms, 1s, 2s (máx. 4).
- Se **409 CONFLICT** + mesma Idempotency-Key → considerar enviado (mostrar sucesso).
- Se offline → enfileirar em **queue** e **flush** ao reconectar.

```
async function postWithIdem(url, body){
  const idem = uuid()
  return http.post(url, body, {headers: {'Idempotency-Key': idem}})
}
```

4) Offline-First (cache seguro)

- **Chaves:** respostas em `/cache/test/<testId>/answers` .
- **Criptografia:** AES-256 em client storage (Keychain/Keystore).
- **Política:**
 - Enfileirar **answers** quando offline.
 - Reconexão → flush ordenado por **stepIndex** .
 - Auto-expirar sessão em **24h** (limpar cache).
- **Assets:** imagens/áudios pré-carregados em cache (Service Worker/AssetBundle).

5) Finalização 202 + Push/Polling

- **POST /teste/finalizar** → **202** (sempre).
- UX mostra **"sintonizando..."** (animação leve).
- **Preferir Push** (Firebase/APNs). Payload mínimo: `{ test_id, ready:true }` .

- **Fallback Polling:**
 - Intervalo: 2s (até 5 tentativas), depois 5s (até 6 tentativas). Máx. ~40s.
 - Se `404/425` : continuar polling; se `500` : backoff e banner de erro suave.
- Ao receber pronto → `GET /teste/resultado/:test_id` → render.

6) Fallbacks de UX

Falha	Tratamento
Perda de rede no meio da pergunta	Salvar local, banner “Sem internet, continuando offline”
<code>POST /resposta</code> falha	Re-tentar automatic; se esgotar → manter na queue
<code>POST /finalizar</code> falha	Mostrar botão “Tentar novamente”; manter sessão válida
Resultado demora > 60s	CTA “me avise quando estiver pronto” (push) + voltar ao app
Asset de áudio/imagem falha	Fallback de categoria (ex.: 432Hz, imagem neutra)

7) Performance (budget FE)

- **TTI alvo** (primeira tela do teste): < **1.5s** (p95).
- **Asset por pergunta:** < **1MB** total (img + áudio).
- **Pré-carregar** 2 próximas perguntas em background.
- **Memória:** descartar assets antigos a cada 2 passos.
- **Animações:** 60fps (usar Lottie/SVG, evitar bitmaps pesados).

8) Acessibilidade

- Texto alternativo para imagens; **transcrições** para sons.
- Controles acessíveis (focus, tamanho hit area ≥ 44px).
- Suporte a **screen readers** e tema alto contraste.
- Redução de movimento (respeitar “Reduce Motion” do SO).

9) Telemetria de UX (eventos)

Emitir (com `request_id` e `test_id`):

- `view_consent`, `consent_accepted`
- `session_created`
- `question_view` (id, ordem)
- `answer_submitted` (tempo_ms, idem_key, retry_count)
- `finalize_clicked`
- `result_processing_view`

- `push_received` | `poll_result_ok`
- `result_viewed`
- `error_shown` (código, recoverable?)

Enviar via SDK analítico (Mixpanel/GA4) + log técnico (observabilidade).

10) QA Checklist (FE)

- ☐ State machine cobre todos os caminhos (incl. offline/retomada)
- ☐ Filas de respostas persistem criptografadas e esvaziam ao reconectar
- ☐ Idempotency-Key por resposta; 409 tratado como sucesso
- ☐ Finalização sempre retorna 202 e entra em `PROCESSING`
- ☐ Push atualiza de imediato; polling funciona como fallback
- ☐ Fallbacks de assets (áudio/imagem) operacionais
- ☐ A11y validada (screen reader, contrast, reduce motion)
- ☐ Telemetria dispara em todos os marcos (sem PII)

11) Pseudocódigo de Fluxo (resumido)

```
await startSession(consents) // → SESSION_READY
for (let q of questions) {
  show(q) // → QUESTIONING
  const ans = await collect()
  await submitAnswer(q.id, ans.choiceld, ans.ms) // fila/offline+idem
}
await finalizeTest() // → PROCESSING (202)
await waitPushOrPoll()
renderResult()
```

← END Fechamento da Camada

Com esta camada, o app fica **robusto no mundo real**: sem travar no 3G, resiliente a quedas de conexão, e com **telemetria** rica para evoluir a UX.

✅ CAMADA 24 — DESIGN SYSTEM & GUIDES (TOKENS, COMPONENTES, ESTADOS, MOTION & DARK MODE)

Objetivo da Camada

Definir o **Design System técnico-sensorial** do módulo do Teste, cobrindo:

- Tokens de cor, tipografia e espaçamento,
- Componentes de UI (básicos e específicos do teste),
- Estados de interação,
- Motion/animizações,
- Temas claro/escuro.

1) Tokens Fundamentais

Categoria	Token	Valor
Cores primárias	color.solar	#FFD966 (amarelo dourado)
	color.lunar	#9E7CF4 (roxo/lilás)
	color.neutral	#6F6F6F
Feedback	color.success	#4CAF50
	color.error	#F44336
	color.warning	#FF9800
Backgrounds	bg.dark	#0F0F12
	bg.light	#FFFFFF
Tipografia	font.primary	Inter / system
	font.mono	Roboto Mono
Espaçamento	space.xs	4px
	space.sm	8px
	space.md	16px
	space.lg	24px
	space.xl	32px
Raios	radius.sm	8px
	radius.md	16px
	radius.lg	24px

2) Componentes Reutilizáveis

Botões

- `Button.Primary` → cor vibracional dinâmica (solar/lunar)
- `Button.Secondary` → outline + gradiente suave
- Estados: normal, hover, pressed, disabled

Inputs

- `ChoiceCard` → imagens/sons/frases do teste
- Estados: default, hovered, selected, error

Layouts

- `Screen.Consent` → checkboxes + CTA
 - `Screen.Question` → ChoiceCard + header de progresso
 - `Screen.Result` → fractal animado + vetores + CTA "Ativar Jornada"
-

3) Estados e Feedback de Interação

- **Hover** (desktop) → leve elevação (shadow-sm)
 - **Pressed** → redução de 5% da escala (tap feedback)
 - **Disabled** → opacidade 50% + cursor bloqueado
 - **Loading** → spinner sutil ou esqueleto no ChoiceCard
-

4) Motion & Animações

- **Transição entre perguntas** → fade-in/out (300ms)
- **Seleção de escolha** → pulso energético (scale 1.05 → 1.0, 250ms)
- **Resultado** → fractal crescendo + partículas suaves (800ms)
- **Modo sombra detectado** → fundo escurece levemente (overlay 20%)

Lottie/SVG preferidos; assets bitmap devem ser comprimidos.

5) Dark Mode & Tema Adaptativo

- **Default:** Dark Mode (bg escuro, cores vibracionais em destaque).
 - **Light Mode:** backgrounds claros, mesmos tokens vibracionais.
 - **Deteção automática:** seguir configuração do SO.
 - **Switch manual:** disponível em configurações.
-

6) Guidelines para Dev/Design

- **Consistência cross-platform:** usar tokens sempre (não hardcode).
 - **Responsividade:** ChoiceCard ≥ 44px de altura (touch target).
 - **Internacionalização:** todos os textos via i18n.
 - **Acessibilidade:** contraste mínimo 4.5:1; animações desabilitáveis em "reduce motion".
-



Fechamento da Camada

Com essa camada, o módulo do Teste opera em cima de um **Design System unificado**, permitindo consistência visual, acessibilidade e personalização sensorial em qualquer plataforma.

✓ CAMADA 25 — PERFIL ENERGÉTICO COMPLETO (MAPA, FRASE-CÓDIGO, POLARIDADE, FREQUÊNCIA)

🎯 Objetivo da Camada

Gerar um **retrato vibracional completo** do usuário após o teste, combinando:

- Vetores normalizados,
- Frequência Hz e polaridade,
- Frase-código personalizada,
- Arquétipo dominante/ secundário,
- Visualização gráfica em mapa energético.

Esse perfil é a **identidade energética ativa** do usuário no FriendApp.

1) Estrutura do Perfil Energético

Campo	Tipo	Descrição
perfil_dominante	String	Arquétipo principal (ex.: Guardião Solar)
perfil_secundario	String/null	Híbrido ou transição
frequencia_hz	Float	Valor vibracional do momento
polaridade	Enum	solar, lunar, neutra
vetores	JSONB	Pontuação em 7 eixos (foco, visão, presença, força, fluxo, sensibilidade, sombra)
frase_codigo	String	Frase vibracional da IA
mapa_vibracional	Blob/ref	Imagem fractal dinâmica
resonance_uuid	String	Identificador único da assinatura

2) Vetores Energéticos (7 eixos)

Exemplo de saída após normalização:

```
{
  "foco": 0.74,
  "visao": 0.68,
  "presenca": 0.82,
```

```
"forca": 0.61,  
"fluxo": 0.47,  
"sensibilidade": 0.59,  
"sombra": 0.37  
}
```

3) Cálculo de Frequência e Polaridade

- **Frequência Hz** → média ponderada dos vetores luminosos, penalizada pela sombra:

```
frequencia = (mean(foco, visao, presenca, forca, fluxo, sensibilidade) * (1 - sombra*0.15)) *  
10
```

- **Polaridade** → derivada de `lum - sombra` (Camada 16):
 - +0.08 → solar
 - < -0.08 → lunar
 - entre → neutra

4) Frase-Código Vibracional

Gerada pela IA Aurah Kosmos a partir do vetor e perfil atribuído.

Exemplo:

```
{  
  "perfil": "Guardião Solar",  
  "frase_codigo": "Você é chama que aquece e guia, mas lembre-se de repousar."  
}
```

Sempre curta, simbólica e validada eticamente (sem termos negativos explícitos).

5) Mapa Vibracional

- **Visualização:** gráfico circular de 7 eixos, renderizado em WebGL ou SVG animado.
- **Cor:** adaptada à polaridade (solar = dourado, lunar = lilás, neutra = cinza/verde).
- **Animação:** pulsação leve sincronizada com `frequencia_hz`.
- **Export Premium:** usuário pode baixar imagem fractal personalizada.

6) Persistência em Banco

Tabela `energy_profiles_active`

Campo	Tipo
<code>user_id</code>	UUID (PK)
<code>perfil_dominante</code>	String
<code>perfil_secundario</code>	String
<code>frequencia_hz</code>	Float
<code>polaridade</code>	Enum
<code>vetores</code>	JSONB
<code>frase_codigo</code>	String
<code>mapa_ref</code>	String (path blob/CDN)
<code>resonance_uuid</code>	String
<code>updated_at</code>	Timestamp

7) Integrações do Perfil

- **Feed Sensorial** → injeta tags do arquétipo e polaridade.
- **Mapa de Frequência Global** → atualiza ponto coletivo.
- **Jogo da Transmutação** → define trilha inicial.
- **Conexões** → recalcula matches compatíveis.
- **Aurah Kosmos** → armazena assinatura para acompanhamento evolutivo.

← END Fechamento da Camada

Essa camada consolida tudo: **dados técnicos, experiência visual e integração prática.**

O Perfil Energético Completo é o **pilar da identidade do usuário dentro do FriendApp**, baseando todas as ativações futuras.

✅ CAMADA 26 — CONEXÃO COM A IA AURAH KOSMOS (PÓS-TESTE, LEITURA E EVOLUÇÃO DO PERFIL)

🎯 Objetivo da Camada

Definir como o resultado do Teste de Personalidade Energética é entregue à **IA Aurah Kosmos**, que passa a:

- Interpretar continuamente os vetores energéticos,
- Aprender padrões de evolução do usuário,
- Personalizar conteúdos e conexões,

- Ajustar a jornada em tempo real conforme feedbacks e histórico.

1) Ativações Automáticas Pós-Teste

Assim que o resultado é consolidado:

- IA recebe payload completo (`perfil` , `vetores` , `frequencia` , `polaridade` , `sombra`).
- Gera **assinatura vibracional** (`resonance_uuid`) para acompanhamento.
- Salva no seu **grafo energético interno** para detectar padrões coletivos.

2) Estrutura do Payload para IA

```
{
  "user_id": "U123",
  "perfil_dominante": "Guardião Solar",
  "perfil_secundario": "Oráculo Lunar",
  "frequencia_hz": 9.28,
  "polaridade": "solar",
  "vetores": {
    "foco": 0.74,
    "visao": 0.68,
    "presenca": 0.82,
    "forca": 0.61,
    "fluxo": 0.47,
    "sensibilidade": 0.59,
    "sombra": 0.37
  },
  "flags": {
    "sombra_ativa": false,
    "transicao": false
  },
  "resonance_uuid": "8fa32b90-23e4-4f87-95e1-99a9a6f2c5de",
  "timestamp": "2025-09-04T20:10:00Z"
}
```

3) Responsabilidades da IA Aurah

Função	Descrição
Aprendizado contínuo	Reajusta pesos da matriz com base em feedbacks dos usuários
Contexto dinâmico	Ajusta tom das mensagens no chat/jogo/feed de acordo com estado atual
Predição de ciclos	Detecta se usuário está entrando em expansão, colapso ou transição
Feedback adaptativo	Sugere micro-calibrações ou trilhas de autocuidado
Integração coletiva	Usa dados anonimizados para atualizar o Mapa de Frequência Global

4) Estados da IA no Pós-Teste

- **Ativo** → acompanha usuário normalmente.
- **Cuidadoso** → se sombra moderada detectada, reduz intensidade de sugestões.
- **Cuidadoso+** → se sombra alta, ativa linguagem de acolhimento e limita estímulos.
- **Transição** → se perfil indefinido, acompanha com perguntas abertas e feed exploratório.

5) Painel Interno (Aurah Insights)

Campos exibidos em dashboards técnicos e de IA:

- Último `perfil_dominante` atribuído.
- Score de confiança e nível de sombra.
- Histórico de vetores nas últimas 3 leituras.
- Recomendações em andamento (feed, missões, conexões).
- Alertas de inconsistência (autoengano, manipulação, incoerência).

6) Persistência

Tabela: `aurah_context`

Campo	Tipo
<code>user_id</code>	UUID
<code>resonance_uuid</code>	String
<code>perfil_atual</code>	String
<code>frequencia_hz</code>	Float
<code>polaridade</code>	Enum
<code>shadow_index</code>	Float
<code>ia_mode</code>	Enum (normal, safe, safe+)
<code>last_update</code>	Timestamp

7) UX para o Usuário

- Aurah envia uma **mensagem inicial pós-teste**, ex.:

“Você despertou como Guardião Solar. A partir de agora vou caminhar ao seu lado, ajustando seu espaço no FriendApp de acordo com a sua energia.”
- Interações seguintes adaptadas ao perfil (ex.: motivacionais para solar, introspectivas para lunar).
- Usuários Premium podem visualizar **relatórios vibracionais** gerados pela Aurah (histórico + tendência).

← **Fechamento da Camada**

A Camada 26 conecta o **perfil energético do usuário** à **IA Aurah Kosmos**, transformando o resultado em um **sistema vivo e adaptativo**, que acompanha, protege e expande a experiência ao longo do tempo.

✓ **CAMADA 27 — SISTEMA DE ATUALIZAÇÃO DE PERFIL ENERGÉTICO (RETESTE, EVOLUÇÃO E VERSÃO ATIVA)**

🎯 **Objetivo da Camada**

Permitir que o usuário atualize seu **perfil energético** de forma cíclica e inteligente, através de:

- **Reteste completo** (novo teste integral),
- **Micro-calibrações** (curtas, acionadas por feedback ou inconsistência),
- **Reavaliações automáticas** (ciclos de tempo ou mudanças abruptas),
- **Gestão de versões** (manter histórico e marcar a versão ativa).

1) Modos de Atualização

Tipo	Gatilho	Frequência Permitida	Efeito
Reteste Completo	Solicitação manual do usuário	1x a cada 60 dias	Novo perfil substitui versão ativa
Micro-Calibração	Feedback “não me representou” ou IA detecta incoerência	Até 1x/semana	Ajusta vetores e recalibra perfil ativo
Reavaliação Automática	Ciclo de 90 dias ou delta vibracional abrupto (>20%)	Automática	Notificação para refazer o teste
Emergencial	Evento crítico detectado (ex.: sombra alta súbita)	Imediata	IA dispara nova leitura guiada

2) Estrutura de Versões

Tabela: **energy_profile_versions**

Campo	Tipo	Descrição
version_id	UUID	Identificador da versão
user_id	UUID	FK
perfil_dominante	String	Arquétipo
perfil_secundario	String/null	Híbrido ou vazio

Campo	Tipo	Descrição
<code>frequencia_hz</code>	Float	Frequência medida
<code>polaridade</code>	Enum	solar, lunar, neutra
<code>vetores</code>	JSONB	Score nos 7 eixos
<code>frase_codigo</code>	String	Frase da IA
<code>ativa</code>	Boolean	Se é a versão atual
<code>created_at</code>	Timestamp	Data da geração

| Apenas 1 versão ativa por usuário em qualquer momento.

3) Fluxo de Reteste

```
function solicitarReteste(user_id):
  if diasDesdeUltimoTeste < 60:
    return "Ainda cedo para novo ciclo."
  nova_sessao = criarNovaSessao(user_id)
  return nova_sessao
```

- Nova sessão é criada com `status=started`.
- Quando concluída → gera `version_id` novo.
- Marca versão anterior como `ativa=false`.
- Nova versão substitui perfil atual e dispara ativações.

4) Micro-Calibrações

- Gatilhos:
 - Usuário responde “não me representou” (Camada 06).
 - IA detecta incoerência > 15% entre comportamento no app e perfil ativo.
- Fluxo:
 - 2 a 3 perguntas rápidas (imagem, som ou frase).
 - Atualizam vetores incrementais.
 - Ajustam **frase-código** ou refinam `perfil_secundario`.

5) Reavaliação Automática

- Executada pelo worker em lote:
 - Verifica usuários cujo último teste > 90 dias.
 - Verifica delta vibracional abrupto no histórico (`delta_hist > 0.22`).

- Notificação push:

| "Sua energia está em mudança. Vamos atualizar sua frequência?"

6) Emergenciais

- Detectadas por IA Aurah (sombra alta ou colapso).
 - Usuário convidado a um reteste imediato.
 - Modo cuidadoso ativo até conclusão.
-

7) Persistência e Auditoria

- Cada versão fica registrada em `energy_profile_versions`.
 - Apenas `ativa=true` é usada nos sistemas (feed, jogo, conexões).
 - Histórico completo disponível para Premium.
 - Logs de atualização em `profile_update_logs` (com `motivo` e `gatilho`).
-

8) UX do Usuário

- Tela de **Histórico Vibracional** (Premium): gráfico mostrando evolução de frequência Hz, polaridade e arquétipo ao longo do tempo.
 - Badge especial:
 - "Recalibrado" → após 2 retestes.
 - "Em Transição" → quando perfil indefinido.
 - "Nova Frequência" → quando delta > 20%.
-

Fechamento da Camada

Com essa camada, o perfil energético deixa de ser estático e passa a ser **um organismo vivo**: evolui, é recalibrado, e se adapta às mudanças internas do usuário.

Isso garante que o FriendApp acompanhe **a vida real** e nunca entregue resultados ultrapassados.

CAMADA 28 — SISTEMA DE SOMBRAS OCULTAS & TRAVAS ENERGÉTICAS (RASTREAMENTO, ALERTAS E TRANSMUTAÇÃO)

Objetivo da Camada

Identificar **travas internas invisíveis** durante e após o teste, classificá-las em **tipologias**, acionar **alertas internos** na IA Aurah e propor **caminhos de transmutação** para liberar o usuário desses bloqueios.

1) Fontes de Detecção

Fonte	Sinal Oculto	Exemplo
Tempo de resposta	Hesitação ou pressa excessiva	<300ms (impulsividade) ou >10s (evasão)
Repetição de escolhas	Padrão viciado sem contexto	Sempre escolhendo "isolamento"
Incoerência semântica	Respostas contraditórias	"Quero conexão" + frases de fuga
Desvios emocionais	Tokens de linguagem defensiva	"Tanto faz", "Não importa"
Histórico vibracional	Queda súbita de frequência	$\Delta > 20\%$ entre versões
Engajamento no app	Falta de resposta às ativações	Ignora missões/ feed por > 14 dias

2) Índice de Trava (lock_index)

```
lock_index = 0.3*(tempo_anormal)
            + 0.2*(repeticao_score)
            + 0.2*(incoerencia_score)
            + 0.15*(tokens_defensivos)
            + 0.15*(delta_hist)
```

lock_index $\in [0,1]$

- **Leve:** 0.3–0.49
- **Moderada:** 0.5–0.69
- **Alta:** ≥ 0.7

3) Tipologia das Travas Ocultas

Código	Nome	Características	Impacto
T-IMP	Impulsividade	Tempo de resposta muito baixo	Foco distorcido
T-HES	Hesitação	Demora exagerada para responder	Perda de fluxo
T-EVA	Evasão	Tokens de fuga: "não sei", "tanto faz"	Bloqueio emocional
T-REP	Repetição	Escolhas idênticas sem contexto	Estagnação
T-COL	Colapso	Queda brusca de frequência no histórico	Risco vibracional alto

4) Ações do Sistema

- **Leve** → IA apenas registra; feed mostra conteúdos suaves.

- **Moderada** → IA ativa **modo cuidadoso**, sugere **micro-calibração** em 48h.
- **Alta** → IA ativa **modo cuidadoso+**, reduz estímulos no feed, libera apenas **missões curtas de cura** no jogo, e sugere reteste emergencial em até 24h.

5) Integração com Aurah Kosmos

- Aurah salva `lock_index` e tipologia em `aurah_context`.
- IA envia **mensagens adaptativas**:
 | “Sinto que sua energia está hesitante. Vamos com calma, em pequenos passos.”
- **Feed** → mostra conteúdos de estabilização.
- **Conexões** → prioriza perfis acolhedores, evitando confrontos vibracionais.

6) Persistência em Banco

Tabela: `energy_locks`

Campo	Tipo
<code>lock_id</code>	UUID
<code>user_id</code>	UUID
<code>test_id</code>	UUID
<code>lock_index</code>	Float
<code>primary_code</code>	Enum (T-IMP, T-HES, T-EVA, T-REP, T-COL)
<code>secondary_code</code>	Enum opcional
<code>state</code>	Enum(none, light, moderate, high, resolving)
<code>created_at</code>	Timestamp

7) Fluxo de Resolução

```

if lock_index < 0.3:
    state = none
elif 0.3 <= lock_index < 0.5:
    state = light
elif 0.5 <= lock_index < 0.7:
    state = moderate
else:
    state = high

on missao_cura_concluida or micro_calibracao_ok:
    lock_index -= 0.1
    if lock_index < 0.3:

```

state = resolving → resolved

8) Telemetria & Observabilidade

Métricas monitoradas:

- % de usuários em cada tipo de trava.
- Tempo médio até resolução.
- Impacto no engajamento (feed/jogo).
- Correlação entre travas ocultas e taxa de abandono.

← END Fechamento da Camada

Essa camada garante que até os **bloqueios invisíveis** sejam detectados, tratados e guiados à cura.

O sistema não apenas diagnostica, mas acompanha e libera o usuário, mantendo a experiência **segura, empática e contínua**.

✅ CAMADA 29 — TRILHAS EVOLUTIVAS PERSONALIZADAS (CAMINHOS DE TRANSFORMAÇÃO PÓS-TESTE)

🎯 Objetivo da Camada

Transformar o resultado do teste em **caminhos práticos de evolução**, criando **trilhas personalizadas** de acordo com perfil, sombra e tendência vibracional.

As trilhas ajudam o usuário a **transmutar bloqueios**, **expandir potenciais** e **cultivar equilíbrio energético**.

1) Estrutura das Trilhas

Cada trilha contém:

- **Blocos Temáticos** → coragem, silêncio, fluxo, cura, equilíbrio.
- **Missões Sequenciais** → desafios curtos (5–10min).
- **Recompensas** → XP vibracional, selos, desbloqueios no jogo.
- **Feedback IA** → adaptação conforme progresso.

2) Tipos de Trilhas

Tipo	Indicador de Entrada	Foco Evolutivo
Trilha de Expansão	Perfil solar / tendência de crescimento	Ação, coragem, liderança
Trilha de Cura	Sombra moderada/alta detectada	Autocuidado, aterramento
Trilha de Potencialização	Perfil híbrido (dominante+secundário)	Integrar dons secundários
Trilha de Transição	Sem perfil dominante claro	Autodescoberta, clareza

3) Fluxo de Ativação

```
ao_finalizar_teste(user_id, resultado):
  if sombra > 0.5:
    trilha = "cura"
  elif perfil_dominante and perfil_secundario:
    trilha = "potencializacao"
  elif resultado.flags.transicao:
    trilha = "transicao"
  else:
    trilha = "expansao"
  iniciarTrilha(user_id, trilha)
```

4) Exemplo de Bloco de Trilha

```
{
  "trilha": "cura",
  "bloco": "Aterramento",
  "missao": "Respire por 3 minutos ouvindo 432Hz",
  "recompensa": { "xp": 10, "selo": "Raiz" },
  "tempo_estimado": "5min"
}
```

5) Integração com o Jogo da Transmutação

- Trilhas alimentam **missões diárias** do jogo.
- XP da trilha = XP do jogo.
- Conclusão de trilha desbloqueia **novos totens vibracionais**.
- Usuários em **Trilha de Cura** → missões mais curtas (limite 1/dia).

6) Persistência em Banco

Tabela: `evolution_tracks`

Campo	Tipo
track_id	UUID
user_id	UUID
track_type	Enum (expansao, cura, potencializacao, transicao)
status	Enum (ativa, pausada, concluida)
etapa_atual	FK → track_steps
xp_acumulado	Int
created_at	Timestamp

7) Telemetria e Observação

- % de usuários em cada tipo de trilha.
- Taxa de conclusão de blocos.
- Impacto no shadow_index após trilhas de cura.
- Tempo médio em cada trilha.

8) UX do Usuário

- Tela **"Minha Jornada"** → exibe trilha ativa, progresso e próximas missões.
- Notificações push:
 - "Sua trilha de Cura tem uma nova missão disponível."
 - "Continue sua trilha de Expansão: um desafio de coragem aguarda você."
- Usuários Premium → acesso ao **histórico de trilhas concluídas**.

← END Fechamento da Camada

Com essa camada, o teste deixa de ser um diagnóstico e vira **um caminho prático de evolução**.

Cada usuário recebe uma trilha sob medida, que se adapta à sua energia e mantém o engajamento ativo no ecossistema.

✅ CAMADA 30 — DESBLOQUEIOS NO FEED, CONEXÕES, JOGO E MAPA DE FREQUÊNCIA

🎯 Objetivo da Camada

Definir como os resultados do teste geram **desbloqueios imediatos e personalizados** no ecossistema, garantindo que o usuário veja efeitos reais no app após concluir sua leitura vibracional.

1) Tipos de Desbloqueio

Área	Condição	Desbloqueio
Feed Sensorial	Perfil dominante atribuído	Conteúdos energéticos compatíveis liberados
Jogo da Transmutação	Perfil + sombra	Trilha inicial e missão de desbloqueio ativadas
Mapa de Frequência	Frequência Hz > 0	Ponto vibracional do usuário aparece no mapa
Conexões Autênticas	Perfil validado	Sugestões de matches vibracionais compatíveis

2) Fluxo de Ativação

```
onResultadoPronto(user_id, resultado):  
  if resultado.perfil_dominante:  
    desbloquearFeed(user_id, resultado.perfil_dominante)  
    iniciarJogo(user_id, resultado.perfil_dominante, resultado.sombra)  
    atualizarMapa(user_id, resultado.frequencia_hz, resultado.polaridade)  
    recalcularConexoes(user_id, resultado.perfil_dominante, resultado.sombra)
```

3) Experiência do Usuário

- Ao concluir o teste, usuário vê tela de resultado → botão **“Ativar Minha Jornada”**.
- Clicar inicia os desbloqueios:
 - Feed mostra posts compatíveis com o perfil.
 - Jogo inicia trilha energética personalizada.
 - Mapa exhibe o ponto vibracional recém-ativado.
 - Conexões são recalculadas e sugeridas.
- Usuário recebe **notificação sensorial**:

“Sua energia foi desbloqueada. Explore seu feed, jornada e novas conexões.”

4) Persistência dos Desbloqueios

Tabela: `energy_unlocks`

Campo	Tipo	Descrição
<code>unlock_id</code>	UUID	Identificador
<code>user_id</code>	UUID	Usuário dono
<code>test_id</code>	UUID	Origem do desbloqueio
<code>area</code>	Enum(feed, jogo, mapa, conexoes)	Onde foi aplicado
<code>status</code>	Enum(ativo, falhou, revertido)	Estado

Campo	Tipo	Descrição
timestamp	Timestamp	Momento da ativação

5) Regras de Falha e Retry

- Se feed não atualizar → retry automático 3x em 30s.
- Se jogo não iniciar → fallback para trilha genérica "Autodescoberta".
- Se mapa falhar → sincroniza em segundo plano.
- Se conexões não recalcularem → IA dispara recalibração em até 1h.

6) Telemetria

- % de desbloqueios aplicados com sucesso.
- Tempo médio entre finalização do teste e desbloqueio (< 1s ideal).
- Impacto nos primeiros 7 dias:
 - Posts consumidos no feed.
 - Missões concluídas no jogo.
 - Matches aceitos.
 - Acessos ao mapa.

Fechamento da Camada

Essa camada mostra que o Teste não é só diagnóstico: ele **abre portais dentro do FriendApp**.

Cada desbloqueio é imediato, tangível e conectado à energia real do usuário, garantindo engajamento e sentido prático à experiência.

CAMADA 31 — SISTEMA DE LOGS, SEGURANÇA, CONSENTIMENTO E RASTREAMENTO VIBRACIONAL

Objetivo da Camada

Garantir que todo o ciclo do Teste de Personalidade Energética seja:

- **Auditável** (logs completos e imutáveis),
- **Seguro** (dados criptografados e acessos controlados),
- **Ético** (consentimento explícito e rastreável),

- Conforme LGPD/GDPR (direitos do usuário assegurados).

1) Tipos de Logs

Tipo	Conteúdo	Retenção
API Audit	Rota chamada, <code>user_id</code> , <code>request_id</code> , IP mascarado, latência, status	2 anos
Eventos Vibracionais	Vetores consolidados, <code>perfil_dominante</code> , <code>shadow_index</code> , <code>lock_index</code>	18 meses
Consentimento	Aceites de termos (leitura, privacidade, emocional), timestamp, hash de sessão	5 anos
Privacy Actions	Exportação, exclusão, revogação de consentimento	5 anos
Worker Processing	Eventos de fila, latência de IA, worker_id	12 meses

2) Estrutura de Tabelas

api_audit_logs

Campo	Tipo
<code>log_id</code>	UUID
<code>user_id</code>	UUID
<code>endpoint</code>	String
<code>status_code</code>	Int
<code>latencia_ms</code>	Int
<code>ip_mask</code>	String
<code>request_id</code>	String
<code>timestamp</code>	Timestamp

vibration_events

Campo	Tipo
<code>event_id</code>	UUID
<code>user_id</code>	UUID
<code>test_id</code>	UUID
<code>vetores</code>	JSONB
<code>perfil_dominante</code>	String
<code>shadow_index</code>	Float
<code>lock_index</code>	Float
<code>created_at</code>	Timestamp

privacy_actions

Campo	Tipo
<code>action_id</code>	UUID
<code>user_id</code>	UUID
<code>tipo</code>	Enum(export, delete, consent_revoke)
<code>status</code>	Enum(queued, done, failed)
<code>hash_receipt</code>	String
<code>created_at</code>	Timestamp

3) Consentimento e Rastreamento

- Cada teste exige aceite explícito dos 3 termos (Camada 07).
- Consentimentos são armazenados em `consents_log` com hash criptográfico.
- Cada vetor vibracional tem seu `resonance_uuid` único → usado em todas as integrações para rastreabilidade.
- Logs conectam `resonance_uuid` → `ativacoes` → `trilhas/jogo` → **ciclo completo rastreável**.

4) Segurança

- **Criptografia:** AES-256-GCM nos campos sensíveis.
- **Chaves:** KMS/HSM com rotação a cada 90 dias.
- **Acesso:** RBAC mínimo necessário, break-glass com dupla aprovação.
- **Logs:** armazenados em storage imutável (WORM).
- **PII guard:** pipeline bloqueia dados sensíveis em logs (emails, telefone, respostas textuais).

5) Rastreamento Vibracional

- Cada execução gera:
 - Vetores (7 eixos)
 - Índices (`shadow_index` , `lock_index`)
 - Polaridade e frequência
 - Assinatura `resonance_uuid`
- Esses dados alimentam:
 - **Mapa de Frequência Global** (coletivo, anônimo)
 - **Aurah Kosmos** (IA evolutiva)
 - **Trilhas de evolução** (Camada 29)

6) LGPD/GDPR — Direitos do Usuário

- **Exportação** → pacote JSON+CSV com vetores, perfis, histórico.
 - **Exclusão** → apaga PII e marca dados como anonimizados.
 - **Revogação de consentimento** → bloqueia futuras coletas.
 - **Auditoria** → cada ação gera recibo com hash + timestamp em `privacy_actions`.
-

7) Observabilidade

- Painéis (Grafana/Prometheus):
 - Volume de testes/dia.
 - Latência API p95/p99.
 - Shadow/lock index médios.
 - Taxa de exportações/exclusões LGPD.
 - Alertas:
 - Falhas repetidas na fila.
 - Exceções em auditoria.
 - Tentativa de acesso não autorizado.
-

Fechamento da Camada

Com essa camada, o Teste se torna **totalmente rastreável, seguro e ético**:

cada clique, resposta e decisão da IA é registrado, criptografado e auditável, assegurando transparência ao usuário e robustez para os devs.

CAMADA 32 — GLOSSÁRIO TÉCNICO-VIBRACIONAL DO TESTE (TERMOS, CÓDIGOS E CONCEITOS)

Objetivo da Camada

Unificar a terminologia usada no Teste de Personalidade Energética, padronizando:

- **Nomes e códigos internos** (usados em banco, APIs, logs, IA).
 - **Definições claras** de cada termo vibracional.
 - **Símbolos e representações visuais**.
 - **Evitar ambiguidades** entre áreas (UX, backend, IA, analytics).
-

1) Vetores Energéticos

Termo	Código	Definição
Foco	<code>vetor_foco</code>	Clareza mental, direção da energia.
Visão	<code>vetor_visao</code>	Amplitude, percepção do todo.
Presença	<code>vetor_presenca</code>	Capacidade de estar no aqui e agora.
Força	<code>vetor_forca</code>	Determinação e ação prática.
Fluxo	<code>vetor_fluxo</code>	Capacidade de soltar e seguir o curso.
Sensibilidade	<code>vetor_sensibilidade</code>	Receptividade emocional.
Sombra	<code>vetor_sombra</code>	Bloqueios e resistências inconscientes.

2) Índices e Scores

Termo	Código	Definição
Índice de Sombra	<code>shadow_index</code>	Score $\in [0,1]$ que mede intensidade de bloqueio.
Índice de Trava	<code>lock_index</code>	Score $\in [0,1]$ que mede travas ocultas.
Score Vetorial	<code>score_vector</code>	Vetor final normalizado (7 eixos).
Confiança de Perfil	<code>conf_dominante</code>	Probabilidade de acerto do perfil dominante.

3) Perfis Energéticos

Termo	Código	Definição
Perfil Dominante	<code>perfil_dominante</code>	Arquétipo principal do usuário.
Perfil Secundário	<code>perfil_secundario</code>	Arquétipo complementar (híbrido).
Modo Transição	<code>flag_transicao</code>	Estado onde nenhum perfil tem confiança suficiente.

4) Frequência e Polaridade

Termo	Código	Definição
Frequência Vibracional	<code>frequencia_hz</code>	Valor Hz calculado dos vetores.
Polaridade Solar	<code>polaridade=solar</code>	Energia ativa, expansiva.
Polaridade Lunar	<code>polaridade=lunar</code>	Energia introspectiva, receptiva.
Polaridade Neutra	<code>polaridade=neutra</code>	Estado de equilíbrio.





5) Frase-Código e Símbolos

Termo	Código	Definição
Frase-Código	<code>frase_codigo</code>	Mensagem curta da IA que sintetiza o estado energético.
Mapa Vibracional	<code>mapa_vibracional</code>	Gráfico circular animado dos vetores.
Assinatura Vibracional	<code>resonance_uuid</code>	Hash único que identifica leitura vibracional.

6) Estados de IA

Termo	Código	Definição
Modo Normal	ia_mode=normal	IA ativa em sua linguagem padrão.
Modo Cuidadoso	ia_mode=safe	IA reduz estímulos, tom acolhedor.
Modo Cuidadoso+	ia_mode=safe+	IA limita interações e só mostra conteúdos de autocuidado.

7) Símbolos Visuais

Símbolo	Significado
	Expansão solar.
	Introspecção lunar.
	Polaridade neutra.
	Sombra atuante.
	Estado de transição.
	Cura/Regeneração.

8) Padrões de Log

Exemplo de log vibracional:

```
{
  "event": "resultado_gerado",
  "user_id": "U123",
  "test_id": "T-9841",
  "perfil_dominante": "Guardião Solar",
  "conf_dominante": 0.81,
  "frequencia_hz": 9.28,
  "polaridade": "solar",
  "shadow_index": 0.37,
  "resonance_uuid": "a3f7-...",
  "timestamp": "2025-09-04T21:30:00Z"
}
```

← **Fechamento da Camada**

Com este glossário, todo o ecossistema FriendApp fala a **mesma língua técnica e vibracional**.

Não importa se é backend, frontend, IA ou UX: todos os termos estão definidos, codificados e livres de ambiguidade.

✓ CAMADA 33 — INTEGRAÇÕES E DEPENDÊNCIAS CÍCLICAS DO ECOSISTEMA (ENTRADAS, SAÍDAS, GATILHOS, LEITURAS)

🎯 Objetivo da Camada

Descrever **como o Teste conversa com o restante do FriendApp**:

- Quais sistemas **alimentam** o teste,
- Quais sistemas ele **entrega dados**,
- Quais **gatilhos** são disparados,
- E como se forma um **ciclo contínuo de feedback** no ecossistema.

1) Entradas para o Teste

- **Cadastro Consciente** → ativa a sessão inicial do teste após perguntas vibracionais básicas.
- **Aurah Kosmos** → fornece contexto histórico e perfil anterior.
- **Banco de Perguntas & Choices** → alimenta os estímulos visuais, auditivos e frases.
- **Configurações de IA (matriz)** → versão atual da Matriz de Ponderação Energética usada para o cálculo.

2) Saídas do Teste

- **Perfil Energético Completo** → alimenta IA, feed, jogo, conexões, mapa.
- **Assinatura Vibracional (`resonance_uid`)** → chave única para rastreamento em logs e IA.
- **Triggers de desbloqueio** → feed, jogo, mapa, conexões.
- **Relatórios para o usuário Premium** → histórico vibracional, relatórios da IA.

3) Gatilhos Técnicos Pós-Teste

Sistema	Gatilho	Descrição
Feed Sensorial	<code>feed.injectTags(perfil)</code>	Injeta tags vibracionais no algoritmo do feed.
Jogo da Transmutação	<code>jogo.start(trilha)</code>	Inicia trilha do jogo baseada no perfil.
Mapa de Frequência	<code>mapa.update(coord)</code>	Posiciona o usuário no mapa global.
Conexões Autênticas	<code>conexoes.rebuild(perfil)</code>	Reconstrói matches com base no arquétipo.
Aurah Kosmos	<code>aurah.sync(resultado)</code>	IA passa a acompanhar evolução do usuário.
RA / Experiências Imersivas	<code>ra.enableSymbols</code>	Ativa fractais e símbolos em RA (Premium).

4) Dependências Cíclicas

- O **teste gera dados** que **alimentam a IA Aurah**,
- A IA **aprende com o usuário** e **ajusta a matriz** para futuros testes,
- O resultado **desbloqueia jogo, feed e conexões**,
- O engajamento nesses sistemas gera **novos sinais**,
- Esses sinais são **registrados e enviados de volta** para IA + histórico,
- O ciclo fecha quando o usuário refaz o teste → incorporando toda a evolução no caminho.

5) Observabilidade e Rastreamento

- **Logs** → cada integração gera evento em `integration_logs`.
- **Métricas chave:**
 - Tempo de propagação de gatilhos.
 - % de integrações concluídas com sucesso.
 - Correlação entre `shadow_index` e uso do feed/jogo.
- **Tracing** → cada `resonance_uuid` rastreia ciclo completo (teste → IA → feed → conexões → histórico → reteste).

6) Representação Cíclica

flowchart TD
Cadastro → Teste
Teste → AurahKosmos
Teste → Feed
Teste → Jogo
Teste → Mapa
Teste → Conexoes
AurahKosmos → Matriz
Feed → Logs
Jogo → Logs
Conexoes → Logs
Logs → Historico
Historico → Reteste
Reteste → Teste

← END Fechamento da Camada

O Teste de Personalidade Energética não é uma funcionalidade isolada: ele é o **coração cíclico do FriendApp**.

Cada execução cria um fluxo de entrada, processamento e saída que alimenta todo o ecossistema, fecha o ciclo de aprendizado e prepara o usuário para a próxima evolução.