DOCUMENTO DE PLANIFICACIÓN

Índice

1. Resumen del Proyecto

Nombre aplicación: NowWeather

Ofrece al usuario información meteorológica en tiempo real basada en su ubicación actual. La app consume la API pública de OpenWeatherMap para mostrar datos actualizados, incluyendo temperatura y condiciones climáticas. El sistema gestiona diferentes estados de la interfaz (carga, error y datos disponibles) y ofrece mensajes claros ante cualquier error que se produzca.

2. Objetivo General

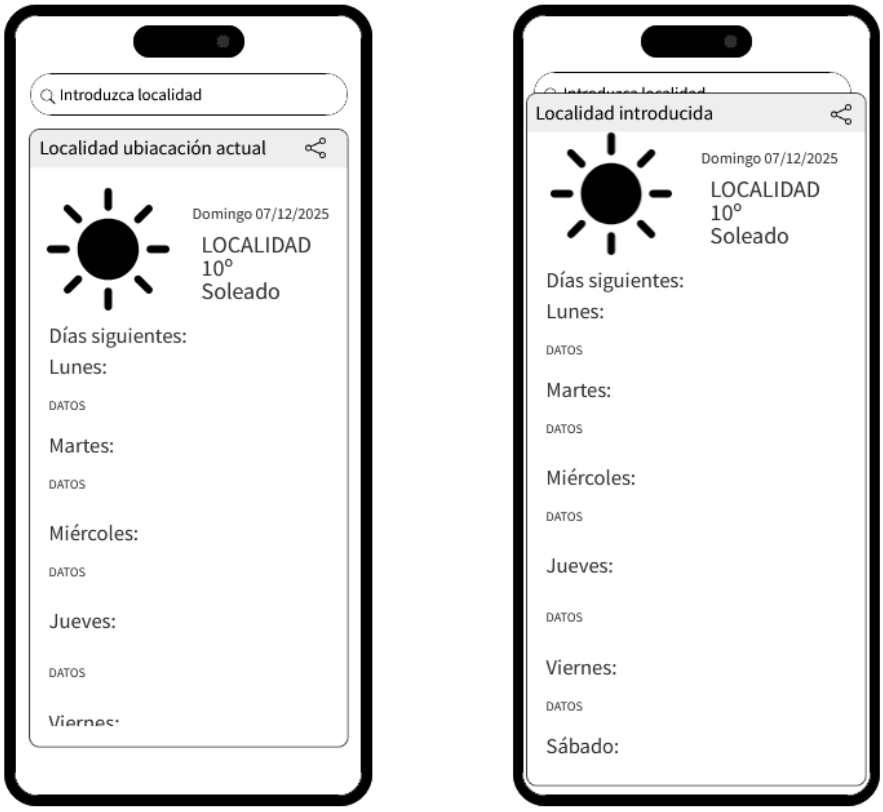
El objetivo principal de NowWeather es ofrecer al usuario una experiencia rápida, clara y útil, facilitando el acceso inmediato a información meteorológica fiable y actualizada.

5. MVP (Producto Mínimo Viable)

Inicialmente, la aplicación será funcional con las siguientes características mínimas:

- La pantalla principal indicará el tiempo actual.
- Dispondrá de un apartado para buscar el tiempo de una ubicación concreta.
- Permitirá compartir el tiempo seleccionado con otro usuario
- Manejo de estados y errores:
  - Mostrará estados de carga mientras consulta la API
  - Mostrará estados de error concretos para cada caso:
    - Sin conexión a internet
    - Error devuelto por la API
    - La ciudad no se encuentra
    - No se tiene permisos de localización

6. Boceto / Croquis Inicial



7. Notas de Diseño y Usabilidad

Estilo visual: Diseño simple, limpio y minimalista. Paleta de colores suaves (azules y verdes claros) para transmitir calma y organización. Tipografía legible, sin adornos a excepción de iconos que ayuden a clarificar la acción o función.

Consistencia: Para mantener una coherencia visual en todos los componentes, la interfaz se estructurará mediante **UITableView**, utilizando celdas personalizadas que encapsulan cada sección de contenido. Este enfoque permite organizar la información de forma modular, facilita la reutilización de elementos y garantiza un comportamiento uniforme del diseño en diferentes tamaños de pantalla. Además, el uso de celdas simplifica la gestión del scroll y mejora la adaptabilidad del layout.

Usuarios y acceso: La aplicación está pensada para un uso general por cualquier persona que necesite consultar el tiempo de forma rápida y fiable con acceso directo. No requiere login ni autenticación.

Flujo de uso: El flujo será directo y rápido: al abrir la app, se mostrará el estado del tiempo actual. El usuario podrá refrescar los datos fácilmente y acceder a funcionalidades adicionales (como compartir el tiempo o buscar otra ciudad) mediante accesos claros y minimalistas. Se priorizará la reducción de pasos innecesarios y el acceso inmediato a la información principal.

8. Ideas técnicas preliminares

Lenguajes y tecnologías:

- **Backend**: La aplicación consumirá la API pública de **OpenWeatherMap**.
- **Frontend móvil**: Aplicación iOS en Swift como framework UIKIT.

Arquitectura:

- Se aplicará la arquitectura MVVM y Coordinators, garantizando la separación clara de responsabilidades y escalabilidad.

- La interfaz se estructurará mediante **UITableView**, donde cada sección estará representada por una celda personalizada.

- Se utilizará un Drawer Protocol para que cada celda sea responsable de configurarse y dibujarse a sí misma, centralizando la lógica de presentación dentro de cada componente.

Infraestructura y despliegue:

- La app se ejecutará de forma nativa en dispositivos iOS

Sincronización:

- No se requiere sincronización multiusuario. La información se obtiene en tiempo real desde la API y se actualiza bajo demanda o según interacción del usuario.

Base de datos y persistencia:

- No es necesaria una base de datos. De forma opcional, se podrán cachear temporalmente las últimas condiciones meteorológicas para mejorar la experiencia en ausencia de conexión.

9. Riesgos y retos previstos

• **Disponibilidad de la API externa:**
La aplicación depende completamente de OpenWeatherMap. Fallos en la API, límites de uso o cambios en el servicio pueden afectar a la disponibilidad de los datos.

• **Gestión de permisos de localización:**
Es necesario manejar correctamente los estados de permisos (concedido, denegado, restringido). Un manejo incorrecto puede impedir mostrar el clima basado en la ubicación del usuario.

• **Manejo de estados offline/online:**
El usuario puede abrir la app sin conexión. Es necesario mostrar mensajes

claros y, opcionalmente, los últimos datos almacenados en caché para evitar una pantalla vacía.

• **Manejo de errores y experiencia de usuario:**
Se deben gestionar errores comunes: falta de conexión, ciudad no encontrada, errores de red, datos incompletos o respuestas no válidas. La app debe presentar mensajes claros y mantener una experiencia fluida.

• **Rendimiento y optimización:**
Las llamadas a la API deben ser eficientes y la interfaz debe actualizarse sin bloqueos. El sistema debe evitar sobrecargar la red con solicitudes innecesarias.

• **Precisión y actualización de datos:**
Garantizar que la información mostrada sea reciente y coherente, gestionando correctamente refrescos manuales o automáticos.

• **Testing y QA:**
Es necesario validar estados de la vista (loading, error, éxito), la gestión de permisos, el comportamiento ante errores de API y el correcto renderizado de los datos en diferentes tamaños de pantalla.

• **Mantenimiento y evolución:**
A medida que la app crezca, será necesario mantener la arquitectura, actualizar dependencias, realizar optimizaciones y adaptarse a nuevas versiones de iOS sin comprometer la experiencia del usuario.

10. Plan de trabajo tentativo

Fecha de inicio de proyecto: 9/12/2025

Fecha de entrega de MVP: 12/12/2025