# PRISM Cognitive Profiling System – Technical Overview and Architecture

## 1. Scoring Architecture

**Flow of Scoring:** PRISM scoring runs as an Edge Function (score\_prism) that transforms a session’s raw question responses into a cognitive profile. When a user finishes an assessment session, the finalizeAssessment function invokes score\_prism with the session ID[[1]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L119-L128). The scoring function fetches all responses for that session from assessment\_responses, deduplicates by question (keeping the latest answer)[[2]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L209-L219), and then applies the scoring key mappings. Each question is mapped to one or more trait indicators via the **Assessment Scoring Key** table. For example, Likert-scale “strength” items carry tags like Ti\_S, Ne\_S etc., indicating they measure a particular cognitive function’s strength[[3]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L70-L78)[[4]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L100-L108). These responses are normalized to a common 1–5 scale (accounting for reversed items) and aggregated per trait: all \_S items for a function are averaged to compute that function’s **strength** (1–5 scale)[[5]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L462-L470)[[6]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L543-L552). Separately, “dimensional” items (tagged with \_D) measure the *experience level* or dimensionality of each function. PRISM averages each function’s \_D responses and maps that average to a 1–4 **dimension** level using thresholds (e.g. avg ≤2.1 ⇒ 1D, ≤3.0 ⇒ 2D, ≤3.8 ⇒ 3D, else 4D)[[7]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L20-L23)[[8]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L544-L553). This yields a strengths vector (Ti, Te, Fi, Fe, Ni, Ne, Si, Se strengths) and a dimensions vector (Ti–Se dimensions).

**Versioned Scoring Functions:** The scoring engine is modular and versioned. PRISM *v1.1.x* introduced reliability weighting and confidence calibration, while *v1.2.0* (“Phase 3/4”) further unified these and added refined validity gates[[9]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L711-L719)[[10]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L890-L899). The Edge Function code is annotated with the version (e.g. “PRISM v1.2.0 Enhanced Scoring Engine – Phase 3”[[11]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1-L9)) and uses a results\_version flag in the scoring\_config table to toggle behaviors[[12]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L249-L258). Different versions primarily adjust how Likert and forced-choice data are blended and how confidence is computed. For example, v1.1.2 introduced z-score blending of Likert vs. Forced-Choice responses (improving method agreement) and isotonic regression calibration for confidence[[13]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L8-L16)[[14]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L18-L26). In v1.2.0, these calibrations were unified and made always-on, and new fields like conf\_raw, conf\_calibrated, and conf\_band were added to profiles[[15]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L17). The system logs its active version and will respond with a maintenance status if an incompatible version or disabled state is set in the scoring\_config (via system\_status)[[16]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L264-L272).

**Trait & Type Computation:** After aggregating trait scores, PRISM assigns a personality **type** using Socionics’ 16-type Model A. Each type (e.g. LIE, IEI, etc.) is defined by a **prototype** – an assignment of each of the 8 functions to Model A positions (base, creative, role, etc.)[[17]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L16-L25). The scoring engine evaluates all 16 types by summing weighted trait scores according to the type’s prototype: for each function f, if the prototype places f in a strong position (e.g. Base or Creative), the function’s strength contributes with a higher weight; if in a weaker position (e.g. Vulnerable, Ignoring), it contributes minimally or not at all[[18]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L604-L613)[[19]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L600-L608). The weight constants for each block are defined (Base=1.00, Creative=0.70, … Vulnerable=0.10, etc.)[[20]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L39-L48). Thus, a type’s raw score scoreType(type) is essentially a weighted sum of the user’s 8 function strengths, plus: (a) a small bonus for higher dimensionality on that function (functions with 3D/4D get up to 10% extra on their weight)[[21]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L609-L617), (b) a small bonus for evidence from forced-choice questions (FC support, described below)[[21]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L609-L617), and (c) a penalty if the user’s *opposite* functions are stronger than the type’s valued functions (e.g. if a type’s Base is Ti, but the user’s Fe > Ti by >0.5, subtract a penalty)[[22]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L580-L589)[[23]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L586-L595). This opposite-function penalty helps disqualify types where the user shows a strong contrary orientation (e.g. strong Fe would penalize Ti-base types)[[22]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L580-L589)[[24]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L591-L599). The engine also enforces Model A consistency: it skips any type whose base–creative function pairing is invalid (ensuring, for example, that if Base is a Perceiving function, Creative must be one of its allowed partners)[[25]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L636-L645).

**Forced-Choice and Overlays:** PRISM uses two kinds of questions: Likert-scale self-assessments (discussed above) and **Forced-Choice (FC)** scenario questions. Forced-choice items ask the user to pick among options that correspond to either cognitive function usage or cognitive “block” preference. Each FC question has an fc\_map in the scoring key mapping options (A, B, C, D…) to either a function or a functional block[[26]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L375-L382). For example, a 4-option scenario might map A→“Core”, B→“Critic”, C→“Instinct”, D→“Hidden” – the four Model A blocks[[27]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L180-L188). Another FC might map choices directly to functions (e.g. A→Ni, B→Ne, C→Si, D→Fe, etc.)[[28]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L166-L175). The scoring function tallies these: each time an option is selected, it increments either the corresponding block count or function count[[26]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L375-L382)[[29]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L377-L385). From this it derives a **FC support** vector per function: essentially the proportion of FC scenarios in which each function (or its block) was favored[[30]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L625-L634)[[31]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L626-L634). This FC support (fcSupport[f]) enters the type scoring as a small additive term, weighted by the same block weights (up to 10% bonus)[[21]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L609-L617). The net effect is that if a user’s choices consistently favor certain blocks or functions, types containing those as Base/Creative get a slight boost.

**Raw Score to Result:** The outcome of the above is a raw “fit” score for each type. PRISM first converts the weighted sum scores into a 0–100 **Fit** percentage. It computes an “ideal” target profile for each type (the prototype’s block weights mapped onto a 1–5 scale) and measures the Euclidean distance between the user’s function strengths and that prototype[[32]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L756-L765)[[33]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L773-L782). This distance is normalized and inverted to a 0–100 scale called *Absolute Fit*[[33]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L773-L782)[[34]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L783-L786). In practice, raw fit scores range ~0–6.5 before scaling, which maps linearly to 0–100%[[35]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L788-L796)[[36]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L796-L800). The top type is the one with highest fit%. The engine doesn’t finalize the type immediately, though: it also calculates a *relative likelihood* distribution. It applies a softmax over the raw type scores to get a **Share %** for each type (i.e. probability-like distribution summing to 100%)[[37]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L839-L847)[[38]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L841-L844). The **Top-3 types** by fit are identified, and tie-breakers are applied if fit scores are very close (considering share%, dimensional “coherence”, etc.)[[39]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L853-L862)[[40]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L865-L873). Finally, the chosen type code, along with detailed metrics, is saved to the profiles table for that session[[41]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1022-L1031)[[42]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1033-L1041).

**Overlays (Neuroticism & State):** Beyond the 16 type codes, PRISM appends “overlay” markers to reflect secondary traits. The two overlays are **Neuroticism** and **State**. The assessment includes a Neuroticism index (24 Likert questions, alternating normal/reversed scoring, tagged N or N\_R[[43]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L139-L147)[[44]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L140-L148)). The user’s average Neuroticism score (z-scored against cohort norms) determines the **Neuroticism Overlay**: if the user is at least 0.5σ above the mean in N, they get a “+” overlay; if 0.5σ below, a “–” (otherwise “0”)[[45]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L668-L676)[[46]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L670-L678). This “+” or “–” essentially flags if the individual is more anxious/reactive vs. more stable than average. Separately, PRISM assesses the user’s momentary **State** (stress, time pressure, sleep, focus) via designated questions. A weighted index of those state ratings is computed[[47]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L674-L683). If the user reports high stress/time or poor sleep/focus (index beyond cut), they get a “+” state overlay (meaning an agitated state during test); if unusually low stress & well-rested (index below –cut), a “–” overlay[[48]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L678-L685). In the current UI, the **primary overlay** shown is neuroticism (the overlay field is set equal to overlay\_neuro for backward compatibility)[[49]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L682-L690)[[50]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L16-L24). Thus, a final type might be displayed as “LIE+” meaning the LIE type with a high Neuroticism overlay. (The state overlay “±” is also stored and could be shown separately – e.g. in admin views – but historically only one overlay is appended in the public results[[50]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L16-L24)[[51]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L38-L42).) These overlays enrich the type interpretation by indicating affective temperament and test conditions.

**Session Validity & Confidence:** The scoring architecture also generates indices to gauge profile quality. It records **attention check failures** and pairs of inconsistent answers. Certain question pairs (defined by pair\_group tags) measure the same concept in reverse; large discrepancies raise an inconsistency index[[52]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L444-L452). A social-desirability index is computed from any items flagged social\_desirability=true (these are special agree/disagree items where consistently giving “ideal” answers could imply less candor)[[53]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L20-L28). The engine classifies each profile’s **validity\_status** as “pass”, “warning”, or “fail” based on thresholds for inconsistency, response variance (standard deviation index), and attention checks[[54]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L696-L705)[[9]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L711-L719). It also computes a **confidence level** (High/Moderate/Low) which is tied to validity: for example, any fail-level validity issue yields Low confidence, one warning-level issue yields Moderate, etc.[[55]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L713-L721)[[56]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L715-L723). This “confidence” essentially tells the user how reliable their result is likely to be. PRISM v1.1.2+ further refines confidence by calibrating it against retest outcome data – effectively using isotonic regression on historical retest hit rates to map confidence to a calibrated probability that the top type will remain the same on retest[[13]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L8-L16)[[57]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L22). The profile stores both raw and calibrated confidence scores (as percentage odds) and a final confidence band[[57]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L22)[[58]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L59-L67). Overall, the scoring output includes rich metadata: the top type and its fit/share, the next two types, trait scores, all 16 type fit scores, overlays, and confidence diagnostics[[41]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1022-L1031)[[59]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1074-L1082). This data is returned to the caller and saved in the profiles table[[60]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1089-L1098)[[61]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1116-L1124).

## 2. Psychometric Foundations

**Theoretical Grounding:** PRISM’s architecture is grounded in **Socionics** (a Jungian-derived typology) combined with elements of the **Big Five**. The 16 cognitive types in PRISM correspond to Socionics types, each defined by an information metabolism model (“Model A”). The eight cognitive **functions** measured – Ti, Te, Fi, Fe, Ni, Ne, Si, Se – are the standard Jungian functions (logical/ethical, introverted/extroverted, etc.)[[62]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L8-L13). Socionics arranges these in a hierarchy of eight functional *positions* (Base, Creative, Role, Vulnerable, etc.), which PRISM uses as the template for scoring[[63]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L16-L24). For example, type **LIE** (Logical Intuitive Extrovert, roughly analogous to ENTj) has Te as Base and Ni as Creative by definition[[63]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L16-L24). PRISM’s block weight scheme (Base=1.0, Creative=0.7, etc.) reflects the presumed strength of each position in Model A[[20]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L39-L48), meaning the instrument assumes a Base function manifests most strongly, Creative second, etc. This theoretical model drives how raw responses map to type: a user scoring high on Te and Ni (and lower on their opposites Fi/Se) will naturally score well as LIE[[64]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L600-L609)[[65]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L619-L627). The built-in check for valid Base–Creative pairs also comes from Socionic intertype structure (only certain function combinations are considered a consistent ego block)[[66]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L636-L644).

In addition to these cognitive function scales, PRISM includes a measure of **Neuroticism**, one of the Big Five personality factors. Neuroticism (sometimes labeled Emotional Stability when reversed) was incorporated because it captures a key aspect of temperament – how prone someone is to stress, worry, and mood swings – which is largely independent of cognitive information-processing style. The Neuroticism questions in PRISM were adapted from standard Big Five inventories (with about 12 positive and 12 reverse-scored items)[[67]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L156-L165)[[68]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L172-L180). The resulting Neuroticism score is used both for the overlay (± marker) and in analysis of confidence (users with very high neuroticism might have more variable self-perceptions). While other Big Five traits (Extraversion, Openness, Agreeableness, Conscientiousness) are not explicitly measured, some aspects may correlate with the Socionic factors. For instance, Socionics types have information elements that roughly map to MBTI (which itself correlates with Big Five to some extent), but PRISM’s focus is on the cognitive axis rather than broad traits like Extraversion. **Socionics vs MBTI:** Notably, PRISM’s type codes (e.g. LII, SLE, etc.) are Socionics notation. These can be translated to MBTI-like four-letter codes, but PRISM emphasizes functional makeup over MBTI dichotomies. This allows more nuance (e.g. distinguishing two types that MBTI would label the same, based on which function is base vs creative).

**Core Traits Definitions:** The core of PRISM is the eight **function strength** scores. Each reflects how strongly the respondent self-reports behaviors or preferences aligned with that function’s domain. For example, **Si** (Introverted Sensing) items might involve attention to physical comfort or past experience; high average on those six Likert items yields a strong Si score[[69]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L106-L114)[[70]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L112-L120). The **dimensionality** ratings (1D to 4D) for each function come from Socionics’ concept of functional dimensionality (how experienced or conscious a function is). PRISM approximates this via questions about *confidence and scope* in using the function – e.g. a 4-dimensional function is used broadly, unconsciously, and confidently, whereas a 1D function feels awkward or is avoided. The output dimension level for each function (stored in dimensions) thus enriches the profile by indicating not just *how much* they use a function, but how *comfortable/adept* they are with it[[71]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L552-L561)[[72]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L559-L567). For instance, two people might both score “moderately high Ne,” but one might be 4D Ne (creative, natural brainstormer) and the other 2D Ne (learned to use it but not innate).

**Overlays and Axes:** The **Neuroticism overlay** (“+” or “–”) denotes an emotional stability modifier on the type. A “+” (high neuroticism) suggests the person may exhibit more anxiety or reactivity in their cognitive style, whereas “–” indicates a calmer, more stable affective style[[46]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L670-L678). This overlay is conceptually drawn from Big Five Neuroticism (high N corresponds to “+”). The **State overlay** (“+” or “–”) is meant to flag if the test results might be situationally skewed – e.g. “+” if the user was in a high-stress, sleep-deprived state, which could make their answers more erratic or pessimistic than usual[[73]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L678-L686). In practice, PRISM has used state overlays in research to study *state-trait separation* – ensuring that momentary stress (state) doesn’t overly alter the stable trait scores. The system logs a “state\_index” from -1 to +1 roughly, and the overlay is a coarse summary of that[[47]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L674-L683). (The front-end may not explicitly display the state overlay in the user-facing results, focusing on the neuro overlay as the main “extra” symbol[[50]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L16-L24), but state is recorded and can be used in analyses of consistency.)

**Reliability and Calibration:** The psychometric soundness of PRISM is monitored through various methods. There are database **views for test-retest reliability** that correlate function strength vectors between a user’s two sessions[[74]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L4-L13)[[75]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L26-L34). For example, view v\_test\_retest\_strength\_r calculates the Pearson correlation of the 8-dimension strength profile for each pair of sessions a user has, and also checks if their top type stayed the same[[76]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L30-L38). Internal studies (like the “First 100 Assessments” analysis) have been conducted to ensure that the function scales have acceptable internal consistency and that retest correlations are reasonably high (PRISM’s documentation targeted improving Likert vs FC method agreement to >0.60 correlation after v1.1.2 improvements[[14]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L18-L26)[[77]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L20-L27)). PRISM also explicitly tracks **type stability:** in the retest view, it flags whether the top-1 type remained identical and whether any “adjacent” type swap occurred (e.g. first and second swapped)[[78]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L32-L40). These metrics feed back into confidence calibration: empirically, profiles with “High” confidence labels should correspond to high probability of stable type on retest. Indeed, a v\_calibration\_confidence view groups completed retests by the initial test’s confidence band and computes the hit rate – the fraction of time the same type reappeared – to validate the calibration[[79]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L45-L54)[[80]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L51-L59). This data was used to adjust the mapping of confidence% to High/Moderate/Low so that, for instance, “High” confidence roughly means ~80% chance of same type on retest, etc.[[13]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L8-L16)[[81]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L10-L18).

To ensure trait calibration, PRISM also looks at score distributions cohort-wide. For example, after each scoring run, it pulls recent profiles to compute the mean and SD of raw fit scores in the last 90 days, and uses that to z-score an individual’s fit against population[[82]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L801-L809)[[83]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L812-L820). The calibrated fit (score\_fit\_calibrated) places the user’s top-type fit on a T-scale (mean 50, SD 15) bounded to 20–85 for stability[[84]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L820-L828)[[85]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L824-L832). This helps users interpret their fit as “weak/average/strong” relative to others. All these calibration steps (fit and confidence) are periodically retrained or checked. The system has a calibration\_model table and a train\_confidence\_calibration function for updating the isotonic mapping when enough new data accumulates[[57]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L22)[[58]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L59-L67). In summary, PRISM’s psychometric design is a fusion of a Jungian cognitive model with a Big Five emotional stability lens. It employs data-driven calibration to make the results as interpretable and stable as possible, and it actively measures its own reliability (e.g. method agreement indices, retest correlations) to guide ongoing improvements[[14]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L18-L26)[[86]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L24-L27).

## 3. Supabase Backend Integration

**Data Model – Tables:** The backend uses a PostgreSQL (Supabase) database with relational tables for sessions, responses, and results. Key tables include:

* **assessment\_sessions** – one row per test session (fields: id UUID PK, user\_id (if logged in; null for guest), started\_at, completed\_at, status, etc.)[[87]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L2-L10)[[88]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L16-L24). A session is created when a user begins the assessment. The status is updated as they progress (e.g. from 'in\_progress' to 'completed')[[89]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L10-L18)[[90]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L20-L28). There’s a unique index to ensure a given email can only have one active in-progress session (to prevent parallel duplicates)[[91]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L14-L19). Sessions also carry a share\_token (a UUID) set upon completion, used to securely share/view results without authentication[[92]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L172-L180)[[93]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L174-L182).
* **assessment\_responses** – records each answer. Columns include session\_id (FK to session), question\_id (an integer or code identifying the item), the answer\_value (text) and an answer\_numeric if applicable, plus timestamps[[94]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L16-L25). To simplify scoring, question text and type are also stored here at insert time (denormalized)[[95]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L18-L26). During scoring, these responses are joined with the scoring key to map them to traits[[96]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L69-L72)[[97]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L106-L114). RLS policies allow users (or at least the service role) to insert and select their responses. In early implementation, selection was broad (anyone could read – since there was no auth yet, it was effectively open)[[98]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L64-L69). Later RLS was hardened to restrict reads (likely only via safe views or with appropriate user match once login was introduced).
* **assessment\_scoring\_key** – defines how each question contributes to scoring. Each row has a question\_id (matching responses), a tag (like Ti\_S, N\_R, etc.), a scale\_type (enum: e.g. LIKERT\_1\_5, FORCED\_CHOICE\_4, META, etc.), a reverse\_scored flag, an optional fc\_map JSON (for FC questions) and other metadata[[99]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L18-L26)[[100]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L66-L74). This is essentially the answer key for scoring. The key is loaded in the scoring function (skey query) and cached in a map by question ID for lookups[[101]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L222-L231)[[102]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L233-L240). Because the assessment content has evolved, migrations have updated this table in place (e.g. a recent migration rebuilt all entries with new question numbering and mappings)[[103]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L1-L9)[[104]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L29-L37). The design allows the scoring logic to remain generic – it just iterates over answers and uses the key to decide what to do with each (e.g. add to likert[f], increment a block count, etc.)[[105]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L361-L370)[[106]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L371-L379).
* **profiles** – stores the computed results for each session (essentially the output of score\_prism). There’s a one-to-one relationship: profiles.session\_id = session UUID. The profile contains columns for the type code, the two letter functions (base\_func, creative\_func), the overlay(s), and numerous JSONB fields for detailed results (e.g. strengths JSON mapping each function to its 1–5 score, dimensions JSON for each function’s 1–4 level, trait\_scores for any additional trait like N)[[41]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1022-L1031)[[107]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1073-L1080). It also stores scalar fields for key summary metrics: score\_fit\_raw, score\_fit\_calibrated (the user’s top type fit % before and after cohort calibration), fit\_band (High/Moderate/Low fit strength band), confidence (High/Moderate/Low), conf\_raw and conf\_calibrated (numerical confidence scores), close\_call (boolean if top 2 are close), top\_gap (difference between first and second fit)[[42]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1033-L1041)[[108]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1034-L1042). The JSON fields type\_scores (containing each type’s fit\_abs and share\_pct) and top\_types (array of the top 3 type codes) are also saved for richer client display[[109]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L869-L878)[[110]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L866-L874). The profile table is the main source for retrieving results after the fact (so that scoring does not have to be re-run on each view). On scoring, an upsert is done: if a profile already exists for that session, it’s updated (with a recomputed\_at timestamp preserved)[[111]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L50-L59)[[112]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L62-L70). Otherwise a new row is inserted, stamping submitted\_at to now[[113]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L56-L61)[[114]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1103-L1111). (This allows rescoring if the algorithm updates or if partial data was filled in later.)
* **Other tables:** A scoring\_config key-value table is used to store various parameters and feature flags (e.g. dim\_thresholds, neuro\_norms, softmax\_temp, etc.)[[115]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L250-L258)[[116]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L260-L268). This enables tweaking algorithm constants via DB without code changes – for instance, adjusting the dimension mapping thresholds or turning off calibration. There are also tables like session\_method\_metrics (storing the z-scored Likert vs FC values per session for analysis of method agreement)[[117]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L22-L26)[[86]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L24-L27) and potentially calibration\_model for confidence calibration mapping[[15]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L17)[[58]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L59-L67). Additionally, some “analytics” or meta-views are backed by tables (e.g. storing function usage distribution for research). The schema also includes reference tables for static data (question text, etc.), but these are often exposed as *views* combining the scoring key and actual question content.

**Supabase Edge Functions:** The system uses Supabase Edge Functions (serverless Deno functions) for critical backend logic. The primary ones:

* **score\_prism** – as described, it encapsulates the scoring algorithm. It reads the session’s responses and scoring key, computes the profile, and writes to profiles. It returns a JSON with status and the profile data[[118]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1134-L1142)[[119]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1136-L1144). This function runs with service-role access (so it can read/write all needed tables ignoring RLS)[[120]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L4-L13)[[121]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L6-L8). It includes robust error handling for partial sessions, maintenance mode, etc., ensuring that if something is wrong (e.g. no responses yet, or scoring disabled), it returns a specific status string[[122]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L196-L204)[[123]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L396-L404).
* **finalizeAssessment** – this function orchestrates session completion. It is called by the frontend when a user submits the final question. It takes the session ID (and optionally the responses, though in practice it now just uses session ID)[[124]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L32-L40)[[125]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L46-L54). finalizeAssessment first checks if a profile already exists (meaning the session was already scored, perhaps user double-submitted)[[126]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L50-L58). If profile exists, it simply marks the session completed and returns the existing profile and a results URL[[127]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L56-L64)[[128]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L82-L90). If not, it fetches the session, then calls score\_prism via supabase.functions.invoke[[1]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L119-L128). It handles any error or maintenance response from scoring (propagating an error status if needed)[[129]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L123-L131)[[130]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L139-L147). On success, it updates assessment\_sessions to status='completed', sets a share\_token (for results sharing), and records the number of questions answered[[92]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L172-L180). It then returns a payload to the client containing the profile data and a constructed results\_url that includes the session ID and share token as query param[[131]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L216-L224). This function ensures the session record and profile stay in sync and is the single point that transitions a session to completed.
* **get-results-by-session** – a protected function to retrieve results for viewing. This function requires a session\_id and the correct share\_token[[132]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L16-L24). It checks the session’s status and token; if the token matches or the session is already completed/public, it will fetch the profile and return it[[133]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L62-L70)[[134]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L99-L107). This is used by the results page to securely fetch the profile without exposing the entire profiles table to the client. It enforces that without a valid token, an in-progress session’s results cannot be accessed (HTTP 403)[[133]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L62-L70)[[135]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L68-L76). In essence, this function acts as a gatekeeper: it looks up the session (ensuring the ID format is valid UUID to prevent SQL injection)[[136]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L24-L32), verifies completion or token, then returns the latest profile row for that session[[134]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L99-L107). Notably, if a profile somehow didn’t exist (which should not happen after finalize runs), it will attempt to call finalizeAssessment itself as a fallback (this can handle edge cases where a results link is accessed before finalize completed)[[137]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L116-L125)[[138]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L120-L128). The presence of this function means the front-end doesn’t directly read profiles via the anon key; it goes through a controlled endpoint with the share token check.
* (Other functions include things like start\_assessment (for creating a new session record), notify\_admin (to send an email or log when someone completes an assessment)[[139]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L70-L78)[[140]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L203-L211), and possibly calibration or analytics triggers. For example, notify\_admin is called in finalizeAssessment fire-and-forget to ping admin on new completions[[139]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L70-L78).)

**Views and Security:** The database defines several **views** to simplify queries and enforce RLS (Row-Level Security). For instance, there is a view v\_sessions that combines session info with response counts and infers a status label[[141]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L2-L10)[[142]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L12-L20). It joins assessment\_sessions with profiles and aggregates from assessment\_responses to list how many responses were given, first/last answer timestamps, etc., and outputs a status as 'completed', 'in\_progress', or 'started'[[143]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L14-L22)[[144]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L16-L24). This view helps in admin dashboards to quickly see session progress. Another set of views (often suffixed \_v11 for version) exist for analytics/KPIs, e.g. v\_kpi\_overview\_30d\_v11 which shows counts and averages over last 30 days of profiles (like number of assessments, average fit, % high confidence, etc.)[[145]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L44-L52)[[146]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L54-L62), or v\_latest\_assessments\_v11 which lists recent results with key fields (timestamp, type code, overlay, fit, share of top type, etc.)[[147]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L12-L20)[[148]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L22-L30). These views often filter by results\_version (e.g. only include v1.x results in certain analyses to exclude future v2 data)[[149]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L32-L40).

Crucially, **RLS policies** are applied such that the front-end can only read from certain safe views, not the raw tables. For example, the migration that added v\_profiles\_ext granted SELECT on that view to anon and authenticated roles[[150]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L62-L68). v\_profiles\_ext is basically SELECT p.\*, COALESCE(p.overlay\_neuro, p.overlay) as overlay\_compat FROM profiles p – exposing all profile columns plus a unified overlay field[[151]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L36-L43). By granting access to this view and not to profiles directly, we control what sensitive info is exposed. (The view omits anything considered sensitive; the main PII would be user\_id, which is in profiles – but since user\_id is a UUID, it’s arguably not easily identifying by itself. Nonetheless, it’s something to monitor with RLS.) Similarly, an assessment\_questions\_view likely joins questions with scoring keys for client consumption, and it’s whitelisted via a callable function (getView) rather than direct table access[[152]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/getView/index.ts#L16-L24)[[153]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/getView/index.ts#L58-L66). Initially, RLS was permissive (to allow prototyping without login) – e.g. policies allowed any SELECT on assessment\_sessions/responses[[154]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L46-L55)[[98]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L64-L69). Over time, these were tightened as auth was added. The script preflight.ts includes an RLS test: it tries to select a session with anon key to ensure it’s forbidden (expect 401/403) and verifies only the service role can read certain config tables[[155]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L87-L96)[[156]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L99-L107). This indicates in CI they enforce that anon cannot arbitrarily read protected tables.

**Error Handling and Integrity:** The backend migrations include various fixes to maintain data integrity. For instance, a migration added status column to sessions with allowed values and default[[89]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L10-L18), to ensure all sessions have a known state (in\_progress/completed). Another added session\_kind and parent\_session\_id to profiles to track if a profile is an initial test or a retest of a previous session[[157]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L2-L7). This helps with linking sessions for test-retest analysis and ensuring only one “initial” profile per user unless they explicitly redo. The system also uses triggers for timestamp updates (e.g. an update\_updated\_at\_column() trigger to auto-set updated\_at on updates)[[158]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L71-L79)[[159]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L80-L84) on key tables so that data freshness is tracked.

During scoring, several potential error conditions are checked: If no responses are found but the session is marked partial, it returns a status indicating “No responses yet – continue assessment” rather than scoring[[122]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L196-L204). If the session hasn’t answered enough forced-choice questions to be reliable (by default, fewer than 50% of expected FC items), it returns a "partial\_insufficient" status prompting for more data[[123]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L396-L404)[[160]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L399-L407). These statuses are handled on the client to possibly resume the test instead of showing results. After scoring, the functions carefully handle upserts – preserving original submission timestamp if recalculating so that the created\_at doesn’t all cluster at re-run time[[114]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1103-L1111)[[161]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1104-L1112). They log events for unusual occurrences (e.g. if no valid type combination was found – which theoretically shouldn’t happen except extreme input – it logs and falls back to a guess based on top two functions)[[162]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L650-L658).

Finally, migrating from one version to another is done via SQL migrations (like the one that renumbered questions and rebuilt scoring key on 2025-09-03). These migrations illustrate how the system can “re-score” or adjust on the fly: in that particular migration, they delete all existing scoring key entries and reinsert them with new mappings and question ranges[[103]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L1-L9)[[104]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L29-L37), ensuring consistency. They likely accompany such big changes with either code gating (e.g. a results\_version switch to not break active sessions) or precompute transitional data. In practice, after that migration, the front-end would load the new question set and any old sessions would have had the old results stored already.

In summary, the Supabase backend ties everything together: it stores the assessment content and user data, executes the scoring logic securely, and exposes results through controlled channels (either via secure views or via token-protected functions). The design emphasizes **consistency** (with transactions or careful ordering so that a session marked completed always has a corresponding profile) and **security** (using share tokens and RLS to ensure only authorized access to personal results).

## 4. Frontend Integration (Lovable Frontend)

On the front-end, PRISM has a dynamic results UI that consumes the Supabase data and presents it in a user-friendly way. Key integration points:

**Result Page Workflow:** When a user finishes the assessment, the client calls the finalizeAssessment function via Supabase’s JS client. The response contains profile data and a results\_url[[163]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L84-L92)[[164]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L86-L90). The client can then redirect the user to the /results/{session\_id}?token={share\_token} page. On the results page, a React component will either use the profile data already obtained (if carried in state/navigation) or call the get-results-by-session function to fetch it (supplying the session\_id and token). The project includes a getResultsBySession API wrapper in the front-end for this purpose. The token in the URL is used by the Edge Function to authorize access[[133]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L62-L70), as described earlier. This way, even if the user refreshes the results page or shares the link, the front-end can retrieve the profile securely.

**Components for Results Display:** The results page is implemented with modular React components such as ResultsV2, TraitPanel, OverlayChips, etc. The **ResultsV2 component** is the main container for the new results design (version 2 of the results page). It pulls the profile data and renders various sections: top type summary, top-3 comparison, trait bars, overlays, narrative, etc. For example, ResultsV2.tsx defines constants for thresholds to visually highlight certain aspects (like marking any function that is ≥3D as a highlight, or tagging strengths ≤2.4 as “suppressed”)[[165]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L11-L19). It also defines a list of functions and a Profile TypeScript type that mirrors the backend profile structure[[166]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L115-L123)[[167]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L119-L127). This type includes fields like type\_code, base\_func, creative\_func, overlay, strengths, dimensions, blocks, blocks\_norm, neuroticism (with raw\_mean and z), validity (inconsistency and sd\_index), confidence (High/Moderate/Low), plus the calibrated fields and version[[167]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L119-L127)[[168]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L129-L137). By typing the profile, the component can easily access e.g. profile.strengths.Ti or iterate FUNCS to generate a bar chart.

Various subcomponents handle specific parts of the UI: - **Type and Fit Display:** For instance, FitInfo in ResultsV2 shows an explanatory tooltip for the fit score, including what counts as strong vs weak fit and when “Close Call” badge appears (when topGap < 5)[[169]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L36-L44)[[170]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L38-L46). It uses the profile.top\_gap and profile.close\_call fields to decide if a badge should be shown[[171]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L31-L39)[[172]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L51-L59). - **Top-3 Fit Chart:** The Top3FitChart component renders a bar chart comparing the fit % and share % of the top three types[[173]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L60-L68)[[174]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L72-L80). It uses the profile.top\_types list (or a processed equivalent) to get codes and their fit/share values, and highlights the primary type in a darker color[[175]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L64-L72)[[176]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L70-L78). The chart includes reference lines at 35, 55, 75 to visually band weak/moderate/strong fits, with a legend explaining these ranges[[177]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L92-L100)[[178]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L102-L110). All these thresholds align with the calibration notes (e.g. fit ≥75 is very strong prototype match)[[170]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L38-L46)[[179]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L40-L48).

* **Trait Panels:** The UI likely has a section per cognitive function showing the strength and dimension. A TraitPanel component (not excerpted above, but implied by naming) would take each function (Ti, Te, etc.) and display its numeric strength (perhaps as a bar or dial) and its dimension (1D–4D). The LABEL\_THRESH.dimHighlight = 3 in ResultsV2 suggests they visually mark functions at 3D or 4D as “coherent strengths”[[165]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L11-L19) – indeed likely labeling those as core competencies. Conversely, functions with strength below ~2.4 might be faded or labeled “suppressed” indicating the user reported rarely using them[[180]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L12-L19). The Profile type in the code confirms strengths and dimensions are accessible for this purpose[[181]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L119-L125).
* **Overlay Chips:** If the UI displays overlays, there may be an OverlayChips component that reads profile.overlay (neuro overlay) and possibly profile.overlay\_state. It could render small “+ Neuroticism” or “– Neuroticism” badges, and similarly for state if desired. In earlier versions, the overlay was just a single character appended to type (like “IEE–”); in the new UI they might break it out as a chip with explanation. The StateLegend component imported in ResultsV2[[182]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L8-L16) suggests they do present something related to state, possibly explaining what the “state” overlay means in a legend.
* **Validity/Confidence UI:** The results page also communicates the confidence level. It might display a message if confidence is Low (warning that results may be less reliable). According to the Roadmap, “confidenceDistribution” and achieving ≥70% High/Moderate is a metric, so the UI likely shows a badge or text like “Confidence: High”. The confidence field is directly in profile[[183]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L124-L132). Also, since v1.1.2, conf\_calibrated is a percentage (like probability) – the UI might show something like “Confidence: High (≈85%)”. The Roadmap’s updated results plan included tooltips explaining calibration[[184]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L89-L97)[[185]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L94-L102). ResultsV2 indeed has a section for Fit that includes an InfoTip about how to read fit and share[[170]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L38-L46) – likely similarly they provide info for confidence.

**API Client Logic:** The front-end uses the Supabase JS client for both real-time and HTTP calls. For edge functions, it likely uses supabase.functions.invoke or a wrapper around it. From the code, finalizeAssessment is probably called via something like supabase.functions.invoke('finalizeAssessment', { body: { session\_id, responses } }). In fact, in AssessmentComplete.tsx or AssessmentForm.tsx, after the last question, they call finalize and handle the result. The search result shows AssessmentComplete.tsx, which likely uses the returned results\_url – possibly providing a button “View Results” that navigates to it. The share token mechanism means a user can copy the URL and someone else with the link can view results. The front-end doesn’t directly filter by user\_id or such; it relies on the token check in the function.

**Versioning and Rendering Logic:** Because the system may evolve (v2.x), the UI has to handle older profiles gracefully. For example, older profiles (v1.0 or v1.1) might not have type\_scores or share\_pct fields. The code references some fields with fallback logic. In v\_latest\_assessments\_v11 view, it derives share\_pct as (p.type\_scores->(p.top\_types->>0)->>'share\_pct')::numeric[[186]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L18-L25)[[187]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L20-L28) – meaning in v1.1 results, they compute share% on the fly if it wasn’t stored. The front-end Profile type currently expects type\_scores and top\_types. It’s likely that by v1.2 these are always present. If a profile came from v1.0 (no share\_pct), the UI might detect profile.results\_version and adjust. The results\_version field is included in Profile type and should indicate “v1.2.0” or “v1.1.2” etc.[[188]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1032-L1039)[[189]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1079-L1087). The Results page could show a notice if an older version (e.g. “Results generated with PRISM v1.1 – some metrics like share% unavailable”). However, since PRISM is a live system, older profiles might have been migrated or at least the view functions supply missing info.

**Frontend Components (Cont’d):** Another piece is the narrative or descriptions. PRISM likely has textual descriptions for each type to display. The TYPE\_CORE\_DESCRIPTIONS and prismTypes imported in the ResultsV2 component[[190]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L6-L14) suggest a data file with type descriptions (strengths of that type, etc.). The results page might incorporate a narrative section like “You are a LIE, which means …” possibly customizing if overlays are present (e.g. “with a high Neuroticism overlay, indicating …”). The roadmap explicitly mentions “dynamic narrative” as part of Results Page v2[[191]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L170-L178). That likely means the front-end composes some narrative based on the profile metrics (e.g. highlighting coherent vs unique functions: if certain functions are high-dimensional and also base/creative of their type, those are “coherent strengths” as noted in UI threshold constants[[165]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L11-L19)). The profile provides dims\_highlights.coherent and dims\_highlights.unique lists (functions that were high dimension *and* either part of type’s ego, vs high dimension but not in ego)[[192]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L923-L931). These can feed into text: e.g. “Your profile shows **Ne** and **Fi** are especially well-developed and align with your type’s strengths (coherent), while **Te** is uniquely high for you compared to most in your type” etc. The UI likely leverages these to provide personalized interpretation.

**Retest Comparison:** The roadmap indicates a “Retest View (beta)” where a user can compare two sessions side by side[[193]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L176-L184). This implies the front-end can fetch multiple profiles for the same user and compute deltas. The view v\_retest\_pairs in the DB pre-joins pairs of sessions for each user[[194]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L4-L11). On the front-end, perhaps there’s a page where logged-in users can select two of their past sessions to compare (if accounts are implemented). Each profile stores user\_id if available[[195]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1024-L1031), and session\_kind plus parent\_session\_id to chain retests[[157]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L2-L7). The UI could call a Supabase RPC or view to get the comparison data. In absence of direct multi-profile view, the client can simply fetch two profiles and compute differences in strengths and blocks (the profile JSONs make that straightforward).

**Integration with Supabase Auth:** The system initially allowed anonymous usage. If a user is logged in (perhaps using Supabase Auth), user\_id in session gets populated and carried into the profile[[195]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1024-L1031). The front-end likely checks if a Supabase auth.session exists to decide whether to link the result to the account. They had logic to prevent more than one incomplete session per email[[91]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L14-L19), meaning if the user provides an email and tries again, they might mark the new one as retest. The “History” or “Account” page (if any) would use something like v\_user\_sessions\_chrono view which lists all sessions for a user in chronological order with gap between them[[90]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L20-L28). Indeed History.tsx was in search results, probably listing a user’s completed sessions. This ties into RLS: an authenticated user likely can select \* from profiles where user\_id = my id. Initially, though, given open usage, they rely on share links instead of requiring login.

In conclusion, the frontend is tightly coupled with the backend via Supabase’s client. It invokes the scoring pipeline through finalizeAssessment, then displays the rich profile data. The UI components are designed to reflect all the technical details computed: they visualize absolute vs relative fit (fit vs share), mark close calls[[172]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L51-L59)[[196]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L73-L81), highlight high-dimensional “coherent” functions, and clearly explain what the numeric scores mean. This ensures that the complex calculations done on the back end are translated into an intelligible, engaging user experience. Furthermore, the front-end is built to accommodate ongoing improvements – for instance, the presence of “v2” in component names and explicit handling of calibrated vs raw scores shows it’s prepared for version upgrades. The *Lovable* part likely refers to making results feel personalized and positive, despite being generated from a rigorous algorithmic process.

## 5. CI/CD and DevOps

PRISM’s development pipeline incorporates automated testing of the database and functions, given the critical nature of scoring logic. The project uses **GitHub Actions** to run continuous integration checks on each commit/PR. These checks include:

* **Type Checks and Linting:** The codebase (likely a TypeScript/React front-end and TypeScript Deno functions) is subject to tsc compile checks and ESLint in CI. This prevents type mismatches between front-end expectations and back-end profile structure. For example, the profile JSON structure is typed in the front-end, so any change in the function output (back-end) would cause a TypeScript definition update that must pass CI.
* **Database Migrations Testing:** The team has a practice of validating Supabase migrations in CI before deploying. A custom script scripts/preflight.ts is used as a “smoke test” for SQL migrations[[197]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L125-L133)[[198]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L161-L169). This script essentially spins up a fresh database (or uses the test project database), applies the SQL migration, and then runs sanity checks. Specifically, it reads a consolidated SQL file (sql/2025-ground0-clean.sql) which likely contains the full schema (or the new migration) and executes it on a test DB using the service role key[[199]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L35-L43)[[200]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L45-L53). It catches any SQL errors to ensure no migration will break the DB. Next, it *pings critical Edge Functions* to ensure they deploy and respond. The script calls each function (start\_assessment, finalizeAssessment, score\_prism, etc.) with an OPTIONS then a dummy POST, recording the HTTP status[[201]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L170-L178). A non-200 response might indicate a deployment or code error (for example, if the function fails to load due to a syntax error or missing import, the status could be 500; the preflight would catch that). This effectively serves as an integration test of the serverless functions within CI.
* **RLS and Data Integrity Checks:** The preflight script also verifies that Row-Level Security is correctly configured. It programmatically inserts a test session via the service role, then attempts to read it with an anon key, expecting a failure (status not 200)[[155]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L87-L96)[[156]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L99-L107). It also checks that using the service role can read it (and captures the email to ensure it matches)[[202]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L101-L109)[[203]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L115-L118). This confirms RLS policies like “users can’t read others’ sessions” are in effect. It similarly tests that certain tables like scoring\_config are NOT accessible to anon (expecting a 401/403)[[204]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L117-L123). These automated RLS tests guard against accidentally opening up sensitive data in a migration.
* **Analytics Queries Smoke Test:** After migrations, the preflight runs a set of key read-only queries against the new schema to ensure nothing essential broke. For instance, it runs a distribution query on profiles to get min/max/avg fit in last 30 days, a query on the latest assessments view, a check for any profiles with invalid type combos flagged, etc.[[205]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L132-L140)[[206]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L144-L152). It even checks that the calibration fields are populated (e.g. counts how many profiles are missing calibrated fit)[[207]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L150-L158). By capturing these metrics, the CI can detect if, say, a field that should always be filled is coming out null due to a code bug. The metrics are saved (metrics.json) and a markdown report is generated in the repo (perhaps as an artifact or commit)[[208]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L15-L23)[[197]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L125-L133), providing a snapshot of system health after each run.
* **Deployment Pipeline:** On successful checks, changes are deployed likely to a staging environment first. Supabase allows database branching or separate projects for dev vs prod. The presence of environment variable config and .env.local in the script suggests the CI might apply migrations to a test project (with its own URL and keys)[[209]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L5-L13). For Preview deployments (each PR), they might use Supabase’s shadow DB or Spin up a new project – but since Supabase doesn’t automatically do that, possibly they use a single “staging” project that the preflight resets to baseline each time by running the ground0 SQL. This ensures the migrations are idempotent and can run on a fresh DB.
* **Supabase Migrations via SQL:** The project likely uses Supabase CLI to manage migrations, generating SQL files for each change. The migrations are sequentially numbered (we see timestamps in filenames) and stored in VCS. Applying them in CI imitates what will happen in production. If any migration fails, CI catches it before it hits prod.
* **Smoke Testing Functions:** After deployment (for example, to prod), the team likely monitors logs or uses some uptime check on the functions. The CI preflight itself covers most of it by invoking functions, but they may also have a post-deploy check (perhaps calling a trivial start\_assessment and finalize with test data to see end-to-end).
* **Continuous Deployment & Staging:** The roadmap indicates that version 2.x launched on Aug 18, 2025[[210]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L112-L120), implying a coordinated deployment of both front-end and back-end changes. Likely the team merges to a main branch that auto-deploys to production. For feature branches (PRs), they might deploy to a Vercel or Netlify preview for the front-end and either use the staging Supabase DB or a fork. If a PR includes a DB migration, they would apply it to staging for testing before merging. The CI tests give confidence, but a staging environment allows manual QA.

It’s not explicitly stated, but given typical practices: The Supabase **service\_role** key is used in CI to apply migrations and run tests. The **anon** key is used to simulate client access. They ensure not to commit these keys (using env vars as seen in preflight script expecting .env.local with keys)[[211]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L8-L16)[[212]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L25-L33). The preflight approach is quite comprehensive: it ensures each push doesn’t break the delicate interplay between DB and edge functions.

Additionally, GitHub Actions might run unit tests if any (the heavy logic is in the score\_prism function, which could in theory have unit tests outside the DB context – but since it’s tightly coupled to DB, integration test via preflight is the approach). They might also run front-end tests (if they wrote any Jest/React tests), though not mentioned, but type checks at least.

**DevOps – Monitoring and Migrations:** Supabase migrations are **idempotent** and often include guards (like IF NOT EXISTS in DDL)[[157]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L2-L7)[[213]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L2-L10) so they can be re-run safely (the preflight does a full rebuild). The team uses structured logging in edge functions (with event codes like evt:scoring\_complete or warnings)[[118]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1134-L1142)[[214]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L382-L385), which would appear in Supabase function logs for debugging issues in production. The notify\_admin function likely emails internal stakeholders when something notable happens (like an assessment completed, or an error occurred)[[139]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L70-L78)[[140]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L203-L211). This helps the team react quickly if a bug slips through – e.g., if scoring fails for a user, finalizeAssessment would catch it and could notify admin, and the logs/alerts would show that.

Finally, for **staging vs production**: They might maintain separate Supabase projects for dev and prod. The config file supabase/config.toml was present, possibly indicating environment specific settings. There’s mention of preview setups for PR branches with migrations, meaning they might use Supabase’s upcoming branch feature or manual clones. At minimum, developers can run the preflight locally against a local docker Postgres or a test Supabase instance to ensure migrations produce intended results.

In summary, the CI/CD pipeline is designed to maintain **high reliability for scoring** – crucial because any error in scoring (like a broken SQL or a bug in the function) directly affects user results. By automatically applying migrations and executing the end-to-end scoring flow in a test environment for each change, the team catches issues early. This allows them to move fast with improvements (like introducing v1.2.0 changes, new questions, etc.) without breaking the live system. The extensive logging and metrics generation also provide continuous feedback to guide further development (for example, tracking if the share\_pct is present for all new profiles, or if any profiles ended up with an invalid type combo flag due to some edge answers – which they track with invalid\_combo\_flag and query via modelGuard metric)[[206]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L144-L152)[[215]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L140-L148).

## 6. Roadmap and Recommendations

As PRISM evolves, there are plans to unify and extend its architecture. **Short-term roadmap** (v2.x, launched August 18, 2025) focuses on improving the results presentation and data consistency. This includes a **Results Page v2** with top-3 likelihoods, dimensional highlights, block breakdowns, and dynamic narrative[[191]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L170-L178). These features leverage the rich data already being collected (e.g. showing which functions are “coherent” vs “unique” as described, and providing a PDF export of results). The **retest comparison** feature (allowing users to see changes between an initial test and a retest) is also in beta[[193]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L176-L184). On the backend, enabling such comparisons requires linking sessions by user and ensuring the profile contains all info needed to compute deltas (which it does, since strengths/dimensions are stored). The roadmap’s “Retest core trackability” suggests they want to clearly label if a user’s cognitive core stayed same or not – likely through that stable and adjacent\_flip flags we saw in v\_test\_retest\_strength\_r[[78]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L32-L40). A recommendation here is to expose those stability metrics directly to the user in an intuitive way (e.g. a “stability badge” if type stayed the same, per the roadmap[[216]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L178-L184)).

In terms of **unifying scoring and migrations**, one challenge is that whenever the question set or scoring logic updates, profiles from older versions become incomparable or missing fields. It would be wise to implement a **versioning strategy** where major changes (like introducing new trait dimensions or changing item texts) result in a new results\_version (which they do tag) and possibly a migration that can backfill or recompute old profiles for continuity. For example, if PRISM v2.0 adds a new trait (say Openness), those fields will be null in v1 profiles. The system could either leave them null or attempt a backfill (not trivial without original responses). A more practical approach is to clearly communicate version and let the UI handle differences. To that end, maintaining backward-compatible views (like v\_profiles\_ext that coalesces overlay fields for old/new) is good practice[[51]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L38-L42). Also, ensuring that the profile JSON schema only **appends** new fields (rather than renaming existing ones) helps the front-end not break. Recommendations include creating a **contract test** for the profile schema: e.g., a test that serializes a Profile and ensures required fields are present under various versions. This could be integrated in CI to avoid accidentally removing or renaming a field that the front-end expects.

Looking forward, **PRISM v2.0 and beyond** might expand the theoretical model. One idea is to incorporate additional Big Five dimensions. They already have Neuroticism; adding, for example, an **Extroversion scale** could be valuable, especially since Socionics types do have an I/E aspect (though determined by function extroversion). However, directly adding Big Five factors might complicate the messaging (“you are type X and also high in Y trait”). A more integrated approach could be to derive Big Five proxies from the cognitive data (some research correlates MBTI with Big Five). Still, if user appetite exists, adding separate scales for e.g. **Openness** or **Conscientiousness** (with separate question sets and overlays) might give a fuller personality profile. This would mean expanding the scoring key and handling those like the N trait – perhaps adding O, C, E, A tags with their own overlay or output sections. The system architecture already supports arbitrary trait aggregation (via trait\_scores JSON and overlay fields)[[217]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L2-L9)[[218]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L4-L8), so this expansion is feasible. Just be cautious of assessment length – the test is already ~190 questions. One recommendation is to make any new trait scales *optional or situational*, or use **adaptive testing** to shorten the cognitive part if adding new sections, to keep total time reasonable.

**Adaptive/AI enhancements:** A promising direction is to make the assessment adaptive or personalized. Currently, every user answers the same fixed set of questions. An adaptive approach (e.g. using an Item Response Theory model or simpler branching logic) could shorten the test and increase precision. For example, if early answers indicate extremely high Fi and low Te, the system might not need to ask all 6 Te\_S questions – it could infer enough and skip ahead to something else. The infrastructure can support this: since the front-end is driving question order, it could query an endpoint for “next question given current answers”. Implementing this would require calibrating item difficulties and perhaps using AI to choose the most informative next item. A simpler heuristic approach: detect after half the questions if one or two types are dominating in likelihood, then present some **differentiator questions** targeting the remaining ambiguity (possibly forced-choice focusing on the top candidate vs runner-up type). This could cut down test length for decisive respondents. AI/ML techniques could also be used to analyze **free-text explanations** if any are collected or to cluster profiles beyond the strict type definitions (maybe discovering subtypes or patterns in high-dimensional data).

Another area for AI assistance: **Narrative generation.** Instead of static type descriptions, use GPT-style models to generate a personalized summary from the profile data. Privacy and consistency are concerns here, but it could produce richer, user-tailored insights (“You scored as LIE, with very strong Te. That suggests you excel at organizing and implementing plans. However, your lower Fi might mean you sometimes struggle with personal value-based decisions. Your high Neuroticism overlay indicates you probably experience self-doubt – but note that many LIEs are known to manage stress by doubling down on logic, which you might relate to…” etc.). This could greatly enhance user engagement if done carefully (and perhaps vetted by human experts to avoid errors).

On the **DevOps side**, as the team moves fast, one recommendation is to implement Supabase **database branching** (once stable) to deploy ephemeral databases per PR. This way, a developer could get a full copy of schema + some anonymized data for a feature branch (especially useful if making large changes to scoring key or tables) without impacting others. The test harness is already applying migrations on a blank DB; with branching, they could visually inspect a branch DB in Supabase Studio as well. Also, as logic gets more complex, writing **unit tests for score\_prism** in a controlled environment (using sample input sets and expected outputs) would increase confidence. Right now, much testing is via integration (end-to-end). Isolating the scoring calculation (perhaps by refactoring it into pure functions that can be fed a set of trait values and produce type outcomes) could allow deterministic tests (e.g. feed an extreme profile where one function is maxed, ensure the chosen type is the one with that base function, etc.). This would guard against regressions when adjusting weights or adding new traits.

Finally, to unify everything and make future updates smoother: consider building a **scoring configuration management** interface. Right now, weights and thresholds are in scoring\_config table (e.g. dim\_thresholds, fitBandThresholds)[[115]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L250-L258)[[219]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L262-L265). An internal admin UI to tweak these and run simulations on historical data could greatly speed up iteration. For example, they could simulate raising the penaltyOpp weight from 0.20 to 0.30 and see how many past profiles would change type. Or adjust fc\_block\_map\_default if the forced-choice strategy changes. This kind of tool, possibly combined with an AI that suggests optimal thresholds to maximize stability, would enhance the *evidence-based tuning* of PRISM.

In summary, the roadmap for PRISM is to continue refining both **depth** (more dimensions, better adaptive accuracy) and **accessibility** (clearer UI, more engaging feedback). The architecture is robust and modular enough to handle these: it cleanly separates content (scoring key) from algorithm (score\_prism code) and presentation (front-end). By maintaining this separation and using version flags, the team can introduce “PRISM v2.0” with perhaps additional trait axes or a new algorithm (e.g. maybe a neural network to classify types instead of linear weights) while still supporting v1 results for comparison. Embracing ML for adaptivity and narrative generation, and extending the trait spectrum (e.g. covering Big Five fully or other frameworks like Dark Triad for those interested) are natural next steps to cement PRISM as a comprehensive cognitive and personality profiling system. Each new feature should be tested for reliability (the same way they calibrated confidence, they’d calibrate any new scale) and integrated into the unified scoring pipeline so that all results remain **comparable, interpretable, and actionable** to the end-user.

**References:**

* PRISM Scoring Engine (v1.2) code – type scoring with function weights and dimensional bonuses[[64]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L600-L609)[[220]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L612-L620)
* Scoring key (question mapping) – example entries for function strengths and Neuroticism items[[3]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L70-L78)[[67]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L156-L165)
* Supabase finalizeAssessment – session completion and result output logic[[1]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L119-L128)[[131]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L216-L224)
* Profile data structure – stored fields including version, fit scores, confidence, overlays[[221]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1024-L1032)[[42]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1033-L1041)
* Results Page logic – interpreting fit and displaying top-3 comparison (React component)[[170]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L38-L46)[[173]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L60-L68)
* Psychometric improvements in v1.1.2 – confidence calibration and method blending[[13]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L8-L16)[[14]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L18-L26)
* Database view for latest assessments – how overlays and fit are exposed[[50]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L16-L24)[[222]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L36-L44)
* Preflight CI script – applying migrations, pinging functions, and verifying RLS[[199]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L35-L43)[[155]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L87-L96)

[[1]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L119-L128) [[92]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L172-L180) [[93]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L174-L182) [[120]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L4-L13) [[121]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L6-L8) [[124]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L32-L40) [[125]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L46-L54) [[126]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L50-L58) [[127]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L56-L64) [[128]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L82-L90) [[129]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L123-L131) [[130]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L139-L147) [[131]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L216-L224) [[139]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L70-L78) [[140]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L203-L211) [[163]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L84-L92) [[164]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts#L86-L90) index.ts

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/finalizeAssessment/index.ts>

[[2]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L209-L219) [[5]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L462-L470) [[6]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L543-L552) [[7]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L20-L23) [[8]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L544-L553) [[9]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L711-L719) [[10]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L890-L899) [[11]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1-L9) [[12]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L249-L258) [[16]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L264-L272) [[17]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L16-L25) [[18]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L604-L613) [[19]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L600-L608) [[20]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L39-L48) [[21]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L609-L617) [[22]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L580-L589) [[23]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L586-L595) [[24]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L591-L599) [[25]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L636-L645) [[26]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L375-L382) [[29]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L377-L385) [[30]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L625-L634) [[31]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L626-L634) [[32]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L756-L765) [[33]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L773-L782) [[34]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L783-L786) [[35]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L788-L796) [[36]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L796-L800) [[37]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L839-L847) [[38]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L841-L844) [[39]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L853-L862) [[40]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L865-L873) [[41]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1022-L1031) [[42]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1033-L1041) [[45]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L668-L676) [[46]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L670-L678) [[47]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L674-L683) [[48]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L678-L685) [[49]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L682-L690) [[52]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L444-L452) [[53]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L20-L28) [[54]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L696-L705) [[55]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L713-L721) [[56]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L715-L723) [[59]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1074-L1082) [[60]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1089-L1098) [[61]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1116-L1124) [[62]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L8-L13) [[63]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L16-L24) [[64]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L600-L609) [[65]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L619-L627) [[66]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L636-L644) [[71]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L552-L561) [[72]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L559-L567) [[73]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L678-L686) [[82]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L801-L809) [[83]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L812-L820) [[84]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L820-L828) [[85]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L824-L832) [[101]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L222-L231) [[102]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L233-L240) [[105]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L361-L370) [[106]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L371-L379) [[107]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1073-L1080) [[108]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1034-L1042) [[109]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L869-L878) [[110]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L866-L874) [[111]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L50-L59) [[112]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L62-L70) [[113]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L56-L61) [[114]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1103-L1111) [[115]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L250-L258) [[116]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L260-L268) [[118]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1134-L1142) [[119]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1136-L1144) [[122]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L196-L204) [[123]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L396-L404) [[160]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L399-L407) [[161]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1104-L1112) [[162]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L650-L658) [[188]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1032-L1039) [[189]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1079-L1087) [[192]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L923-L931) [[195]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1024-L1031) [[214]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L382-L385) [[219]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L262-L265) [[220]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L612-L620) [[221]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts#L1024-L1032) index.ts

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/score_prism/index.ts>

[[3]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L70-L78) [[4]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L100-L108) [[27]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L180-L188) [[67]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L156-L165) [[68]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L172-L180) [[69]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L106-L114) [[70]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L112-L120) [[99]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L18-L26) [[100]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L66-L74) [[213]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql#L2-L10) 20250817235706\_759238b8-9285-41fa-a273-4d7c5551882b.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817235706_759238b8-9285-41fa-a273-4d7c5551882b.sql>

[[13]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L8-L16) [[14]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L18-L26) [[15]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L17) [[57]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L14-L22) [[58]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L59-L67) [[77]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L20-L27) [[81]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L10-L18) [[86]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L24-L27) [[117]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L22-L26) [[184]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L89-L97) [[185]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md#L94-L102) v1.1.2-calibration-mai.md

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/docs/v1.1.2-calibration-mai.md>

[[28]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L166-L175) [[43]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L139-L147) [[44]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L140-L148) [[103]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L1-L9) [[104]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql#L29-L37) 20250903015940\_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250903015940_8eae6a30-b0b2-4d00-a581-ad58f8a78d6e.sql>

[[50]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L16-L24) [[51]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L38-L42) [[145]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L44-L52) [[146]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L54-L62) [[147]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L12-L20) [[148]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L22-L30) [[149]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L32-L40) [[150]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L62-L68) [[151]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L36-L43) [[186]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L18-L25) [[187]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L20-L28) [[217]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L2-L9) [[218]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L4-L8) [[222]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql#L36-L44) 20250820\_prism\_traits\_overlays.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250820_prism_traits_overlays.sql>

[[74]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L4-L13) [[75]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L26-L34) [[76]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L30-L38) [[78]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L32-L40) [[79]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L45-L54) [[80]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L51-L59) [[96]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L69-L72) [[97]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L106-L114) [[194]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql#L4-L11) 20250819183747\_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819183747_9fa7e2ba-1b8c-48e9-a323-f1bbccecf649.sql>

[[87]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L2-L10) [[88]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L16-L24) [[94]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L16-L25) [[95]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L18-L26) [[98]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L64-L69) [[154]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L46-L55) [[158]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L71-L79) [[159]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql#L80-L84) 20250817221326\_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250817221326_d7a8fa4e-95d7-4882-9d35-ea2d7bf63cf2.sql>

[[89]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L10-L18) [[90]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L20-L28) [[91]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L14-L19) [[157]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql#L2-L7) 20250819052121\_7b2b052a-1748-4255-bc92-430b05a37a12.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250819052121_7b2b052a-1748-4255-bc92-430b05a37a12.sql>

[[132]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L16-L24) [[133]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L62-L70) [[134]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L99-L107) [[135]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L68-L76) [[136]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L24-L32) [[137]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L116-L125) [[138]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts#L120-L128) index.ts

<https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/supabase/functions/get-results-by-session/index.ts>

[[141]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L2-L10) [[142]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L12-L20) [[143]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L14-L22) [[144]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql#L16-L24) 20250904050000\_fix\_v\_sessions.sql

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/migrations/20250904050000_fix_v_sessions.sql>

[[152]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/getView/index.ts#L16-L24) [[153]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/getView/index.ts#L58-L66) index.ts

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/supabase/functions/getView/index.ts>

[[155]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L87-L96) [[156]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L99-L107) [[197]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L125-L133) [[198]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L161-L169) [[199]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L35-L43) [[200]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L45-L53) [[201]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L170-L178) [[202]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L101-L109) [[203]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L115-L118) [[204]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L117-L123) [[205]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L132-L140) [[206]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L144-L152) [[207]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L150-L158) [[208]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L15-L23) [[209]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L5-L13) [[211]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L8-L16) [[212]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L25-L33) [[215]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts#L140-L148) preflight.ts

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/scripts/preflight.ts>

[[165]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L11-L19) [[166]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L115-L123) [[167]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L119-L127) [[168]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L129-L137) [[169]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L36-L44) [[170]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L38-L46) [[171]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L31-L39) [[172]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L51-L59) [[173]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L60-L68) [[174]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L72-L80) [[175]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L64-L72) [[176]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L70-L78) [[177]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L92-L100) [[178]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L102-L110) [[179]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L40-L48) [[180]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L12-L19) [[181]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L119-L125) [[182]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L8-L16) [[183]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L124-L132) [[190]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L6-L14) [[196]](https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx#L73-L81) ResultsV2.tsx

<https://github.com/ultomotone/prism-discover-yourself/blob/fa1ed8b7ae4537db7196a0334aa4060ecd1af602/src/components/assessment/ResultsV2.tsx>

[[191]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L170-L178) [[193]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L176-L184) [[210]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L112-L120) [[216]](https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx#L178-L184) Roadmap.tsx

<https://github.com/ultomotone/prism-discover-yourself/blob/41e414228dc7acc5a7b3e72b71caf6b65d9148e7/src/pages/Roadmap.tsx>