Data Layer Migration & Performance Interview Script

Opening (0:00-0:10)
"We�re midway through migrating from ad-hoc Supabase calls to a typed React Query data layer, and I�ve been shepherding the coexistence strategy."

Legacy vs New Hooks (0:10-0:35)
"`useUserWiggs.ts` still talks directly to Supabase, but it now respects an `enabled` flag so `MediaTile` can disable it when the new data layer is active. The replacement hook, `src/data/hooks/useUserWiggsDataLayer.ts`, wraps TanStack Query, exposes the same API shape, and pipes through a shared `wiggPointsClient`."

Caching Guardrails (0:35-0:55)
"We saw dashboards spiking 118 Supabase calls on load. `useTitleMetrics.ts` fixes that by setting a 15-minute stale time, disabling refetch-on-focus, and pushing GC tails out to 30 minutes because those metrics barely change."

Testing & Infra (0:55-1:15)
"We back this with Playwright monitoring in `tests/api-performance.spec.ts`, which groups network calls and fails the build if counts exceed thresholds. On the unit side we�re moving toward MSW handlers in `src/data/mocks` so hooks can be tested against realistic responses."

Migration Plan (1:15-1:30)
"`docs/data-layer-migration-plan.md` codifies the phased rollout�Phase 2 lets old and new hooks coexist via feature flags, and Phase 3 will retire the legacy services once coverage hits 100%."

Wrap (1:30-1:35)
"It�s a practical example of shipping incremental architecture without regressing performance in production."