UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Análisis técnico del aplicativo Healthwatcha .

ALUMNOS

Genis Cruz Lourdes Victoria

ASIGNATURA

Computo Móvil

PROFESOR

Marduk Pérez de Lara Dominguez

GRUPO

03

FECHA

23/05/2025

Semestre

2025-2

EQUIPO

Z

Ciudad Universitaria, Cd. Mx., 2025

Contenido

Reglas de negocio

A continuación, se mostrarán los lineamientos operativos y restricciones lógicas de nuestra aplicación. Todas estas reglas nos garantizan el funcionamiento adecuado de nuestro sistema para que tanto usuarios como entrenadores hagan un buen uso de la plataforma.

Para crear una cuenta, los usuarios deben tener al menos 18 años. Una vez registrados, pueden pausar su plan de entrenamiento una vez al mes sin perder su progreso. Los retos semanales están disponibles únicamente durante las primeras 24 horas después de su publicación. Con el objetivo de fomentar la participación, los usuarios solo pueden comentar en rutinas que hayan completado previamente y tienen permitido agregar hasta cinco amigos con quienes compartir su progreso.

En cuanto a la gestión de contenido, los entrenadores deben ser verificados por el administrador antes de poder publicar planes o rutinas. Todos los entrenadores deberán contar con los siguientes certificados:

- Certificación de entidad acreditada (ACE, NASM, ISSA, ACSM, CREF).
- Formación en primeros auxilios y RCP.
- Título en Ciencias del Deporte, Fisioterapia o afines.
- Registro en un colegio profesional o entidad reguladora.

Los usuarios pueden obtener puntos por completar rutinas, los cuales pueden ser canjeados por recompensas digitales dentro de la aplicación. Si un usuario no logra completar tres rutinas diarias consecutivas, se le sugerirá automáticamente un plan de entrenamiento más accesible acorde a su rendimiento. Para evitar abusos en el sistema de reportes, únicamente se permite enviar un reporte de error por rutina.

Además, las suscripciones se renuevan de forma automática, salvo que el usuario cancele con un mínimo de 24 horas de anticipación. Finalmente, las estadísticas individuales del usuario solo se actualizan cuando se ha registrado actividad real, ya sea completando el video de la rutina o confirmando la sesión mediante autoevaluación.

Requerimientos del sistema

Requerimiento	Categoría
RF-01	Autenticación y Registro
RF-02	Entrenamiento
RF-03	Seguimiento y Progreso
RF-04	Recordatorios y Motivación
RF-05	Contenido Premium
RF-06	Administración de Contenido
RF-07	Integración con Dispositivos
RNF-01	Rendimiento
RNF-02	Compatibilidad
RNF-03	Compatibilidad
RNF-04	Seguridad
RNF-05	Seguridad

RNF-06	Usabilidad
RNF-07	Disponibilidad
RNF-08	Rendimiento

Requerimientos funcionales (como historias de usuario del backlog)

Requerimiento	Descripción del requerimiento	Prioridad
RF-01	Como usuario nuevo, se podrá crear una cuenta usando un correo y una contraseña o se podrá acceder a través de cuenta Google.	Alta
RF-02	Como usuario, se podrá visualizar distintas rutinas.	Alta
RF-03	Como usuario, se podrá registrar el progreso (peso, medidas, repeticiones) para ver el avance.	Alta
RF-04	Como usuario, se podrá recibir notificaciones que lo motiven y le recuerden entrenar.	Media
RF-05	Como usuario premium, se podrá acceder a contenido exclusivo como planes avanzados y recetas.	Media
RF-06	Como administrador, se gestionará el contenido (rutinas, videos, planes).	Alta
RF-07	Como usuario, se podrá sincronizar el smartwatch para que la app registre automáticamente los datos biométricos (ritmo cardíaco, pasos, calorías, etc.).	Media

Requerimientos no funcionales

Requerimiento	Descripción	Prioridad
RNF-01	La app debe mostrar la rutina diaria en menos de 2 segundos.	Alta
RNF-02	Debe funcionar correctamente en iOS 13+	Alta
RNF-03	La app debe ser compatible con Apple Watch.	Alta
RNF-04	Las credenciales deben ser cifradas y almacenadas de forma segura.	Alta
RNF-05	Los datos biométricos sincronizados desde el Apple Watch deben cifrarse antes de almacenarse.	Alta
RNF-06	La interfaz debe ser intuitiva y usable con una sola mano.	Media
RNF-07	El sistema debe estar disponible al menos el 99.5% del tiempo mensual.	Alta
RNF-08	La sincronización con el Apple Watch no debe demorar más de 5 segundos.	Media

Alcance

Nuestra aplicación incluirá registro de usuario, rutinas en video, sincronización básica con el dispositivo Apple Watch. Por el momento no se tiene planeado incluir lo siguiente: entrenamiento con profesores en vivo, tienda, chat con entrenadores.

También consideramos que nuestro proyecto va a constar de cinco etapas, los ultimas dos etapas no son fundamentales. Sin embargo, nos ayudan a completar nuestro proyecto.

1. Etapa de lanzamiento (Surgimiento y creación del MVP)
   Esta esta es la primera fase de nuestro proyecto, la cual tiene como objetivo desarrollar el producto mínimo viable (MVP). El objetivo es probar si la idea funciona con lo siguiente:
   - Registro e inicio de sesión (correo y Google).
   - Visualización de rutinas en video.
   - Registro básico de progreso.
   - Notificaciones básicas.
   - Sincronización con Apple Watch (datos biométricos).
   - Acceso limitado a planes de entrenamiento.

   Como estamos en una fase de prueba es importante comprobar si la app funciona para los usuarios. Por ello se incluyeron funciones básicas, pero funciónales. En esta etapa se espera recibir feedback con el objetivo de hacer mejoras. Al mismo tiempo, estamos buscando alianzas estratégicas con gimnasios, universidades y centros deportivos que puedan apoyarnos en esta etapa inicial, ya que su respaldo nos ayudará a crecer y llegar a más personas.

   La app estará dirigida a usuarios jóvenes/adultos con acceso a dispositivos móviles y smartwatch (opcional), que busquen mejorar su condición física desde casa o el gimnasio.

2. Etapa de crecimiento y monetización
   Esta etapa buscara la formalización de nuestro producto, para ello vamos a aplicar diferentes métodos de monetización como lo son: las suscripciones premium, integración de contenido exclusivo y alianzas con marcas deportivas.

   Necesitaremos ampliar el equipo con especialistas en diseño digital, promoción y manejo de información. Para financiar este crecimiento, exploraremos opciones como buscar socios inversionistas, ingresar a programas de apoyo para startups.

   Funcionalidades previstas:
   - Acceso a contenido premium: recetas, rutinas especializadas.
   - Perfil público para compartir progreso con amigos.
   - Retos semanales y mensuales.
   - Visualización avanzada de estadísticas.
   - Evaluaciones automáticas con IA.
   - Implementación de pagos por suscripción (mensual/anual).

3. Etapa de consolidación y expansión
   En esta etapa, se buscará posicionar la app como una plataforma fitness integral a nivel nacional o regional. Se establecerán alianzas con entrenadores certificados y especialistas en nutrición, además de incluir contenido verificado y recursos profesionales.
   Además, mejoraremos la plataforma tecnológica para garantizar un rendimiento óptimo, implementaremos mayores medidas de protección para los datos sensibles de los usuarios (como información biométrica).

Funcionalidades previstas:

- Validación y perfiles públicos de entrenadores certificados.
- Integración completa con dispositivos de salud (Apple Watch).
- Rutinas grabadas por profesionales.
- Mejoras de rendimiento (procesamiento y disponibilidad del servidor).
- Versiones localizadas en otros idiomas.
- Cumplimiento de normas internacionales de protección de datos.

4. Etapa de diversificación e inteligencia predictiva
   Esta etapa tiene como objetivo la integración de inteligencia artificial que analice patrones de comportamiento para recomendar planes personalizados dinámicamente.
   Nuevas funcionalidades:
   - Algoritmo de ajuste automático de rutinas basado en rendimiento y fallos.
   - Recomendaciones automáticas basadas en datos biométricos e historial.

5. Etapa de internacionalización y ecosistema
   Finalmente, se puede llevar la app a otros países, con otros idiomas y reglas.
   Acciones posibles:
   - Traducción completa de la app (internacionalización).
   - Campañas de concientización en conjunto con gobiernos o ONGs.
   - Certificación de la app como herramienta de salud por instituciones médicas.

MVP (producto mínimo viable) de la app

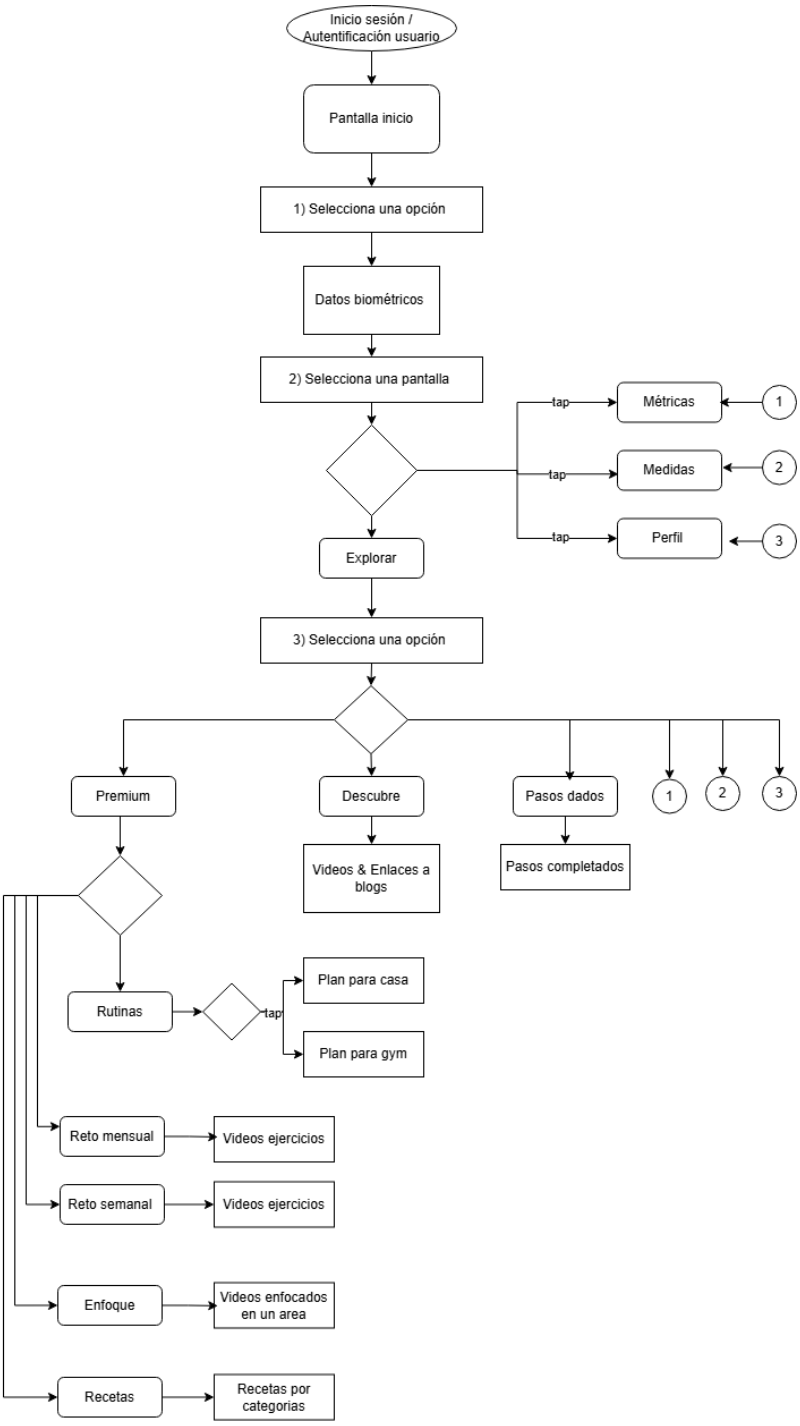
- Registro e inicio de sesión (correo y Google).
- Visualización de rutinas en video.
- Registro básico de progreso.
- Notificaciones básicas.
- Sincronización con Apple Watch (datos biométricos).
- Acceso limitado a planes de entrenamiento.

Wireframes de la app

A continuación, se anexó un link a una carpeta drive que contiene los wireframes de nuestra ampliación. Esto con el objetivo de evitar que la extensión del documento pase el número máximo de cuartillas que se nos pidieron.

https://drive.google.com/drive/folders/1UHUJw5x0XAGp09Kk6QXezeaIRhEKBfr8?usp=sharing

Explicación del flujo



En este diagrama podemos ver lo siguiente:

1) El primer paso es iniciar sesión a través de una contraseña y un correo o mediante la autentificación con una cuenta Google o Facebook.

2) Dentro de la pantalla de inicio vamos a poder visualizar diferentes datos biométricos mediante el tap.

3) El usuario puede elegir la pantalla que desea visualizar (Explorar, métricas, perfil, medidas)

4) Dentro de la pantalla Explorar se podrá redirigir a diferentes pestañas (Premium, Descubre, Pasos dados)

5) La pantalla Descubre nos muestra videos a rutinas gratis y enlaces a blogs.

6) La pantalla Pasos dados nos muestra los pasos completados hasta el momento.

7) Dentro de la pantalla Premium se podrá redirigir a diferentes pestañas (Rutinas, reto mensual, reto semanal, enfoque, recetas).

8) Dentro de la pantalla Reto mensual el usuario podrá visualizar la rutina a completar.

9) Dentro de la pantalla Reto semanal el usuario podrá visualizar la rutina a completar.

10) Dentro de la pantalla enfoque el usuario podrá visualizar distintas rutinas en forma de video según el are de enfoque.

11) Dentro de la pantalla recetas el usuario podrá visualizar distintas recetas.

12) Dentro de la pantalla rutinas el usuario seleccionar las siguientes opciones:

- Plan para casa: Rutinas que se pueden realizar sin equipo.
- Plan para gimnasio: Rutinas que se pueden realizar dentro del gimnasio.

Explicación de las pantallas

Human body (Home)

Función principal: Mostrar de forma rápida la información biométrica del usuario, incluyendo datos como las horas de sueño, la frecuencia cardíaca, la presión arterial, la saturación de oxígeno en sangre (SpO_2) y la cantidad de pasos dados.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Horas de sueño	Número (float)	Diario	Apple HealthKit / sensor	Consulta, sincronización automática
Frecuencia cardíaca	Número (int)	Cada minuto	Sensor smartwatch	Consulta, alerta si está fuera de rango
Presión arterial	Número (float)	Diario	Sensor smartwatch	Consulta
Saturación de oxígeno (SpO_2)	Porcentaje (int)	Diario	Sensor smartwatch	Consulta, comparación con rangos normales
Pasos dados	Número (int)	Diario	Sensor smartwatch	Consulta, acumulación

Análisis de datos

Servicio/API	Tipo de operación	Autenticación	Formato	Notas
Apple HealthKit / Google Fit	Consulta	Permisos del sistema (OAuth)	JSON	Acceso controlado por el usuario desde ajustes del sistema
API interna de salud	Registro/consulta	Token JWT	JSON	Guarda los datos históricos para análisis

Recorrido de los datos

1) Entrada (Input)
    - El usuario toca (tap) una parte del cuerpo en la pantalla para consultar un dato biométrico específico.
2) Procesamiento
    - La app identifica qué dato biométrico se solicita (por ejemplo, horas de sueño, frecuencia cardíaca, etc.).
    - Se realiza una consulta a HealthKit (framework de iOS) o a una base de datos local/remota (como Firebase) para obtener los datos correspondientes.
    - Se valida y transforma el tipo de dato si es necesario:
        - Horas de sueño → Float
        - Frecuencia cardíaca → Int
        - Presión arterial → Float
        - Saturación de oxígeno (SpO_2) → Int
        - Pasos dados → Int
3) Salida (Output)
    - Los datos procesados se muestran en la interfaz de usuario (UI) de la pantalla con visualizaciones ya sea por medio de gráficos o números destacados.

Explicación Almacenamiento

- Local:
    - Datos sincronizados en caché (duración 24 h) para mostrar sin conexión.
    - Guardados en SQLite o similar.
- Remoto:
    - Todos los datos de salud se almacenan en Firestore.
    - Esto permite estadísticas semanales o mensuales, y alertas automáticas si algo se detecta fuera del rango.

Interacciones especiales o gestos

- Tap sobre cualquier valor → muestra información rápida

Costos y requerimientos de los servicios

- Apple HealthKit: Este método es gratuito, pero requiere que el usuario otorgue los permisos necesarios. La autenticación se realiza a través de OAuth del sistema. Los datos se obtienen en formato JSON, como en el caso de Apple. Además, es indispensable que el usuario tenga instalada y activa la aplicación correspondiente para que el sistema funcione correctamente.
- API interna de salud: Si se decide guardar el historial en la nube, una opción viable es utilizar Firebase. Los costos dependerán del plan seleccionado y del número de lecturas y escrituras realizadas. La autenticación se puede manejar mediante JWT. Este enfoque ofrece beneficios como la posibilidad de realizar análisis, generar alertas, obtener estadísticas y contar con un respaldo seguro de los datos.

Medidas corporales

Función principal: Permite al usuario ingresar sus datos físicos actuales (peso y altura) y calcula automáticamente su grasa corporal aproximada. El objetivo es dar seguimiento a su progreso físico y salud.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Peso	Número (float)	Usuario manual	Entrada del usuario	Registro, actualización, consulta
Altura	Número (float)	Usuario manual	Entrada del usuario	Registro, actualización, consulta
Grasa corporal	Número (float)	Calculado	Cálculo automático	Consulta (basado en peso y altura)

Análisis de datos

Servicio/API	Tipo de operación	Autenticación	Formato	Notas
API interna de usuario	Registro, consulta, actualización	JWT	JSON	Almacenar en la nube
Sin conexión	Registro local	—	—	Permite almacenar localmente si no hay internet

Recorrido de los datos

1) Entrada (Input)
- El usuario ingresa manualmente su peso (kg) y su altura (cm) en los campos de entrada de la app.
2) Procesamiento

- La app valida que los valores ingresados sean números válidos (por ejemplo: sin letras, sin valores negativos).
- Se realiza un cálculo interno para estimar la grasa corporal usando la fórmula del Índice de Masa Corporal (IMC):
    - IMC = peso / (altura en metros)^2
- El resultado de grasa corporal es generado como un porcentaje estimado (Float).

3) Salida (Output)
    - Los valores de peso, altura y grasa corporal calculada se muestran en la interfaz de usuario, posiblemente en tarjetas o gráficos.

Explicación Almacenamiento

- Local:
    - Guarda los datos introducidos (peso, altura) en almacenamiento local (SQLite o SharedPreferences) para acceso rápido.
    - Ideal para dispositivos sin conexión o para evitar depender solo de la nube.
- Remoto:
    - Sincronizar respaldos, tendencias o reportes, se puede usar Firebase.
    - Solo se suben si el usuario da permiso.

Interacciones especiales o gestos

- Tap en cada campo para editar.

Costos y requerimientos de los servicios

- Local: Gratis, no requiere conexión.
- Remoto (si se usa): En caso de utilizar almacenamiento remoto, se puede emplear Firebase u otra plataforma similar. Los costos asociados varían según la frecuencia de uso del servicio. La autenticación puede realizarse mediante tokens JWT o a través de una cuenta de usuario. Es importante considerar que este enfoque requiere una conexión a internet para poder sincronizar los datos correctamente.

Métricas

Función principal: Muestra al usuario métricas de salud recolectadas automáticamente desde el sensor del Apple Watch. Los datos se presentan numéricamente y, cuando hay suficiente información histórica, se generan gráficas para visualizar los cambios en el tiempo.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Frecuencia cardíaca (bpm)	Número (float)	Diario	Sensor del Apple Watch	Consulta, sincronización
Presión arterial (mmHg)	Dos números Int (sistólica/diastólica)	Diario	Sensor del Apple Watch	Consulta, sincronización
Saturación de oxígeno(SpO_2)	Porcentaje (float)	Diario	Sensor del Apple Watch	Consulta, sincronización

Calorías quemadas	Número (float)	Diario	Sensor del Apple Watch	Consulta, sincronización
Duración del sueño	Double	Diario	Sensor del Apple Watch	Consulta, sincronización
Temperatura corporal	Número (float)	Diario	Sensor del Apple Watch	Consulta, sincronización

Servicios conectados y flujo de datos

Servicio/API	Tipo de operación	Autenticación	Formato	Notas
Apple HealthKit / Apple Watch API	Consulta automática	OAuth / Apple ID	JSON	Requiere permisos del usuario para acceder a datos
API interna del backend	Registro / consulta histórica	Token JWT	JSON	Para guardar en la nube o generar gráficas

Recorrido de los datos

1) Entrada (Input)

- La app accede a los datos recolectados por el Apple Watch mediante el framework HealthKit.
- Datos que se consultan:
    - Frecuencia cardíaca (*heart rate*) – Int
    - Presión arterial (*blood pressure*) – Float
    - Saturación de oxígeno (*oxygen saturation / SpO_2*) – Int
    - Calorías quemadas (*calories burned*) – Float
    - Duración del sueño (*sleep duration*) – Float (horas o minutos)
    - Temperatura corporal (*body temperature*) – Float (°C)

2) Procesamiento

- Se validan los datos obtenidos (que no estén vacíos o corruptos).
- Si hay datos disponibles, se transforman para visualización:
    - Se adaptan al formato necesario para gráficos o números representativos.
- Si hay un historial, se generan gráficas para mostrar la evolución de los valores a lo largo del tiempo.

3) Salida (Output)

- En la interfaz se muestran los valores actuales, como por ejemplo el último registro disponible de cada métrica. Además, si existe un historial suficiente de datos, se

presentan gráficas que permiten visualizar la evolución y las tendencias de los valores a lo largo del tiempo.

Explicación Almacenamiento

- Local:
  - Se almacena una copia de las métricas más recientes para visualización rápida.
  - También se guarda un historial limitado para generar gráficas básicas.
- Remoto:
  - Se sincroniza con la nube si el usuario lo permite.
  - Firebase.

Interacciones especiales o gestos

- Swipe hacia la izquierda/derecha: cambiar entre gráficas de diferentes métricas.
- Tap sobre un valor: abre gráfica expandida si hay historial.

Costos y requerimientos de los servicios

- Apple HealthKit / Watch API: Gratuita, pero requiere permisos y configuración adecuada.
- Firebase o backend propio: Los costos dependen de almacenamiento y frecuencia de sincronización.
- Autenticación: Requiere acceso a cuenta de usuario y token de acceso válido (JWT).

Perfil (Profile)

Función principal: Permite al usuario ver y editar su información personal.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
nombre	String	Permanente (editable)	Ingresado manualmente por el usuario	Consulta / Actualización
email	String	Permanente (editable)	Ingresado manualmente por el usuario	Consulta / Actualización
altura	Float	Permanente (editable)	Ingresado manualmente por el usuario	Consulta / Actualización
peso	Float	Permanente (editable)	Ingresado manualmente por el usuario	Consulta / Actualización

Explicación Almacenamiento

- Local:
  - Los datos se almacenan al guardar los cambios.
  - Datos almacenados localmente: name, email, height, weight.

- Remoto:
    - Los datos podrían sincronizarse usando Firebase.
    - Se usaría POST o PUT para actualizar, y GET para consultar.

Interacciones especiales o gestos

- Tap para activar campos de texto editables.

Registro usuario nuevo

Función principal: Permite al usuario crear una cuenta nueva, ingresando su información personal básica. Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
nombre	String	Permanente	Ingresado manualmente	Registro, Validación
apellido	String	Permanente	Ingresado manualmente	Registro, Validación
email	String	Permanente	Ingresado manualmente	Registro, Validación
contraseña	String	Permanente	Ingresado manualmente	Registro, Validación, Encriptación

Servicios conectados y flujo de datos

Servicio/API	Tipo de operación	Autenticación	Formato	Notas
API interna del backend	Registro	JWT	JSON	El backend valida y almacena los datos del nuevo usuario.
Firebase Authentication	Registro / login	Email/contraseña / OAuth	JSON	se manejan usuarios con validación automática.

Explicación Almacenamiento

- Local:
    - Se utiliza almacenamiento en memoria temporal con variables de estado para manejar la información del formulario.
    - Si se requiere mantener sesión iniciada, se guarda un token de autenticación seguro en el Keychai
- Remoto:
    - Los datos se envían a una API para su almacenamiento en la base de datos de usuarios.
    - Firebase, registra automáticamente el email con la contraseña, que se almacena encriptada en la nube.

Costos y requerimientos de los servicios

- Firebase Authentication: Gratuita hasta cierto límite mensual de usuarios activos.
- Autenticación: Se requiere un sistema seguro de encriptación de contraseñas o el uso de Firebase que lo maneja automáticamente.

Inicio sesión

Función principal: Permite al usuario ingresar a la app utilizando su correo electrónico y contraseña registrados previamente. También se ofrecen opciones de inicio de sesión mediante cuentas de Google o Facebook.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Email	String	Temporal	Ingresado manualmente	Autenticación
Contraseña	String	Temporal	Ingresado manualmente	Autenticación
Cuenta de Google	OAuth Token	Temporal	API de Google	Autenticación con terceros
Cuenta de Facebook	OAuth Token	Temporal	API de Facebook	Autenticación con terceros

Servicios conectados y flujo de datos

Servicio/API	Tipo de operación	Autenticación	Formato
API interna del backend	Autenticación	JWT / Email y contraseña	JSON
Firebase Authentication	Autenticación	Email/Contraseña / OAuth	JSON
Google Sign-In API	Login con terceros	OAuth	JSON
Facebook Login API	Login con terceros	OAuth	JSON

Explicación Almacenamiento

- Local:
  - Datos como el email ingresado pueden mantenerse temporalmente en variables locales para autocompletado.
- Remoto:
  - OAuth (Google/Facebook), verifica el token y registra o inicia sesión.

Interacciones especiales o gestos.

- Tap en botón de Google o Facebook para iniciar sesión.

Costos y requerimientos de los servicios

- Google/Facebook API: Gratuitas, pero requieren registro y configuración previa de la app con sus respectivas plataformas.

Pasos dados

Función principal: Muestra al usuario un gráfico circular indicando el porcentaje de pasos alcanzados respecto a una meta diaria.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Cantidad de pasos	Int	Diario	Sensor del Apple Watch	Consulta, sincronización
Porcentaje alcanzado	Float	Diario	Calculado localmente	Cálculo, visualización
Meta de pasos diaria	Int	Diario	Establecida por el usuario	Consulta, actualización

Servicios conectados y flujo de datos

Servicio/API	Tipo de operación	Autenticación	Formato	Notas
Apple HealthKit / Apple Watch API	Consulta	OAuth / Apple ID	JSON	Extrae el número de pasos dados en el día.
API interna del backend	Registro / consulta histórica	Token JWT	JSON	Guarda historial del usuairo.

Explicación Almacenamiento

- Local:
    - Se guarda el número de pasos del día más reciente para una carga más rápida.
    - Se calcula el porcentaje localmente con base en los pasos actuales y la meta.
- Remoto:
    - Se utiliza Firebase para guardar los pasos.

Costos y requerimientos de los servicios

- Apple HealthKit / Watch API: Gratuita, requiere permisos del usuario para acceder a los datos de actividad.
- Firebase / Backend propio: Costos dependen del uso, almacenamiento y número de sincronizaciones.

Descubre

Función principal: Muestra sugerencias de contenido para el usuario, incluyendo videos recomendados y artículos en formato de blog.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas

Videos	String	Periódico	Base de datos local o remota	Consulta
Blogs	String	Periódico	Base de datos local o remota	Consulta

Servicios conectados y flujo de datos

Servicio/API	Tipo de operación	Autenticación	Formato
Firebase	Consulta	OAuth / Apple ID	JSON

Explicación Almacenamiento

Remoto: Los enlaces o referencias al contenido se almacenan en la nube.

Interacciones especiales o gestos

- Tap sobre video o blog: abre el contenido en una nueva pantalla o navegador interno.

Explorar

Función principal: Es una pantalla índice que redirige al usuario a otras secciones de la app (como métricas, rutinas, premium, etc).

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Botones de acceso	UI Element	Permanente	UI local	Navegación

Interacciones especiales o gestos

- Tap en los botones para navegar a otras pantallas.

Premium

Función principal: Pantalla que permite al usuario acceder a otras funcionalidades exclusivas de la app, redirigiéndolo a otras vistas relacionadas con contenido o funciones para usuarios premium.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Botones de acceso	UI Element	Permanente	UI local	Navegación

Interacciones especiales o gestos

- Tap para navegar a otras secciones premium.

Costos y requerimientos de los servicios

- Para la integración de la suscripción, se requerirá conexión con StoreKit (iOS).

Rutinas

Función principal: Muestra planes de ejercicios dependiendo si son realizados en casa o en el gimnasio. Actualmente solo contiene botones que redirigen a otras pantallas con videos.

Información mostrada

Elemento	Tipo de dato	Vigencia	Origen del dato	Operaciones esperadas
Botón (Plan casa)	UI Element	Permanente	UI local	Navegación
Botón (Plan gimnasio)	UI Element	Permanente	UI local	Navegación

Interacciones especiales o gestos

- Tap en cada botón para acceder a una pantalla diferente (rutina en casa o rutina en gym).

Costos y requerimientos de los servicios

- No aplica actualmente. En caso de integrar rutinas personalizadas o recomendadas, se requerirá backend y posiblemente autenticación.

Compatibilidad de dispositivos y tamaños

Nuestra aplicación esta pensada para desarrollarse exclusivamente en dispositivos iPhone. Los tamaños de pantalla compatible serán todos los tamaños iPhone disponibles, es decir se tomarán en cuenta desde los modelos más chicos como lo son el iPhone SE hasta el Pro Max. La orientación soportada será la vertical mejor conocida como portrait. La orientación horizontal no será posible.

Sistemas operativos

En cuanto a la compatibilidad mínima requerida se necesita de una versión iOS 16 ya que esta versión nos permite trabajar con las funciones de Apple HealthKit de forma estable. Sin embargo, se recomienda tener una versión iOS 17 o superior para una mejor compatibilidad.

Decidimos excluir versiones anteriores debido a que estas versiones no nos garantizan un soporte completo de los sensores del apple watch. Además, algunas funciones pueden generar problemas de compatibilidad con SwiftUI.

Sensores

La información biométrica se estará obteniendo de los sensores del Apple Watch. Los sensores utilizados incluyen el de frecuencia cardíaca, presión arterial (en modelos compatibles), saturación de oxígeno en sangre (SpO_2), temperatura corporal, registro de sueño y sensor de movimiento, entre otros. Todos estos datos se obtienen automáticamente a través del ecosistema de Apple HealthKit, siempre que el usuario otorgue los permisos necesarios. La conexión entre el iPhone y el Apple Watch se realiza vía Bluetooth, y los

datos se sincronizan automáticamente. Para poder acceder a esta información vamos a estar utilizando el framework HealthKit.

Demo

(Da clic en la imagen)

Lenguajes de programación y herramientas

Nuestro programa se esta desarrollando en Swift y estamos utilizando el framework SwiftUI el cual nos ayudara a la construcción de la interfaz. Swift es un lenguaje nativo de Apple que es altamente compatible con Xcode.

En cuanto a la recolección de datos de salud, se emplea el framework HealthKit, que requiere declarar explícitamente los tipos de datos a acceder y solicitar permisos del usuario en tiempo de ejecución. También se contempla el uso del WatchConnectivity Framework para la sincronización de datos entre el iPhone y el Apple Watch.

- **Lenguaje principal:** Swift.
- **Framework para interfaces:** SwiftUI.
- **Entorno de desarrollo:** Xcode
- **HealthKit:** Nos servirá para acceder a los datos de salud del Apple Watch.
- **WatchConnectivity:** Nos ayudara para sincronización entre iPhone y Apple Watch.
- **Firebase :** Como ya se mencionó anteriormente estaremos utilizando firebase para el almacenamiento remoto y la autenticación.

Requisitos para la publicación

Para poder publicar la app en la App Store, debemos considerar varios requisitos importantes. Primero, es obligatorio contar con una cuenta Apple Developer Program. Además, debido a que estaremos usando datos sensibles de salud, Apple nos exige que la app incluya una política de privacidad clara, que detalle cómo se almacenan, procesan y protegen los datos del usuario.

También debemos declarar el uso de HealthKit en nuestro archivo de configuración Info.plist y debemos asegurarnos de cumplir con las políticas de uso de datos de salud impuestas por Apple.

Durante el proceso de publicación, Apple nos exige una justificación del uso de sensores y APIs privadas,y como los relacionados con el Apple Watch. También se deben seguir las guías de diseño de interfaces (Human Interface Guidelines) y garantizar que la experiencia de usuario sea clara, intuitiva y sin funciones engañosas.

Por último, la app será sometida a una revisión manual por parte del equipo de la App Store, quienes evaluarán que cumpla con todas las normativas técnicas, legales y de experiencia de usuario. Cumplir con estas políticas es indispensable para que la aplicación sea aprobada y publicada exitosamente en la tienda.

Equipo de trabajo y roles que intervienen

Para el desarrollo e implementación de esta aplicación requerimos de una serie de personas los cuales se van a enfocar en distintas áreas del software. Se cuenta con un equipo de desarrolladores iOS, encargados de programar la aplicación utilizando Swift y SwiftUI, así como integrar frameworks como el HealthKit y WatchConnectivity. También se tienen a diseñadores UX/UI, quienes se encargan de crear las interfaces amigables y funcionales, alineadas con las guías de Apple.

Además, se tiene a un ingeniero de backend el cual es necesario para implementar el almacenamiento remoto y autenticación, mediante el uso de Firebase, lo cual implica la creación de servicios web (APIs), manejo de bases de datos y protocolos de seguridad. Asimismo, está el especialista en integración de dispositivos el cual se encarga de asegurar la correcta comunicación entre el iPhone y el Apple Watch, así como del manejo de los sensores.

Por último, se requiere de un tester para la detección errores, y la verificación del funcionamiento adecuado de la app el cual nos asegure una experiencia estable. En etapas finales, un gestor de proyecto o product owner supervisa la organización, los tiempos y la coordinación general del equipo.

Nombre	Rol	Tareas
Victoria Genis	Desarrollador iOS	• Programación en Swift/SwiftUI. • Integración de HealthKit, conexión con Apple Watch.
Victoria Genis	Diseñador UX/UI	• Creación de wireframes, prototipos, elección de colores, tipografía y navegación.
Sandra Neri	Ingeniero Backend	• Desarrollo de APIs • Configuración de base de datos. • Autenticación e integración con Firebase.
Sandra Neri	Especialista en integración	• Configuración de WatchConnectivity. • Pruebas con sensores del Apple Watch.
Victoria Genis & Sandra Neri	QA / Tester	• Pruebas funcionales. • Reporte de errores. • Verificación de rendimiento
Victoria Genis & Sandra Neri	Gestor de proyecto	• Planificación de tareas.

Plan de trabajo

TAREAS	Febrero				Marzo				Abril				Mayo			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Creación de wireframes, prototipos, elección de colores, tipografía y navegación.																
Programación en Swift/SwiftUI.																
Integración de HealthKit, conexión																
Configuración de base de datos.																
Autenticación e integración con Firebase.																
Configuración de WatchConnectivity																
Pruebas con sensores del Apple																
Pruebas funcionales																
Reporte de errores.																

Figura 1 Diagrama de Gantt

Análisis de seguridad de la información

Se maneja información sensible, dentro de las cuales destacamos lo siguiente:

- Datos personales: Como lo son el nombre, correo, altura, peso, etc
- Datos de salud: Ritmo cardíaco, presión arterial, SpO_2, sueño, calorías, temperatura
- Credenciales de acceso: Email y contraseña

Todos los datos anteriores es información que se deberá proteger a toda costa. Para ello es esencial evitar el acceso no autorizado a los datos personales y de salud. Asimismo, esta información no podrá ser alterada o falsificada. También los dtaos deben estar disponibles cuando el usuario los necesite. Por último, deberemos proteger las credenciales de inicio de sesión y evitar suplantaciones. Para ello se proponen los siguientes mecanismos:

Mecanismo	Descripción
Cifrado de datos en tránsito	Uso de HTTPS/TLS para todas las comunicaciones entre la app, la nube (Firebase) y APIs como HealthKit.
Cifrado de datos en reposo	Cifrado de la base de datos local y almacenamiento seguro en Firebase.
Autenticación segura	Implementación de login mediante email y contraseña con validación segura (Firebase Auth o backend). También se permite autenticación con cuentas verificadas como Google o Facebook.
Tokens JWT	Para manejar sesiones de usuario de forma segura y controlar el acceso a datos desde el backend.
Permisos granulares	Uso de permisos explícitos para acceder a datos de HealthKit. El usuario debe aceptar qué datos se pueden leer.
Almacenamiento sensible limitado	Se evita almacenar contraseñas o datos sensibles directamente en el dispositivo sin cifrado.
Control de acceso	Validaciones para que cada usuario solo acceda a sus propios datos.

Estimaciones de tiempo de desarrollo y costos

Concepto	Costo estimado (USD)	Detalles
Desarrollador iOS	$2,400 - $4,800	1 persona, 1-2 meses de trabajo. Puede variar.
Diseñador UI/UX	$600 - $1,200	Diseño de pantallas, experiencia de usuario, flujo visual, adaptaciones.
Suscripción Apple Developer	$99 USD / año	Obligatorio para publicar en la App Store.
Firebase	$0 - $60 / mes	Solo si se supera el plan gratuito. Para autenticación y base de datos.
Licencias o software adicional (Figma)	$0 - $120	Para el diseño.
Pruebas en TestFlight / QA	$300 - $600	Para pruebas, correcciones y mejoras antes de publicar.
Total estimado	$3,399 – $6,879	Dependiendo del equipo, herramientas y tiempo de desarrollo.

Referencias

Martins, J. (2025, February 2). Diagrama de Gantt: qué es y cómo crear uno con ejemplos [2025] • Asana. Retrieved May 21, 2025, from Asana website: https://asana.com/es/resources/gantt-chart-basics

Atlassian. (2023). Explicación de los diagramas de Gantt [y cómo crear uno] | Atlassian. Retrieved May 21, 2025, from Atlassian website: https://www.atlassian.com/es/agile/project-management/gantt-chart

Team Asana. (2025, February 11). Roles del equipo: 9 tipos de roles para crear un equipo bien equilibrado [2025] • Asana. Retrieved May 21, 2025, from Asana website: https://asana.com/es/resources/team-roles

Firebase. (2024). Retrieved May 21, 2025, from Firebase website: https://firebase.google.com/?gad_source=1&gad_campaignid=12302357971&gbraid=0AAAADpUDOgD9RPR0r0IdBoSRR82wqmES&gclid=CjwKCAjw87XBBhBIEiwAxP3_A-yxOWce5-vrwbpXaM2t1e_tJmEPhG8p57VsEfA9KM6rEVEgJKhqEBoCDnkQAvD_BwE&gclsrc=aw.ds&hl=es-419

Política de privacidad de Apple - Apple. (2025). Retrieved May 21, 2025, from Apple Legal website: https://www.apple.com/mx/legal/privacy/es-la/

App Review Guidelines - Apple Developer. (2025). Retrieved May 21, 2025, from Apple Developer website: https://developer.apple.com/app-store/review/guidelines/

Apple Inc. (2025). Swift.org. Retrieved May 21, 2025, from Swift.org website: https://www.swift.org/getting-started/

Swift - Apple Developer. (2025). Retrieved May 21, 2025, from Apple.com website: https://developer.apple.com/swift/