Plan integral técnico – Plataforma Web + App Móvil (Instituciones deportivas)

Versión: 0.2
Autor: Reescritura técnica consolidada
Stack base (open source): - **Backend:** Node.js + **Express** (BFF), supabase-js (DB/Auth/Realtime/Storage)
- **Web: React** + Vite, TanStack Query, ShadCN UI/Tailwind
- **Móvil: React Native + Expo** (Android/iOS), React Navigation, Expo Notifications, expo-image
- **Base de datos: Supabase (PostgreSQL)** con RLS, Realtime, Storage
- **Infra:** Supabase Project (DB + Auth + Storage + Realtime), Deploy del BFF en Railway/Fly.io/Clever Cloud (o VPS)
- **Repos separados (no monorepo):** - `sports-backend/` (Express)
- `sports-web/` (React)
- `sports-mobile/` (Expo RN)

---

1) Resumen ejecutivo

Plataforma multi-institución para academias/clubes deportivos que cubre **gestión administrativa** (web) y **operación deportiva** (web + móvil). La app móvil atiende **representantes** y **deportistas**; el panel web atiende **Super Admin (global)**, **Admin de Institución**, **Admin de Sede** y **Colaboradores**. Se incluye **RLS por institución/sede**, **pagos** (efectivo/transferencia en v1; tarjeta en v2), **asistencias con QR**, **convocatorias**, **calendario**, **estadísticas**, **logros/kits** y **feed tipo red social** con sponsors.
En móvil: **push notifications**, **subida de comprobantes**, **modo offline con cola de sincronización**.

---

2) Perfiles de usuario y alcance

- **Super Admin (global):** ve y gestiona *todas* las instituciones y sedes. Crea instituciones, roles globales, usuarios globales.
- **Admin de Institución:** ve su institución y *todas sus sedes*. Crea/edita sedes, colaboradores, roles locales, grupos, publicaciones, notificaciones, pagos.
- **Admin de Sede:** ve solo su sede. Gestiona grupos, eventos, asistencias, convocatorias, publicaciones, pagos y colaboradores de su sede.
- **Colaboradores:**
- **Entrenador:** grupos asignados, asistencias (incl. QR), convocatorias, estadísticas, publicaciones.
- **Secretario/personal técnico:** gestión administrativa (pagos, registros, comunicaciones).
- **Representante:** auto-registro, registra deportistas, ve calendario, asistencias, pagos, notificaciones, feed.
- **Deportista:** ve eventos, convocatorias, estadísticas, logros/kits, notificaciones, feed.

Scopes de acceso: `global`, `institucion_id`, `sede_id`, `grupo_id`.

---

3) Arquitectura y componentes

3.0 Arquitectura MVC (web y móvil) - **Backend (Express) – MVC:** Models (repositorios Supabase), Controllers (casos de uso), Services (Storage, Push, QR), Routes, Middlewares. - **Web (React):** Model (TanStack Query), View (ShadCN/Tailwind), Controller (hooks/handlers). - **Móvil (Expo):** Model (React Query + persistencia), View (pantallas RN), Controller (hooks por flujo).

3.1 Vista lógica - **Cliente Web:** backoffice multi-tenant. - **App Móvil:** front para Representantes/ Deportistas (y Entrenadores v1.1). - **BFF (Express):** orquestación de pagos, QR, notificaciones. - **Supabase:** fuente de verdad con RLS. - **Push:** Expo Notifications.

3.2 Offline: cache persistente + cola de sincronización.

3.3 Storage: buckets para comprobantes, perfiles, publicaciones, sponsors, branding.

3.4 Seguridad: Auth Supabase (email/Google/Apple), RLS, JWT con claims de institución/sede/rol, auditoría.

---

4) Módulos web y pantallas móviles

4.1 Web – Super Admin

- Dashboard global: métricas de instituciones, usuarios, actividad.
- CRUD instituciones y roles globales.

4.2 Web – Admin de Institución

- **Dashboard institucional:** KPIs (deportistas, morosidad, ingresos, asistencia, convocatorias).
- **Sedes:** CRUD (incluye `ubicacion` geográfica) y **branding** (logo, colores, bandera).
- **Colaboradores:** alta/asignación de roles (permisos en JSONB) y sedes.
- **Representantes:** alta/vinculación.
- **Deportistas:** alta, **ficha médica opcional**.
- **Grupos:** roster, entrenador principal + entrenadores adicionales.
- **Horarios:** definición de `horarios_entrenamiento` por grupo.
- **Eventos/Partidos:** creación y enlace `eventos.partido_id`; convocatorias; resultados.
- **Asistencias:** pase de lista/QR con estados (`estado_asistencia`).
- **Pagos (opcional si habilitado):** órdenes, comprobantes, conciliación.
- **Publicaciones (mini red social):**
- Crear/editar/eliminar publicaciones con **media** (imágenes/video) en Storage.
- **Visibilidad**: `global` (institución), `sede`, `grupo`.
- **Programación** (`publicar_en`), **borradores**, **previsualización** y **moderación**.
- Se muestran en **Inicio** de la **app móvil** (feed). Reacciones/comentarios (fase 2).
- **Notificaciones (compositor):**
- Crear notificación con **título**, **cuerpo**, tipo **push** (Expo) y/o **in-app** (persistente en `notificaciones`).
- **Segmentación** por **audiencia**: `representantes`, `deportistas`, `entrenadores`, `colaboradores`, o combinaciones; filtros por **institución/sede/grupo**.
- **Variables** (p.ej. `{{nombre}}`, `{{grupo}}`, `{{fecha_evento}}`) y **plantillas**.
- **Enviar ahora** o **programar**; vista previa; registro de métricas (entregadas, leídas).

4.2.1 Web – Admin de Sede

- Dashboard de sede: KPIs locales (asistencia, morosidad, próximos partidos).
- CRUD grupos, eventos, asistencias, convocatorias, publicaciones **de su sede**.
- **Notificaciones de sede**: compositor con segmentación a representantes/deportistas/entrenadores **de su sede**.
- Conciliación de pagos y gestión de colaboradores de la sede. Web – Admin de Sede
- Dashboard de sede: KPIs locales (asistencia, morosidad, próximos partidos).
- CRUD grupos, eventos, asistencias, convocatorias, publicaciones.
- Conciliación de pagos y gestión de colaboradores de la sede.

4.3 App móvil – Representante

- Inicio (feed con sponsors).
- Pagos (estado de cuenta, comprobantes, historial).
- Mis hijos (alta de deportistas, ficha médica opcional).
- Asistencias, calendario, notificaciones, perfil.

4.4 App móvil – Deportista

- Inicio (feed).
- Eventos, convocatorias, estadísticas, logros/kits, perfil.

4.5 App móvil – Entrenador (v1.1)

- Grupos, pase de lista (QR), convocatorias, resultados.

---

5) Flujos clave

1. **Registro/Login**
2. Web: admins/colabs.
3. Móvil: representante (elige sede) y crea deportista vinculado. **Ficha mínima** obligatoria; **ficha médica** opcional editable luego.
4. **Organización deportiva**
5. Grupos por sede; entrenador principal + adicionales; horarios semanales.
6. **Eventos/Partidos**
7. Crear entrenamientos/partidos; si es partido, enlazar `partidos` y gestionar **convocados**; registrar **rendimiento** post-partido.
8. **Asistencias con QR**
9. Generar QR por evento/grupo; entrenador escanea y registra en `asistencias` (idempotente por `UNIQUE`).
10. **Pagos**
11. Efectivo (orden + confirmación), transferencia (comprobante desde móvil). **Tarjeta** en fase 2.
12. **Publicaciones → Feed móvil**
13. Desde **web** se crean publicaciones (con media en Storage, visibilidad `global/sede/grupo`, programación).
14. **App móvil** consume `/feed` y muestra en **Inicio** con caché y soporte offline.
15. **Compositor de notificaciones**
16. En **web**: redacción (título/cuerpo), elección de **tipo** (push/in-app), **segmentación** por rol y por **institución/sede/grupo**; variables/plantillas; envío inmediato o programado.

17. En **móvil**: recepción de **push** (Expo) y bandeja de **in-app** persistente; estados *entregada/leída*.
18. **Estadísticas/Logros**
19. Entrenador captura estadísticas y el sistema calcula KPIs; asignación de logros.
20. **Branding**
21. Tema por institución/sede aplicado en web y móvil.

---

6) Modelo de datos

Entidades: instituciones, sedes, roles, usuarios_sistema, representantes, deportistas, grupos, eventos, convocatorias, asistencias, (pagos opcional), **publicaciones**, **notificaciones**, sponsors, estadísticas, logros, kits, chat, branding.

Publicaciones (mini red social)
- `publicaciones(id, institucion_id, sede_id?, grupo_id?, titulo, cuerpo, media_urls jsonb, visibilidad enum('global','sede','grupo'), publicar_en timestamptz, estado enum('borrador','programada','publicada','oculta'), creada_en)`

Notificaciones (compositor)
- `notificaciones(id, titulo, cuerpo, tipo enum('push','inapp','ambas'), audiencia jsonb, institucion_id, sede_id?, grupo_id?, programada_en?, enviada_en?, entregadas int, leidas int, creada_en)` - `audiencia` puede incluir filtros por **rol** (`representante`, `deportista`, `entrenador`, `colaborador`) y listas específicas de `user_ids` cuando aplique.

Branding: logo, bandera, colores (primario/secundario/acento), tema (light/dark), validación de contraste.

---

7) Permisos y RLS

- Clientes con `anon` + RLS; server con service role solo en tareas internas.
- Roles por módulo/acción (`pagos.confirmar`, `grupos.editar`, `publicaciones.crear`, `notificaciones.enviar`).
- Claims JWT: `rol`, `institucion_id`, `sedes_ids`, `grupos_ids`.

Políticas ejemplo (conceptual):

```
-- Publicaciones: lectura por alcance y pertenencia
create policy sel_publicaciones on publicaciones
for select using (
  case visibilidad
    when 'global' then auth.jwt()->>'institucion_id' = institucion_id::text
    when 'sede'   then (auth.jwt()->>'sedes_ids')::jsonb ? sede_id::text
    when 'grupo'  then exists (
      select 1 from grupos g
      where g.id = publicaciones.grupo_id
```

```
        and (auth.jwt()->>'sedes_ids')::jsonb ? g.sede_id::text
    )
  end
);

-- Publicaciones: escritura solo admin de institución/sede correspondiente
create policy ins_upd_publicaciones on publicaciones
for all using (
  auth.jwt()->>'rol' in ('admin_institucion','admin_sede','super_admin')
) with check (
  auth.jwt()->>'institucion_id' = institucion_id::text
);

-- Notificaciones: solo admins pueden crear/enviar dentro de su institución
create policy ins_notificaciones on notificaciones
for insert with check (
  auth.jwt()->>'rol' in ('admin_institucion','admin_sede','super_admin')
  and auth.jwt()->>'institucion_id' = institucion_id::text
);

-- Notificaciones: lectura por destinatario (in-app)
create policy sel_notificaciones_inapp on notificaciones
for select using (
  -- visible si audiencia incluye mi rol y mi sede/grupo
  (audiencia->>'rol') is null
  or (audiencia->'roles') ? (auth.jwt()->>'user_type')
);
```

Storage: `public/branding` y `public/sponsors` lectura pública; media de publicaciones en `public/publicaciones` (lectura pública) o `private/publicaciones` (si se requiere control fino). Comprobantes/fotos de perfil en **privado** con signed URLs.

---

8) API Express – endpoints principales

- **Auth:** login/logout, perfil.
- **Instituciones/Sedes:** CRUD, branding.
- **Roles/Usuarios:** CRUD roles y colaboradores.
- **Representantes/Deportistas:** auto-registro, fichas mínima/extendida.
- **Grupos/Eventos:** CRUD, convocatorias, QR.
- **Pagos:** órdenes, comprobantes, confirmaciones (si módulo activo).
- **Publicaciones (mini red social):**
- `POST /publicaciones` `{ titulo, cuerpo, media_urls?, visibilidad, publicar_en? }`
- `PUT /publicaciones/:id` • `DELETE /publicaciones/:id`
- `GET /feed?scope=institucion|sede|grupo&id=...` (paginado, con `publicar_en <= now()`)
- **Notificaciones (compositor):**

- `POST /notificaciones/preview` `{ titulo, cuerpo, tipo, audiencia }` → render + conteo de destinatarios
- `POST /notificaciones/enviar` `{ titulo, cuerpo, tipo, audiencia, programada_en? }` → crea registro y dispara **Expo Push** a `push_tokens`; guarda **in-app** en `notificaciones`
- `POST /me/push-token` `{ token }` para registrar/actualizar token del usuario actual
- `GET /notificaciones?inapp=true` (bandeja del usuario)
- **Estadísticas/Logros/Kits:** CRUD y asignación.
- **Chat:** threads y mensajes.

Las respuestas se mantienen `{ success, data?, error? }`. Usar PostgREST cuando sea posible; Express para validaciones, programación de envíos, idempotencia y fan-out de push.

---

9) Navegación móvil

- **Representante:** Inicio • Pagos • Hijos • Asistencias • Calendario • Notificaciones • Perfil.
- **Deportista:** Inicio • Eventos • Convocatorias • Estadísticas • Logros • Perfil.
- **Entrenador (v1.1):** Grupos • Evento • Pase de lista (QR) • Convocatorias • Resultados.

---

10) Requisitos no funcionales

- **Offline-first:** cache + cola con reintentos.
- **Rendimiento:** listas virtualizadas, índices DB.
- **Accesibilidad:** contraste, lector pantalla.
- **Idiomas:** ES (default), EN opcional.
- **Compatibilidad:** Android 8+/iOS 14+.
- **Seguridad:** HTTPS, JWT corto, Zod, auditoría.
- **Observabilidad:** logs estructurados, métricas.
- **Personalización segura:** branding con validación de contraste y paleta aprobada.
- **Ficha mínima:** representante (nombres, cédula, email, teléfono), deportista (nombres, nacimiento, grupo/sede).

---

11) Roadmap por sprints

- **Sprint 0:** setup repos, CI/CD, schema base, RLS mínima.
- **Sprint 1:** instituciones/sedes + branding lectura.
- **Sprint 2:** usuarios (colaboradores, representantes, deportistas) + branding edición.
- **Sprint 3:** grupos/eventos + notificaciones push.
- **Sprint 4:** asistencias + QR.
- **Sprint 5:** pagos v1 (efectivo/transferencia).
- **Sprint 6:** feed/sponsors.
- **Sprint 7:** estadísticas/logros.
- **Sprint 8:** entrenador móvil (v1.1).
- **Sprint 9:** pagos v2 (tarjeta).
- **Sprint 10:** chat/avisos.

12) Riesgos y decisiones abiertas

- Pagos tarjeta dependen de proveedor externo.
- Apple Sign-In puede ser obligatorio en iOS.
- Alcance de entrenador móvil en v1 vs v1.1.
- Moderación de chat.
- Definir operaciones bloqueadas sin conexión.
- Branding: limitar para evitar problemas de accesibilidad.

---

13) Checklist técnico

- Claims JWT correctos.
- RLS en todas las tablas sensibles.
- Buckets definidos (`public/branding`, `public/sponsors`, `private/*`).
- Cola offline móvil implementada.
- Push tokens por usuario.
- QR server-side con expiración.
- Auditoría y soft delete.
- Pruebas E2E mínimas.
- Documentación de roles/permisos.
- Validación de contraste en branding.

---

14) Matriz de permisos

Módulo/Acción	Super Admin	Admin Inst.	Admin Sede	Entrenador	Secretario	Representante	Deportista
Instituciones	✔️	✖️	✖️	✖️	✖️	✖️	✖️
Sedes	✔️	✔️	✔️(su sede)	✖️	✖️	✖️	✖️
Branding	✔️	Editar inst.	Editar sede	✖️	✖️	✖️	✖️
Roles/Permisos	✔️	✔️	✔️(sede)	✖️	✖️	✖️	✖️
Colaboradores	✔️	✔️	✔️(sede)	✖️	✖️	✖️	✖️
Representantes	✔️	✔️	✔️(sede)	✖️	✔️	✖️	✖️
Deportistas	✔️	✔️	✔️(sede)	✔️(grupo)	✔️	✖️	✖️
Grupos	✔️	✔️	✔️(sede)	✔️	✖️	✖️	✖️
Eventos	✔️	✔️	✔️(sede)	✔️	✔️	Ver	Ver
Asistencias	✔️	✔️	✔️(sede)	Marcar	✔️	Ver	Ver propia
Convocatorias	✔️	✔️	✔️(sede)	Crear	✔️	Confirmar	Confirmar

Módulo/Acción	Super Admin	Admin Inst.	Admin Sede	Entrenador	Secretario	Representante	Deportista
Pagos	✔️	✔️	✔️(sede)	✖️	Confirmar	Pagar	✖️
Publicaciones	✔️	Publicar	Publicar	Publicar	Publicar	Ver	Ver
Estadísticas	✔️	✔️	✔️(sede)	Cargar	✔️	Ver hijos	Ver propia
Logros/Kits	✔️	Asignar	Asignar	Proponer	✔️	Ver	Ver
Chat/Avisos	✔️	✔️	✔️	Grupo/1:1	✔️	1:1/grupo	1:1/grupo

---

15) Metodología recomendada

Scrum ligero (sprints 2 semanas) + Kanban para bugs: planning, dailies, reviews, retros, grooming.
Alternativa: Kanban estricto con demos quincenales si el equipo es pequeño.

---

16) Diagramas de arquitectura y flujos (texto/mermaid)

> Los siguientes diagramas están en formato **Mermaid** (puedes copiarlos a cualquier visor Mermaid o a tu repo para verlos renderizados).

16.1 Arquitectura lógica (alto nivel)

```
flowchart LR
  subgraph Client[Clientes]
    W[Web (React)]
    M[App Móvil (Expo RN)]
  end

  subgraph Server[Backend]
    BFF[Express BFF
(Controllers/Services)]
  end

  subgraph Supabase[Supabase]
    DB[(PostgreSQL
+ RLS)]
    AUTH[(Auth)]
    RT[(Realtime)]
    ST[(Storage)]
  end

  W <---> BFF
  M <---> BFF
  BFF <---> DB
  BFF <---> AUTH
```

```
  BFF <---> ST
  M <---> RT
  W <---> RT

  subgraph Push[Expo Push]
    PN[(Push Notifications)]
  end

  BFF --> PN
```

16.2 Flujo de pagos (efectivo/transferencia v1)

```
sequenceDiagram
  participant R as Representante (App)
  participant W as Web Admin
  participant B as BFF (Express)
  participant DB as Supabase (DB/Storage)

  R->>B: Generar orden de pago (mes actual)
  B->>DB: INSERT pagos_ordenes
  DB-->>B: OK (orden_id)
  B-->>R: orden_id, estado=pending
  R->>B: Subir comprobante (foto)
  B->>DB: Upload Storage + INSERT pagos_comprobantes
  B-->>W: Notificación nueva orden/comprobante
  W->>B: Confirmar pago (validado)
  B->>DB: UPDATE pagos_ordenes.estado=paid; INSERT pagos_movimientos
  B-->>R: Recibo/estado actualizado
```

16.3 Asistencia con QR

```
sequenceDiagram
  participant E as Entrenador (App)
  participant B as BFF
  participant DB as Supabase

  E->>B: Solicitar QR para Evento/Grupo
  B->>DB: Generar token + guardar expiración
  B-->>E: QR (token)
  E->>B: Escanear y marcar asistencia
  B->>DB: INSERT asistencias (idempotente)
  B-->>E: OK/estado
```

16.4 Navegación móvil (tabs)

```
flowchart LR
  subgraph Representante
    R1(Inicio)-->R2(Pagos)
```

```
  R1-->R3(Hijos)
  R1-->R4(Asistencias)
  R1-->R5(Calendario)
  R1-->R6(Notificaciones)
  R1-->R7(Perfil)
end
subgraph Deportista
  D1(Inicio)-->D2(Eventos)
  D1-->D3(Convocatorias)
  D1-->D4(Estadísticas)
  D1-->D5(Logros)
  D1-->D6(Perfil)
end
```

---

17) Plan de entrega por semanas (2–3 meses)

> **Meta:** MVP funcional en **8–10 semanas** con un equipo de ~**3 devs** (1 backend, 1 web, 1 mobile) + apoyo de diseño part-time.

Alcance MVP (sí)

- Auth (email/Google), instituciones/sedes, roles/alcance, representantes/deportistas, grupos, calendario/eventos, **asistencia con QR**, **convocatorias**, **pagos v1** (efectivo/transferencia), **feed + sponsors**, **branding básico**, push notifications, offline mínimo (cola simple), dashboards básicos.

Fuera del MVP (fase 2)

- Pago con **tarjeta**, chat/avisos avanzados, estadísticas profundas y logros complejos, Apple Sign-In, módulo móvil de entrenador completo.

Cronograma sugerido

- **Semana 1:** Setup repos, CI/CD, esquema base, Auth, RLS mínima, diseño UI kit.
- **Semana 2:** Instituciones/sedes + roles/alcance + branding lectura.
- **Semana 3:** Representantes y deportistas (web y API) + auto-registro móvil (representante).
- **Semana 4:** Grupos y calendario de eventos; estructura de notificaciones.
- **Semana 5:** Convocatorias + push; feed/sponsors (web crea, móvil ve).
- **Semana 6:** Asistencia con **QR** (generación + escaneo) + vistas.
- **Semana 7: Pagos v1**: órdenes, carga de comprobantes, conciliación web.
- **Semana 8:** Offline mínimo (cola), dashboards, pulido, pruebas E2E.
- **Semana 9–10 (buffer/duros):** QA, performance, accesibilidad, hardening, preparación de publicación móvil.

Supuestos

- Requisitos cerrados para MVP (sin cambios mayores).
- Acceso estable a Supabase/credenciales.
- Decisiones rápidas en UX.
- Sin integraciones externas de tarjeta en v1.

Riesgos que pueden alargar

- Cambios de alcance durante el sprint.
- Retrasos en Apple Sign-In o publicación en App Store.
- Falta de disponibilidad de diseño o QA.

---

18) (Sección eliminada)

Se elimina la sección de PROMPT/Replit. El plan integral queda enfocado únicamente en el diseño técnico, modelo de datos, flujos y APIs.
