UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

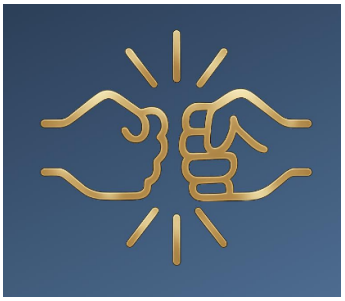
Semestre 2025-2

Cómputo Móvil

Grupo. 3

TRABAJO FINAL

YaQuedamos



Reconectando a las personas mediante actividades auténticas en la comunidad

Equipo: HummingDevs

- Laparra Miranda Sandra – Desarrolladora Frontend – No. de cuenta: 311243563
- Sánchez Manzano Mariana – Desarrolladora Backend – No. de cuenta: 411026622
- Sánchez Sánchez Santiago – Desarrollador SwiftUI – No. de cuenta: 318505725

Fecha de entrega: 23 de mayo de 2025 – 18:00 hrs

Resumen Ejecutivo

La aplicación "YaQuedamos" surge como respuesta a una problemática social actual: a pesar de la hiperconectividad virtual, muchas personas experimentan una profunda sensación de soledad. YaQuedamos busca reconectar a los usuarios en el mundo real mediante actividades que van desde deportes hasta encuentros culturales y eventos comunitarios, creando así lazos auténticos y significativos.

Esta plataforma no solo facilita encuentros presenciales, sino que también fomenta la creación de comunidades proactivas que promueven el bienestar común y el impacto social tangible. A través de perfiles genuinos, grupos temáticos diversos y eventos colaborativos, YaQuedamos transforma la interacción digital en relaciones reales y constructivas dentro de la comunidad.

La aplicación está desarrollada utilizando Swift con SwiftUI, permitiendo un diseño responsivo, moderno y optimizado para dispositivos Apple. Actualmente es compatible con dispositivos iPhone desde iOS 17 en adelante, cubriendo un amplio rango del mercado móvil Apple. Usa funciones nativas como mapas (Apple Maps), notificaciones push, almacenamiento seguro con Keychain para credenciales y JWT para autenticación segura. La arquitectura escalable permite adaptarse fácilmente al crecimiento en usuarios y funcionalidades adicionales, incluyendo futuras integraciones con redes sociales externas y sistemas avanzados de seguridad como la autenticación multifactor (2FA).

Detalle de Requerimientos

Reglas de Negocio

- Los usuarios deben poder registrarse e iniciar sesión de forma segura, usando credenciales únicas.
- Cada usuario tendrá un perfil donde podrá indicar sus intereses, actividades favoritas y nivel de experiencia.
- Se pueden crear grupos temáticos y eventos presenciales públicos o privados, donde otros usuarios pueden participar.
- Las actividades están categorizadas (deportivas, culturales, sociales, etc.) y pueden tener distintos niveles de habilidad y requisitos.
- El sistema permite reportar problemas comunitarios a través de una sección colaborativa.
- YaQuedamos no tiene fines de lucro: la funcionalidad principal es social y comunitaria.

Requerimientos Funcionales (Backlog Técnico)

ID	Historia de Usuario	Épica	Prioridad	Estimación (puntos de historia)	Sub ID	Subtareas
HU-001	Como nuevo usuario, quiero registrarme con mi correo, número de teléfono y nombre, para poder crear una cuenta.	Registro y Autenticación	Alta	5	HU01.1	El formulario de registro debe validar campos obligatorios (nombre, email, contraseña).
					HU01.2	La contraseña debe almacenarse con hashing (bcrypt o similar).
					HU01.3	Se debe enviar correo de verificación para activar la cuenta.
HU-002	Como usuario registrado quiero iniciar sesión con mis credenciales o red social, para acceder a mi cuenta	Registro y Autenticación	Alta	3	HU02.1	El sistema debe validar email y contraseña con autenticación segura y Keychain
					HU02.2	Se debe generar y devolver un token JWT válido al iniciar sesión.
HU-003	Como usuario, quiero recuperar mi contraseña, para poder acceder en caso de olvido.	Registro y Autenticación	Media	3	HU03.1	El usuario debe poder solicitar recuperación de contraseña mediante email.
					HU03.2	Se debe generar un token temporal y enviar por correo.
					HU03.3	El token debe tener expiración (1 día).
HU-004	Como usuario, quiero editar mi nombre, imagen, intereses y ubicación para personalizar mi cuenta	Perfil de Usuario	Media	5	HU04.1	El usuario debe poder editar su nombre, bio, foto y ubicación.
					HU04.2	Debe mostrarse nombre, foto, bio y reuniones públicas.
					HU04.3	Las actualizaciones deben reflejarse de forma inmediata en su perfil.
HU-005	Como usuario, quiero poder eliminar mi cuenta para dejar de usar la app.	Perfil de Usuario	Baja	2	HU05.1	El usuario debe poder eliminar su cuenta con confirmación doble.
					HU05.2	Todos los datos personales deben eliminarse o anonimizarse.
HU-007	Como usuario, quiero poder ver estadísticas, así como los logros y puntos obtenidos en la aplicación	Perfil de Usuari	Baja	3	HU07.1	El backend debe realizar los cálculos de las estadísticas y mostrar en la pantalla principal
					HU07.2	El backend debe actualizar la relación en la tabla de logros.
HU-008	Como usuario, quiero interctuar interacturar con mi feed.	Interacción Social y	Media	3	HU08.1	El usuario puede comentar y dar "me gusta" a publicaciones y actividades.
					HU08.2	El backend debe registrar los comentarios y números de likes por publicación
HU-009	Como usuario, quiero reportar necesidades comunitarias	Interacción Social y Comunidad	Baja	2	HU09.1	El usuario puede reportar necesidades comunitarias (ej. mantenimiento de parques) como una categoría especial
					HU09.2	El usuario puede votar por reportes comunitarios para priorizarlos por impacto.
HU-010	Como usuario, quiero poder descubrir actividades por medio del buscador	Actividades y Reuniones	Alta	5	HU10.1	El usuario puede buscar actividades por nombre, categoría o cercanía, o uso de palabras clave
					HU10.2	Las actividades deben ordenarse por fecha descendente.
HU-011	Como usuario organizador, quiero crear una actividad indicando lugar, fecha y tipo para	Actividades y Reuniones	Alta	5	HU11.1	El formulario debe validar título, descripción, ubicación y fecha.
					HU11.2	El evento debe guardarse en la base de datos y mostrarse en el feed.
HU-012	Como usuario, quiero incluir imágenes o videos al crear una reunión para mostrar de qué trata.	Actividades y Reuniones	Media	3	HU12.1	El usuario puede subir una o más imágenes o vídeos al evento.
					HU12.2	Los archivos deben almacenarse en un bucket seguro (S3 o similar).
HU-013	Como usuario, quiero unirme a reuniones abiertas que me interesen para poder asistir.	Actividades y Reuniones	Alta	3	HU13.1	El usuario debe poder confirmar su asistencia a una reunión.
					HU13.2	El sistema debe registrar el usuario en la lista de asistentes.
HU-014	Como usuario, quiero poder salirme de una reunión si ya no puedo asistir.	Actividades y Reuniones	Media	2	HU14.1	El usuario debe poder cancelar su asistencia antes del evento.
					HU14.2	El sistema debe actualizar la lista de asistentes y liberar el cupo.
HU-015	Como usuario, quiero ver todos los datos de una reunión antes de decidir si me uno.	Actividades y Reuniones	Alta	3	HU15.1	Debe mostrarse nombre, descripción, fecha, hora, ubicación y organizador.
					HU15.2	Botón para unirse debe estar disponible si hay cupo y no es pasada.
HU-016	Como usuario, quiero ver actividades cercanas en el mapa o feed para elegir las más	Geolocalizació n	Alta	4	HU16.1	El sistema debe solicitar permiso de geolocalización al usuario.
					HU16.2	Se deben mostrar actividades en un radio de 10 km desde su ubicación.
HU-017	Como usuario, quiero dar acceso a mi ubicación solo si deseo que se usen datos de proximidad.	Geolocalizació n	Alta	2	HU17.1	La app debe pedir acceso a ubicación con modal del sistema operativo.
					HU17.2	Debe manejarse el rechazo de permiso con mensaje alternativo.
HU-018	Como usuario, quiero poder buscar actividades por nombre para encontrar algo específico.	Búsqueda y Filtrado	Media	3	HU18.1	El usuario puede buscar actividades por nombre o categoría.
					HU18.2	La búsqueda debe hacerse contra nombre y descripción de actividades.
HU-019	Como usuario, quiero aplicar filtros para ver solo las actividades que me interesan.	Búsqueda y Filtrado	Alta	4	HU19.1	Debe haber filtros por categoría, ubicación y fecha.
					HU19.2	Los filtros deben combinarse y actualizar los resultados dinámicamente.
HU-020	Como usuario, quiero recibir notificaciones sobre eventos, invitaciones y recordatorios.	Notificaciones	Alta	3	HU20.1	El usuario debe recibir notificaciones push de eventos próximos o invitaciones, así como de logros desbloqueados.
					HU20.2	Se debe poder guardar eventos en el calendario personal del dispositivo.
					HU20.3	El usuario puede recibir de recordatorios automáticos configurables.
					HU20.4	El usuario debe poder activarse o desactivarse por tipo de actividad.
HU-021	Como usuario, quiero ganar puntos y logros por organizar y asistir a actividades, para motivarme a participar más	Gamificación	Media	5	HU21.1	Cada vez que el usuario organiza o asiste a una actividad, se le asignan puntos.
					HU21.2	Al alcanzar ciertos hitos (e.g. 5 eventos), se desbloquean logros y se sube de nivel en el perfil según participación.
					HU21.3	El usuario puede recibir Insignias para organizadores de eventos populares.
					HU21.4	El usuario recibe notificaciones al desbloquear logros.
					HU21.5	Los logros deben mostrarse en el perfil con nombre, icono y fecha.

Ver Anexo A, , dentro de la carpeta trabajo_final/anexos, para una mejor referencia.

Requerimientos No Funcionales

- **Seguridad:** Uso de JWT para autenticación, Keychain para almacenamiento local seguro, y cifrado de datos personales. Validación de sesión con expiración automática. Protección contra ataques XSS y CSRF.
- **Usabilidad:** Interfaz limpia, moderna y fácil de usar. Navegación intuitiva y consistente. Modo oscuro para accesibilidad.
- **Disponibilidad:** Alta disponibilidad esperada gracias a tecnologías escalables y servicios en la nube. Uso de balanceadores de carga y backups automáticos.
- **Escalabilidad:** Estructura preparada para crecimiento tanto en usuarios como en funcionalidades, con arquitectura desacoplada.
- **Compatibilidad:** Diseño adaptable para dispositivos móviles iOS versión 17 o superior. Preparado para expansión futura a Android mediante Flutter o React Native.
- **Mantenibilidad:** Código estructurado con SwiftUI y componentes modulares reutilizables. Documentación interna para facilitar el mantenimiento.

- **Rendimiento:** Optimización de vistas, uso de listas virtualizadas, carga diferida de contenido multimedia.
- **Internacionalización:** Preparado para incorporar traducciones múltiples y adaptar formatos de fecha y hora según la región del usuario.
- **Analítica y monitoreo:** La app se integrará con servicios de analítica para recolectar métricas de participación, retención y rendimiento.
- **Resiliencia:** La app mostrará mensajes informativos si no hay conexión y conservará datos relevantes en caché.
- **Actualización continua:** Su arquitectura modular permitirá aplicar cambios o mejoras sin afectar la experiencia del usuario final.
- **Accesibilidad:** Compatibilidad con VoiceOver, tamaño de fuente dinámico, y contraste accesible para personas con baja visión.

Alcance y MVP (Producto Mínimo Viable)

Alcance del Proyecto

YaQuedamos está diseñada como una plataforma móvil cuyo objetivo es reconectar a las personas a través de experiencias presenciales organizadas en la comunidad. Su alcance contempla:

- Registro y autenticación segura de usuarios.
- Gestión de perfiles personales auténticos con intereses e historial.
- Creación y participación en actividades presenciales por categorías.
- Formación y administración de grupos temáticos.
- Interacciones sociales básicas (me gusta, comentarios, compartir).
- Visualización de notificaciones y eventos desde un calendario.
- Reporte y votación de necesidades comunitarias.
- Incorporación de elementos de gamificación y reputación.
- Sistema de moderación y comunidad segura.

Producto Mínimo Viable (MVP)

Para una primera versión funcional y validable de la app, se han priorizado los siguientes elementos como MVP:

- Registro y login de usuarios (Swift + JWT + Keychain).
- Creación y visualización de actividades (formulario completo + mapa).
- Filtro por categoría e inscripción en eventos.
- Vista de perfil editable con intereses y foto.
- Feed inicial con publicaciones y comentarios dummy.
- Navegación con pestañas principales: Inicio, Actividades, Perfil.
- Animaciones básicas y splash screen funcional.
- Soporte para modo oscuro y dispositivos iOS 17+.

Este MVP permite probar el flujo principal de valor de YaQuedamos: crear, descubrir y unirse a actividades reales, generando un puente entre lo digital y lo humano desde la primera interacción.

Wireframes

La siguiente sección presenta los wireframes que representan la arquitectura visual de YaQuedamos. Se diseñaron con el objetivo de ilustrar tanto la estética como la funcionalidad de cada pantalla clave. El diseño se centró en simplicidad, accesibilidad y consistencia en la navegación. Ver ANEXO B, disponible dentro de la carpeta trabajo_final/anexos/.

Pantallas incluidas y descripción funcional:

1. **SplashScreenView**: Pantalla de bienvenida animada con logotipo de la app. Dato sin persistencia. Funciona como puente visual antes de determinar si el usuario está logueado.
2. **WelcomeView**: Pantalla que permite elegir entre registrarse o iniciar sesión. No guarda datos localmente, solo navegación.
3. **MainTabView**: Vista con navegación por pestañas entre Inicio, Actividades y Perfil. Permite acceso rápido al contenido principal.
4. **FeedView**: Muestra publicaciones y notificaciones. Datos generados localmente, pensados para futuras consultas a un servidor. Operaciones: lectura, like, comentarios.
5. **ActivitiesView**: Muestra lista filtrable de actividades. Operación de consulta. Datos en memoria local para esta versión.
6. **CreateActivityView**: Formulario para crear actividades. Permite ingresar nombre, lugar, aforo, requisitos, etc. Datos de entrada que serán enviados como solicitud de registro.
7. **ActivityDetailView**: Muestra información detallada de una actividad. Incluye botón para unirse. Opera sobre datos consultados desde la actividad seleccionada.
8. **ProfileView**: Visualiza nombre, intereses, historial. Consulta datos del usuario actual. Puede mostrar logros.
9. **EditProfileView**: Permite editar nombre, intereses, visibilidad y subir foto. Datos de entrada. Operación de actualización.
10. **LoginView**: Campo de correo y contraseña. Datos sensibles. Autenticación vía JWT. Almacenamiento temporal.
11. **SignUpView**: Registro de nombre, correo, contraseña. Autenticación e inicio de sesión. Datos con persistencia parcial (en servidor).
12. **ReportesView**: Vista donde se listan reportes comunitarios. Consulta y votación. Datos temporales, se prevé backend.
13. **GroupView**: Administración de grupos. Crear, unirse, ver miembros. Consulta y operaciones de registro/actualización.
14. **GamificationView**: Vista que muestra logros, puntos y niveles. Datos de solo lectura, calculados desde backend o local.
15. **NotificationSettingsView**: Configuración de alertas y preferencias. Datos persistentes, almacenados localmente.

> Los wireframes completos se incluyen en ANEXO C, dentro de la carpeta trabajo_final/anexos/, en formato PDF para asegurar su visibilidad en alta resolución en caso de requerirse.

Flujo de Pantallas

A continuación se describe el flujo de navegación de YaQuedamos, desde que el usuario abre la app hasta las acciones principales dentro del sistema:

1. **SplashScreenView**: Al abrir la app, el usuario ve una animación breve con el logotipo.
    - Si está autenticado, avanza automáticamente a MainTabView.
    - Si no ha iniciado sesión, lo redirige a WelcomeView.
2. **WelcomeView**: Pantalla de entrada con dos botones:
    - "Iniciar sesión" → abre LoginView.
    - "Crear cuenta" → abre SignUpView.
3. **LoginView / SignUpView**:
    - Tras completar el formulario, el usuario es autenticado.
    - Si tiene éxito, es llevado a MainTabView.
4. **MainTabView**: Contenedor con tres pestañas principales:
    - FeedView: Noticias y notificaciones.
    - ActivitiesView: Lista de actividades filtrables, acceso a ActivityDetailView.
    - ProfileView: Información personal y opciones de configuración.
5. **Desde FeedView**:
    - El usuario puede interactuar con publicaciones o ir a notificaciones relacionadas a eventos o grupos.
6. **Desde ActivitiesView**:
    - Puede acceder a CreateActivityView para registrar una nueva.
    - Al seleccionar una actividad, entra a ActivityDetailView para ver más información y unirse.
7. **Desde ActivityDetailView**:
    - El usuario puede ver ubicación, requisitos y detalles.
    - Puede unirse o abandonar el evento.
8. **Desde ProfileView**:
    - Accede a EditProfileView para modificar sus datos.
    - Accede a GamificationView para revisar logros.
    - Accede a NotificationSettingsView para configurar alertas.
    - Desde aquí también puede ingresar a GroupView para gestionar grupos.
9. **Desde cualquier pestaña**, mediante accesos rápidos o notificaciones:
    - Puede acceder a ReportesView para ver necesidades comunitarias.

En la Figura 1 y 2 se puede ver el flujo anteriormente descrito. Este flujo permite una navegación simple y clara, asegurando que cada vista cumpla con su propósito y que los datos relevantes fluyan entre vistas de forma lógica y eficiente.

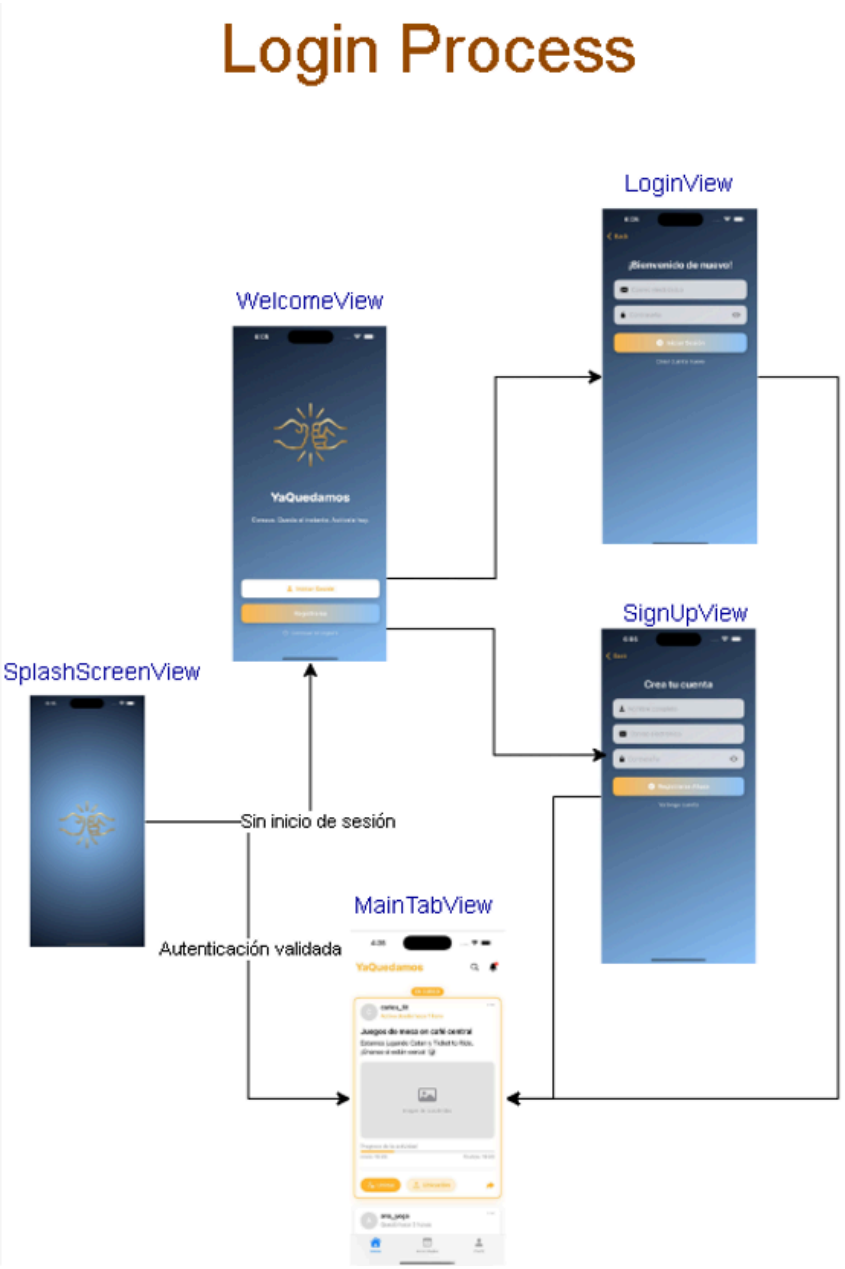


Figura 1: Flujo de pantallas del proceso de Login

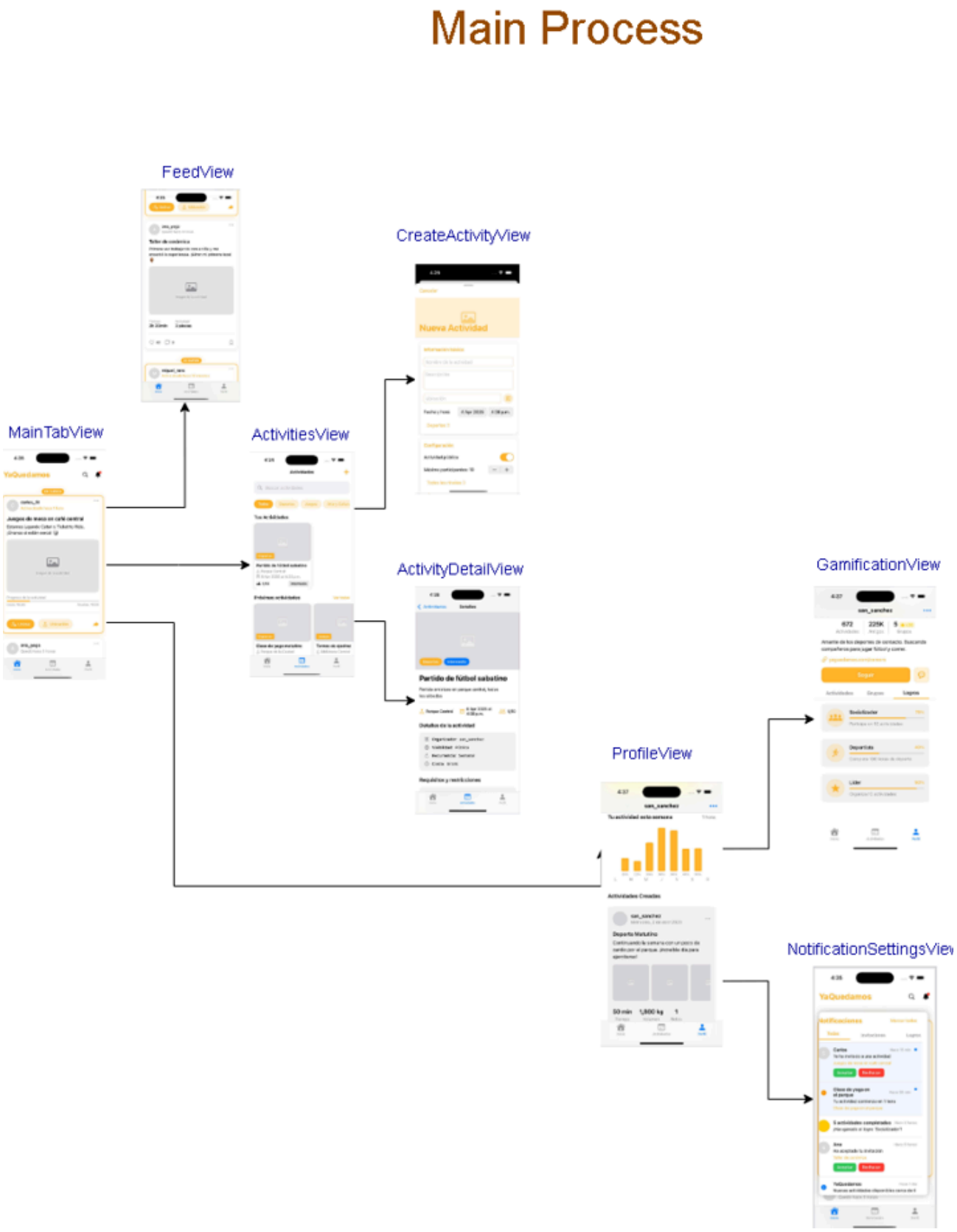


Figura 2: Flujo de pantallas del proceso principal

Documentación Técnica por Pantalla

En el ANEXO C se presentan los wireframes que representan la arquitectura visual de YaQuedamos. Se diseñaron con el objetivo de ilustrar tanto la estética como la funcionalidad de cada pantalla clave. El diseño se centró en simplicidad, accesibilidad y consistencia en la navegación.

Análisis de Datos

Tipos de Datos Clave

1. **Usuarios:**
   - Atributos: userID, name, email, passwordHash, profileImage, interests, visibility, score, level
   - Sensibles: contraseña (hash), visibilidad del perfil, token JWT

2. **Actividades:**
   - Atributos: activityID, name, description, location, date, category, organizerID, attendees[], maxAttendees, requirements[], hasCost, costAmount
   - Relacionados con: Usuarios, Grupos, Mapas

3. **Publicaciones e Interacciones:**
   - Atributos: postID, authorID, image, likes, comments[], timestamp

4. **Grupos y reportes comunitarios:**
   - Atributos: groupID, title, description, members[], isPublic, reportID, votes, status

5. **Notificaciones y preferencias:**
   - Atributos: notificationID, type, message, readStatus, settings: {pushEnabled, reminderTime}

Flujo y Origen de Datos

- **Origen:** Los datos provienen directamente del usuario (formulario), de sensores (ubicación), o del backend (actividades, interacciones).

- **Destino:** Almacenamiento seguro en bases de datos en la nube, sincronizados con vistas SwiftUI a través de modelos locales.

- **Flujo típico:** Usuario crea actividad → Se guarda en backend → Aparece en feed → Otros usuarios consultan y se inscriben → Se actualizan contadores y participantes.

Frecuencia de Actualización

- **Actividades y feed:** Se actualizan bajo demanda (pull) y con notificaciones push opcionales.
- **Perfil y configuración:** Bajo edición directa por el usuario.

- **Interacciones (likes, comentarios):** Tiempo real o casi inmediato mediante WebSockets o polling.

Modelado Relacional de la BD

A continuación se muestra el modelo relacional de la BD en la que se aprecian los principales componentes y datos que se utilizan en la aplicación.

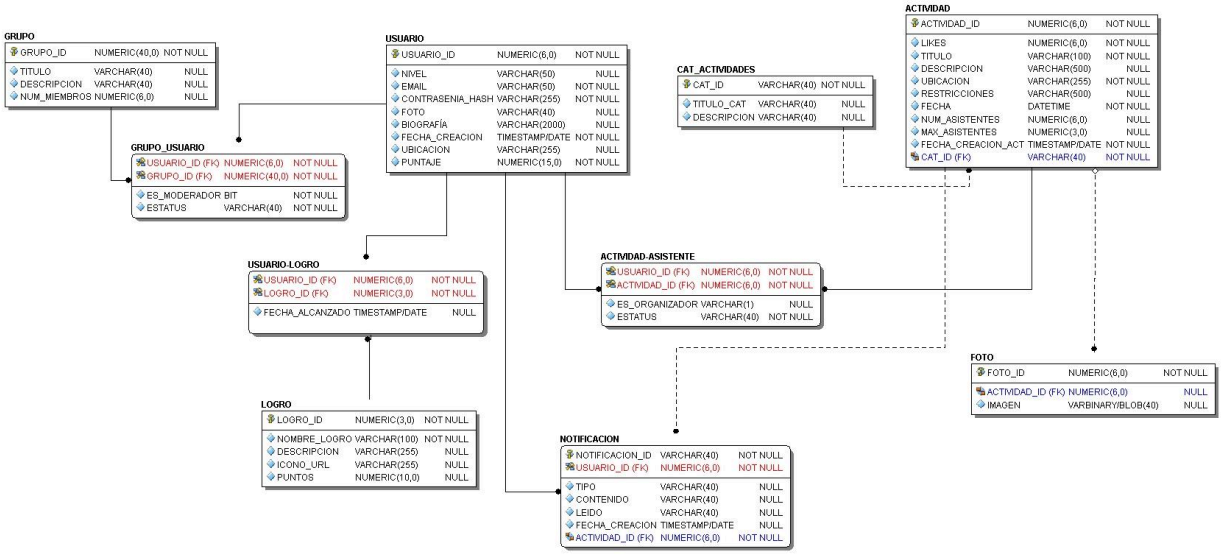


Figura 3: Modelo relacional de la BD. Para mejor definición, ver imagen en trabajo_final/anexo “Modelo Relacional B”

Calidad e Integridad

- Validaciones de entrada: campos requeridos, tipos y formatos.
- Hash y cifrado de datos sensibles.
- Control de errores de red.
- Verificación de relaciones (actividad existente antes de comentar, usuario válido antes de votar).

Consistencia

- La app mantiene consistencia eventual con sincronización mediante tokens de sesión válidos.
- Se espera consistencia fuerte para operaciones críticas (autenticación, inscripción en actividades).

Almacenamiento

- **Local:** Preferencias, tokens de sesión, estado de navegación.
- **Remoto:** Datos estructurados y persistentes como usuarios, actividades, reportes, logros.

- **Servicios externos utilizados:** Firebase (propuesta), REST API personal, Apple Maps.

Este análisis permite comprender cómo se diseña el sistema de información de YaQuedamos para ofrecer una experiencia fluida, segura y eficiente.

Almacenamiento Local y Remoto

YaQuedamos utiliza un enfoque híbrido de almacenamiento, aprovechando tanto los recursos del dispositivo como los servicios en la nube para garantizar eficiencia, velocidad y sincronización entre dispositivos.

Almacenamiento Local

- **UserDefaults:** Se utiliza para guardar configuraciones locales como preferencias de visualización, notificaciones y la pestaña seleccionada.
- **Keychain:** Se emplea para almacenar de forma segura el token JWT y las credenciales del usuario, necesarias para la autenticación persistente.
- **Caché temporal:** Información como el estado de navegación o datos consultados recientemente pueden guardarse localmente para mejorar la experiencia de uso offline parcial.

Almacenamiento Remoto

- **Usuarios:** Perfil, intereses, configuración y logros se almacenan en una base de datos remota (e.g. Firebase, Supabase o backend personalizado).
- **Actividades:** Todas las actividades creadas, junto con detalles como ubicación, participantes y requisitos, se almacenan y consultan desde una API REST.
- **Grupos:** La membresía, reglas y detalles de grupos se manejan desde el backend.
- **Interacciones y Reportes:** Likes, comentarios, votaciones y reportes comunitarios son operaciones que se reflejan directamente en la base de datos remota.

Sincronización

- **Al abrir la app:** Se consulta el backend para sincronizar datos del perfil, nuevas actividades y notificaciones.
- **Eventos críticos:** Crear, editar o eliminar actividades o perfil activa sincronización inmediata.
- **Desconectado:** Si no hay conexión, se muestra el último estado disponible y se reactiva la sincronización al recuperar red.

Uso de Sensores del Dispositivo

YaQuedamos aprovecha varios sensores del dispositivo móvil para enriquecer la experiencia del usuario y mejorar la funcionalidad general de la aplicación. Todos los accesos a sensores están sujetos a permisos explícitos por parte del usuario y cumplen con las políticas de privacidad de iOS.

Sensores Utilizados

1. **Ubicación (GPS):**
    - **Funcionalidad:** Localización del usuario para mostrar actividades cercanas y registrar ubicación al crear eventos.
    - **Permisos requeridos:** NSLocationWhenInUseUsageDescription.
    - **Uso:** Filtrado geográfico en ActivitiesView, asignación de punto en CreateActivityView.
2. **Acceso a galería de fotos:**
    - **Funcionalidad:** Permitir que el usuario suba una foto de perfil o imagen del evento.
    - **Permisos requeridos:** NSPhotoLibraryUsageDescription.
    - **Uso:** ProfileView, EditProfileView, CreateActivityView.
3. **Notificaciones Push:**
    - **Funcionalidad:** Envío de recordatorios de eventos, avisos de nuevas interacciones, confirmaciones de registro.
    - **Permisos requeridos:** Solicitud vía UNUserNotificationCenter.
    - **Uso:** NotificationSettings y eventos importantes.

Consideraciones Técnicas

- Los sensores se activan sólo después de obtener consentimiento.
- Se utilizan mecanismos de fallback para mantener la funcionalidad básica incluso si se deniegan permisos (por ejemplo, selección manual de ubicación).
- Las notificaciones push requieren la integración con servicios como Firebase Cloud Messaging (FCM) o APNs.

Esta integración de sensores permite que YaQuedamos sea una aplicación sensible al contexto, segura y con una experiencia de usuario moderna e intuitiva.

Dispositivos Compatibles y Versiones de Sistema Operativo

YaQuedamos ha sido diseñada para funcionar de forma óptima en dispositivos móviles Apple. A continuación se detallan los dispositivos compatibles y la justificación técnica de los requerimientos mínimos:

Dispositivos Compatibles

- iPhone SE (2da gen) en adelante
- iPhone 11, 12, 13, 14 y 15 series
- Soporte asegurado para pantallas de 4.7", 5.4", 6.1", 6.7"
- Diseño adaptable a orientación vertical (portrait), con elementos responsivos para pantallas compactas

Justificación del Soporte para iOS 17+

- **SwiftUI 3+** (usado en la app) requiere iOS 15 mínimo, pero la app utiliza varios componentes modernos exclusivos de iOS 17 como mejoras en NavigationStack, MapKit, y notificaciones enriquecidas.
- iOS 17 ofrece mejor administración de permisos, integración más estable con Keychain y compatibilidad plena con Firebase SDK moderno.
- Evitar versiones anteriores reduce la carga de mantenimiento y permite aprovechar APIs modernas sin degradar experiencia.

Lenguajes y Herramientas de Desarrollo

YaQuedamos fue desarrollada con herramientas modernas y ampliamente adoptadas en el ecosistema Apple para asegurar escalabilidad, eficiencia y una experiencia de usuario fluida.

Lenguajes Utilizados

- **Swift 5.9+:** Lenguaje principal de programación para la app, por su integración con iOS, rendimiento nativo y seguridad en el manejo de memoria.
- **SwiftUI:** Framework declarativo de interfaz gráfica usado para construir todas las vistas, permitiendo transiciones fluidas, responsividad y modularidad.

Herramientas de Desarrollo

- **Xcode 15+:** Entorno oficial de desarrollo de Apple, utilizado para escritura de código, simulación, pruebas y despliegue.
- **Firebase (opcional/plan futuro):** Para autenticación, almacenamiento remoto, Firestore y notificaciones push.
- **Git y GitHub:** Para control de versiones y trabajo colaborativo.
- **Figma y draw.io :** Para prototipado y diseño de wireframes.

Permisos y Requisitos para Publicación en la App Store

Para que YaQuedamos pueda publicarse en la App Store, se deben cumplir con políticas técnicas y legales impuestas por Apple. A continuación se describen los permisos utilizados y adecuaciones necesarias:

Permisos del Sistema Declarados en Info.plist

- NSLocationWhenInUseUsageDescription: requerido para mostrar actividades cercanas y registrar la ubicación al crear eventos.
- NSPhotoLibraryUsageDescription: necesario para permitir al usuario subir una foto de perfil o imagen de evento.

- NSUserNotificationUsageDescription: para mostrar notificaciones sobre actividades e interacciones importantes.
- NSCameraUsageDescription: reservado para futuras versiones si se habilita carga de imágenes directamente desde la cámara.

Requisitos Técnicos

- La app debe estar firmada con un certificado de desarrollador válido.
- Debe incluir capturas de pantalla, una descripción precisa en App Store Connect y cumplir con la Human Interface Guidelines de Apple.
- El uso de datos debe ser transparente: si se usa Firebase o una API externa, se debe declarar en la sección de privacidad.
- Las solicitudes de permisos deben ser justificadas claramente con mensajes personalizados que expliquen su uso.

Adecuaciones para la Publicación

- Implementar una política de privacidad visible desde la app.
- Integrar un enlace hacia Términos y Condiciones.
- Garantizar que no se generen bloqueos al denegar permisos: la app debe seguir siendo funcional aunque con funciones limitadas.

Cumplir con estas medidas asegura que YaQuedamos pueda ser evaluada y aprobada por Apple sin complicaciones y disponible para los usuarios a través de la App Store.

Equipo de Trabajo y Roles

El desarrollo de YaQuedamos se realizó de manera colaborativa, asignando responsabilidades específicas a cada integrante del equipo para optimizar tiempos y aprovechar sus fortalezas individuales:

Integrantes y Roles:

- **Sánchez Sánchez Santiago** – *Líder de Proyecto / Desarrollador iOS*
  Encargado de toda la programación dentro de Xcode, desarrollo de interfaces con SwiftUI, lógica de negocio y conexión con APIs.
- **Sánchez Manzano Mariana** – *Diseñadora de Arquitectura de Datos y Navegación*
  Responsable de planificar el flujo de navegación de la aplicación y estructurar las bases de datos y la relación entre entidades (usuarios, actividades, grupos, etc.).
- **Laparra Miranda Sandra** – *Encargada de Diseño de Wireframes*
  Desarrolló la propuesta visual de la app mediante wireframes y prototipos, definiendo estructura visual, experiencia de usuario y jerarquía de información.

Plan de Trabajo y Enfoque Ágil

Para el desarrollo de YaQuedamos se utilizó un enfoque ágil basado en sprints, priorizando entregas incrementales de valor mediante historias de usuario organizadas en épicas. El trabajo se dividió entre los tres integrantes del equipo de manera que permitiera avanzar en paralelo en el diseño, desarrollo y arquitectura de datos.

Metodología

- **Duración estimada:** 4 semanas
- **Sprints:** 3 sprints semanales + 1 sprint de cierre y pruebas
- **Herramientas de gestión:** Trello para organización de tareas por rol

Roles asignados

- **Santiago Sánchez Sánchez**
  - Desarrollo de vistas SwiftUI
  - Implementación de lógica de navegación y validaciones
  - Integración con servicios externos y sensores
- **Sánchez Manzano Mariana**
  - Diseño y modelado de base de datos y entidades
  - Definición de rutas y estructura de la API REST
  - Documentación técnica de operaciones CRUD
- **Laparra Miranda Sandra**
  - Diseño visual en Figma
  - Documentación de pantallas y funcionalidades
  - Preparación de los recursos gráficos y prototipo de presentación

Cronograma de actividades (Gantt simplificado)

Semana	Santiago (Dev)	Mariana (Data/API)	Sandra (UX/UI)
1	Setup Xcode + Splash/LoginView	Modelo datos + wireframe base	Diseño wireframes y estilo
2	ActivitiesView + CreateView	Diseño base datos + relaciones	Análisis de flujos y prototipos
3	Integración lógica + navegación	API endpoints + pruebas dummy	Documentación de pantallas

4	Pruebas finales + ajustes	Backup, pruebas de integridad	Ensamble visual y presentación

---

Análisis de Seguridad de la Información

La seguridad de los datos personales y de las actividades dentro de YaQuedamos es una prioridad clave del diseño del sistema. Se adoptan prácticas estándar de la industria para proteger la integridad, confidencialidad y disponibilidad de la información.

Datos que requieren protección:

- **Credenciales del usuario:** contraseña (hash), token JWT
- **Información de perfil:** nombre, correo electrónico, intereses
- **Datos de actividad:** ubicación, participantes, descripciones sensibles

Estrategias de Seguridad Aplicadas:

- **Autenticación segura:** Uso de JWT (JSON Web Token) para validar sesiones activas.
- **Hash de contraseñas:** Implementación de bcrypt o algoritmo equivalente para almacenar contraseñas sin reversibilidad.
- **Almacenamiento seguro en dispositivo:**
    - Token y credenciales: Keychain
    - Preferencias: UserDefaults con acceso restringido
- **Validaciones de entrada:** En formularios se verifican datos requeridos, longitud mínima y formato adecuado.
- **Cifrado de comunicaciones:** HTTPS obligatorio para todas las peticiones API.
- **Control de acceso:** Los endpoints del backend requieren token activo y verificado.
- **Notificaciones privadas:** Solo se muestran si el usuario está autenticado y los permisos han sido otorgados.

Consideraciones adicionales:

- Los datos de ubicación y participación en eventos no se comparten con terceros.
- La app no almacena imágenes sensibles ni información bancaria.
- Se incluirá una política de privacidad visible desde el inicio de sesión y en App Store.

Estas prácticas aseguran que YaQuedamos cumpla con estándares modernos de seguridad y privacidad, promoviendo la confianza de los usuarios.

Estimaciones de Tiempo y Costos

Para proyectar el desarrollo de YaQuedamos, se estimaron los recursos y el tiempo necesario considerando un equipo de 3 personas trabajando en paralelo, utilizando herramientas de código abierto o gratuitas.

Estimación de Tiempo por Fase

Fase	Actividad Principal	Días estimados
Planificación	Análisis de requerimientos y diseño general	2 días
Diseño UI/UX	Wireframes, Figma y flujo de pantallas	3 días
Desarrollo Back + Datos	Modelado de datos y lógica API (sin backend real)	4 días
Desarrollo Frontend iOS	SwiftUI, navegación, formularios, conexión dummy	5 días
Integración y pruebas	Validaciones, revisión UX, solución de errores	3 días
Presentación y documentación	Ensamble del reporte, maqueta, presentación final	3 días

Total estimado: 20 días hábiles

Estimación de Costos (simulado como si fuera un proyecto externo)

Los costos fueron calculados con base en tarifas promedio de la industria mexicana para servicios de desarrollo y diseño freelance, así como licencias y recursos que serían necesarios para lanzar esta app al mercado de manera profesional.

Detalles:

- **Trabajo de desarrollo (3 personas × 20 días):**
    - Estimación basada en una tarifa estándar de $1,000 MXN por día por persona.
    - 3 personas × 20 días × $1,000 = **$60,000 MXN**.
- **Diseño UI/UX:**

    - Estimado en $8,000 MXN por el diseño completo de wireframes, flujo de pantallas y prototipos interactivos.
    - Valor tomado con base en tarifas de diseñadores UX junior a medio tiempo por proyecto.

- **Infraestructura simulada (Firebase):**

    - Considerado sin costo porque se usaría el plan gratuito de Firebase (hasta cierto número de usuarios activos, GB de almacenamiento y peticiones).

- **Licencia Apple Developer:**

    - Costo real de la licencia anual para publicar en App Store.
    - Apple cobra $99 USD por año (~$2,300 MXN al tipo de cambio estimado de $23 MXN/USD).

- **Otros costos (dominios, hosting opcional):**

    - Reservado para pagos futuros relacionados con infraestructura como dominios web o backend personalizado.
    - Monto estimado: **$2,000 MXN** como base de referencia para un primer año.

Estos valores reflejan lo que un equipo independiente requeriría para llevar YaQuedamos a un entorno de producción básico en el ecosistema Apple.

Concepto	**Costo estimado (MXN)**
Trabajo de desarrollo (3 pers. × 20 días)	$60,000
Diseño UI/UX (equivalente externo)	$8,000
Infraestructura simulada (Firebase básico)	$0 (plan gratuito)
Licencia Apple Developer (anual)	$2,300
Otros (hosting futuro, dominios, etc.)	$2,000

Total aproximado: $72,300 MXN (si se llevara al mercado)

Esta proyección ayuda a visualizar el valor de una app de este tipo en un entorno profesional, aunque para fines académicos, el desarrollo fue realizado sin costo económico directo.

Referencias

1. Apple Inc. (2024). *Apple Developer Program*. Disponible en: https://developer.apple.com/programs/
2. Firebase Documentation. Google. *Pricing Plans & Free Tier*. https://firebase.google.com/pricing
3. Swift Documentation – Apple Developer. https://developer.apple.com/documentation/swift
4. SwiftUI Documentation – Apple Developer. https://developer.apple.com/documentation/swiftui
5. Human Interface Guidelines – Apple. https://developer.apple.com/design/human-interface-guidelines/
6. Supabase Docs (2024). *Open Source Firebase Alternative*. https://supabase.com/docs
7. Mercado Libre Freelance México (2024). *Promedios salariales para diseño y desarrollo de apps móviles*.