Nombre del sistema: Sistema de Gestión y Planificación para Gimnasios Modernos.

Objetivo del sistema

Construir un sistema monolítico de escritorio, offline, para gimnasios medianos que:

- Administre socios, membresías, pagos y asistencia.
- Personalice rutinas y planes de nutrición con un builder de arrastrar-y-soltar y catálogos locales.
- Haga seguimiento de progreso (medidas, cargas, IMC) y recomiende ajustes cuando el estancamiento sea evidente.

Diferenciales: funciona sin internet(El sistema se ejecuta de manera local en el computador del gimnasio), incluye planificación personalizada (rutinas + nutrición) con seguimiento visual, y usa patrones de diseño.

Alcance y supuestos

- Plataforma: escritorio (windows).
- Base de datos: relacional local.
- Roles internos: Administrador, Entrenador, Recepción.
- (Opcional) Integración posterior a API nutricional.

Listado de Requerimientos

Requerimientos funcionales

A) Administración de socios y membresías (ADM)

- **ADM-01** Registrar/editar datos de socio (identificación, contacto, salud básica, consentimiento).
- **ADM-02** Crear/editar planes de membresía (tipo, precio, duración, beneficios).

- **ADM-03** Asignar membresía a socio con fechas de inicio/fin calculadas.
- **ADM-04** Cambiar automáticamente estado de la membresía según reglas (Activa, Morosa, Expirada, Suspendida).
- **ADM-05** Bloquear acciones de acceso/asistencia si la membresía no está Activa.
- **ADM-06** Listar socios por estado de membresía y filtrar por vencimientos próximos (≤7 días).

B) Control de acceso y asistencia (ACC)

- **ACC-01** Registrar entrada/salida de socios (búsqueda por nombre o documento).
- **ACC-02** Impedir registro de entrada si la membresía está Expirada/Suspendida/Morosa.
- **ACC-03** Generar conteo diario y picos de asistencia (para capacidad).

C) Pagos y alertas (PAY)

- **PAY-01** Registrar pagos (efectivo, tarjeta, transferencia) y emitir recibo.
- **PAY-02** Marcar membresía como Activa al recibir pago correspondiente.
- **PAY-03** Generar alertas de pago por vencer (7 y 3 días antes) y notificar en panel.
- **PAY-04** Listar morosos y soporte (pausar/reanudar).

D) Catálogos (CAT)

- **CAT-01** Mantener catálogo de ejercicios (nombre, grupo muscular, equipo, descripciones, variantes).
- **CAT-02** Mantener catálogo de alimentos (porción, kcal, macros).
- **CAT-03** Cargar catálogos al iniciar la app para reutilización en toda la sesión.
- **CAT-04** Versionar cambios para auditoría básica (quién/cuándo).

E) Planificación de rutinas (RUT)

- **RUT-01** Crear rutinas semanales por socio (≥1 y ≤6 días configurables).

- **RUT-02** Añadir a cada día ejercicios con series, repeticiones, tempo y peso objetivo.
- **RUT-03** Proveer builder (arrastrar-y-soltar) desde CAT-Ejercicios.
- **RUT-04** Guardar plantillas reutilizables (p. ej., “Full-body 3x”).
- **RUT-05** Registrar ejecuciones y progreso por ejercicio (peso/series efectivas, RPE, notas).
- **RUT-06** Validar incompatibilidades básicas (p. ej., evitar dos sesiones de pierna muy pesadas en días consecutivos si no está marcado como avanzado).

F) Planificación nutricional (NUT)

- **NUT-01** Definir objetivo calórico diario para el socio (mantenimiento/ganancia/pérdida).
- **NUT-02** Componer plan por comidas (desayuno/almuerzo/cena/snacks) a partir del catálogo de alimentos.
- **NUT-03** Calcular totales de kcal y macros por día/semana.
- **NUT-04** Permitir sustituciones manteniendo el objetivo calórico.
- **NUT-05** (Opcional) Consultar API nutricional para nuevos ítems y cachear en el catálogo.

G) Progreso y analítica (PRG)

- **PRG-01** Registrar peso corporal, medidas (cintura, brazo, etc.) con fecha.
- **PRG-02** Graficar evolución (peso, %cambios en 1RM estimado, volumen semanal por grupo).
- **PRG-03** Comparar objetivos vs progreso (semana/mes) y resaltar estancamientos.
- **PRG-04** Exportar reportes PDF básicos (resumen mensual).

H) Recomendaciones y reglas (REC)

- **REC-01** Detectar estancamiento (ej. 12 semanas sin mejora ≥2.5% en 1RM estimado de un levantamiento clave) y sugerir cambio (variar volumen/intensidad/ejercicio).

- **REC-02** Sugerir deload si hay caída de rendimiento ≥10% en ≥2 sesiones consecutivas del mismo patrón.
- **REC-03** Generar alertas no intrusivas en el panel del entrenador y registrar las acciones tomadas.

I) Notificaciones internas / Observer (OBS)

- **OBS-01** Al actualizar medidas o pesos de entrenamiento, notificar panel del entrenador activo.
- **OBS-02** Al modificar una clase grupal, notificar a los socios inscritos en su agenda interna.

J) Seguridad y configuración (SEC)

- **SEC-01** Autenticación por rol (Administrador/Entrenador/Recepción) y permisos mínimos.
- **SEC-02** Backups locales programables y restauración.
- **SEC-03** Registro de auditoría básico (inicios de sesión, altas/bajas de membresías, cambios de catálogos).
- **SEC-04** Cifrado en reposo de datos sensibles (p. ej., contactos/observaciones médicas).

Requerimientos no funcionales (NFR)

- **NFR-01 (Arquitectura):** Monolítico de escritorio.
- **NFR-02 (Rendimiento):** Registro de entrada < 1 s desde búsqueda hasta confirmación en equipos estándar.
- **NFR-03 (Confiabilidad):** No perder datos ante caída; autoguardado transaccional en operaciones críticas (pagos, asistencia).
- **NFR-04 (Usabilidad):** Builder de rutinas drag-and-drop, accesible con teclado; tipografías legibles.
- **NFR-05 (Portabilidad):** Ejecutable en Windows 10+; deseable Linux o macos.

- **NFR-06 (Mantenibilidad):** Tests unitarios para reglas de estado de membresía, IMC y recomendaciones.

Historias de usuario

- **HU-01 (Recepción):** “Como recepción, quiero **registrar entradas** rápidamente para evitar filas.”
    - Dado un socio con membresía Activa, cuando escaneo/busco su documento y confirmo, entonces el sistema registra la entrada y muestra “Acceso permitido” en <1 s.
- **HU-02 (Administrador):** “Como administrador, quiero ver socios por vencer para prevenir morosidad.”
    - Dado la lista de socios, cuando filtro por vencen ≤7 días, entonces obtengo el listado con contacto y monto a pagar.
- **HU-03 (Entrenador):** “Como entrenador, quiero crear una rutina arrastrando ejercicios del catálogo.”
    - Dado un socio, cuando abro el builder y suelto ejercicios en la semana, entonces puedo guardar la rutina con series/reps/peso y reutilizarla como plantilla.
- **HU-04 (Entrenador):** “Como entrenador, quiero ver la evolución de cargas para detectar estancamiento.”
    - Dado las ejecuciones registradas, cuando veo la gráfica del levantamiento clave, then el sistema marca estancamiento si 12 semanas sin mejora ≥2.5% y propone alternativas.

Reglas de negocio clave

- **Activa:** permite asistencia; habilita registro de sesiones.
- **Morosa: no** permite entrada; se reactiva con pago.
- **Expirada: no** permite entrada; requiere renovar.
- **Suspendida: no** permite entrada; solo administrador puede levantar.

Datos principales

Socio, Membresía, Pago, Asistencia, Ejercicio, Rutina, DíaRutina, Ejecución, Alimento, PlanNutricional, Medición, Reglas/Alertas.

Relaciones M:N: Rutina–Ejercicio, **PlanNutricional–Alimento** (con atributos de cantidad/porción).

Listado de Requisitos

Administración de socios y membresías (ADM)

ADM-R1: El sistema debe permitir crear y editar un socio con los siguientes campos obligatorios: identificación (string, único), nombre completo (string), teléfono (string), correo electrónico (string, validación RFC), fecha de nacimiento (date, YYYY-MM-DD), consentimiento escrito sobre tratamiento de datos (boolean), estado de salud básica (enum: Normal / Con Limitación / Contraindicado, con nota opcional (string, ≤500 chars)).

ADM-R2: El sistema debe permitir crear un tipo de membresía con nombre (string), precio (decimal, 2 decimales, ≥0), duración en días (int, >0), beneficios (string, ≤1000 chars). Al asignar una membresía a un socio, el sistema debe calcular fecha inicio (date, YYYY-MM-DD) y fecha fin (date, YYYY-MM-DD) = fecha inicio + duración en días - 1 y almacenar ambos.

Control de acceso y asistencia (ACC)

ACC-R3: El sistema debe registrar entradas/salidas con socio id (string), timestamp (datetime), terminal id (string) y tipo (enum: Entrada / Salida). Antes de crear un registro de entrada, validar que el estado de la membresía sea Activa; si no, bloquear la operación y devolver motivo (enum: Expirada / Morosa / Suspendida).

ACC-R4: El sistema debe generar, para cualquier fecha solicitada, conteo total diario (int) y pico horario (string o time range, p. ej. "17:00–18:00") y exponerlo en panel o exportable CSV (archivo local).

Pagos y alertas (PAY)

PAY-R5: Registrar pagos con pago id (string), monto (decimal, 2 decimales), moneda (string, p. ej. "COP"/"USD"), metodo (enum: Efectivo / Tarjeta / Transferencia), fecha (date o datetime según necesidad), referencia transacción (string, opcional) y generar recibo id (string) con formato `RC-{AÑO}{MES}{DIA}-{secuencia}`. El recibo debe poder imprimirse o guardarse como PDF (archivo local).

PAY-R6: Al registrar un pago cuyo monto cubre la membresía pendiente del socio, el sistema debe poner el estado de la membresía a Activa y calcular/actualizar fecha inicio (date) y fecha fin (date) si corresponde. Si el pago es parcial, marcar deuda pendiente (decimal) y no activar.

Catálogos y gestión de datos (CAT)

CAT-R7: Mantener ejercicios con campos obligatorios: ejercicio id (string), nombre (string, único por gimnasio, ≤150 chars), grupo muscular (enum), equipo (list[string]), descripción (string, ≤2000 chars), variantes (list[string], opcional). Las búsquedas deben soportar coincidencia parcial por nombre y filtrado por grupo muscular.

CAT-R8: Mantener alimentos con alimento id (string), nombre (string), porción (decimal, gramos), kcal por porción (decimal), macros (objeto con proteína (decimal, g), grasa (decimal, g), carbohidrato (decimal, g)). Al iniciar el sistema debe cargarse en memoria (o caché local) el catálogo activo para la sesión; las ediciones deben crear una versión nueva con versión id (string), usuario (string), timestamp (datetime) y diff mínimo (string).

Planificación de rutinas (RUT)

RUT-R9: El sistema debe permitir crear una rutina semanal con entre 1 y 6 días configurables por socio. Cada día contiene una lista ordenada de ejercicios (referencia a ejercicio id (string)) con atributos: series (int, 1–10), repeticiones (int, 1–100), tempo (string, formato e.g. "2-0-1"), peso objetivo (decimal, kg, ≥0). El sistema debe impedir guardar rutinas que violen los límites (por ejemplo, >6 días).

RUT-R10: Permitir guardar una rutina como plantilla con plantilla id (string) y aplicar plantilla a otros socios; al registrar una sesión el sistema debe crear una ejecución que guarde peso real (decimal, kg), series efectivas (int), RPE (decimal), y notas (string) por ejercicio vinculadas a fecha (date o datetime).

Planificación nutricional (NUT)

NUT-R11: Por socio se debe poder definir objetivo calórico diario (int, kcal, ≥800 ≤10000) y componer un plan por comidas (desayuno / almuerzo / cena / snacks) usando alimento id (string) y cantidad (decimal, porciones). El sistema calculará y mostrará kcal totales (decimal) y macros totales (objeto con proteína (decimal), grasa (decimal), carbohidrato (decimal)) por día y por semana (suma de 7 días).

NUT-R12: El sistema permitirá sustituir un alimento en una comida por otro del catálogo siempre que la sustitución mantenga el objetivo calórico diario dentro de ±5% (porcentaje, decimal). Si la sustitución rompe el margen, mostrar alerta (string) y proponer ajuste automático de porciones (decimal) para alcanzar el objetivo dentro del ±5%.

Progreso, analítica y recomendaciones (PRG)

PRG-R13: Registrar peso (decimal, kg), IMC (decimal, calculado), y medidas (objeto con cintura (decimal, cm), brazo (decimal, cm), etc.) con fecha (date). El sistema debe generar gráficas de evolución (peso e IMC) para rangos de fecha solicitados y permitir exportar la gráfica como PNG/PDF (archivo local).

PRG-R14: Implementar regla que detecte estancamiento de un levantamiento clave si en las últimas 12 semanas calendario no hay mejora. Cuando se detecte estancamiento, crear una recomendación automática que incluya: tipo (enum: variar volumen / variar intensidad / cambiar ejercicio), razón (string), y fecha generada (datetime). Registrar acción tomada por el entrenador (enum: Aceptada / Rechazada) junto con usuario (string) y timestamp (datetime).

Seguridad, respaldo y auditoría (SEC)

SEC-R1: Implementar autenticación por usuario (string) / password (string, guardada hasheada). Implementar autorización por rol (enum: Administrador / Entrenador / Recepción) mapeando permisos mínimos (lectura / escritura / borrado por recurso). Las sesiones deben expirar tras 12 horas de inactividad (int, horas).

SEC-R16: Permitir configurar backup interval (enum: Diario / Semanal), retention count (int, ≥1) y almacenar backups locales con backup id (string), timestamp (datetime) y checksum (string). Implementar función de restauración que valide checksum antes de aplicar y registre la operación en auditoría con usuario (string), fecha (datetime) y backup id (string). Registrar en auditoría los eventos: login success (event), login fail (event), membresía alta / baja / cambio (event) y catálogo modificación (event) (con usuario (string) / timestamp (datetime)).