**Designación de roles :**

Product Owner: Juan Dario Sinatra

Scrum Master: Luca Valentino Gómez Bibiloni

Team Members: Carla Aldana Bustos Betetto, Emanuel Emiliano Ortiz Rabino y Matias Elian Martinez Guerra

**Historias de Usuario y Criterios de Aceptación :**

1. Consulta básica del clima por ciudad

* **Como** usuario, **quiero** ingresar el nombre de una ciudad **para** consultar el clima actual

**Criterios de aceptación**:

* El usuario puede introducir el nombre de una ciudad en un campo de búsqueda.
* El sistema debe mostrar información básica del clima (temperatura, estado general - soleado, nublado, lluvioso).
* Si la ciudad no es válida o no se encuentra, el sistema muestra un mensaje de error adecuado.

1. Información meteorológica detallada

* **Como** usuario, **quiero** ver detalles adicionales sobre el clima **para** tener información completa del estado meteorológico actual.

**Criterios de aceptación**:

* + 1. El sistema muestra detalles como:
       1. Temperatura mínima y máxima
       2. Humedad
       3. Sensación térmica
       4. Probabilidad de precipitación
       5. Estado del cielo (nublado, despejado, etc.)
    2. Los datos se deben mostrar de forma clara y ordenada.

1. Selección de unidades de medida

* **Como** usuario, **quiero** tener unidades de medida y seleccionarlas **para** seguir la convención actual de mí país.
* Criterios de aceptación
  + - El usuario puede alternar entre grados Celsius y Farenheit para mostrar la temperatura
    - El sistema deberá guardar la preferencia elegida del usuario para futuras consultas

1. Consulta del pronóstico para varios días

* **Como** usuario, **quiero** ver el pronóstico meteorológico de los próximos días **para** planificar futuras actividades

Criterios de aplicación

* + - El sistema muestra el pronostico meteorológico para al menos los proximos 5 dias
    - Cada día muestra la información básica que requerirá el usuario (Temperatura, estado general del clima)
    - El usuario puede consultar detalles adicionales para cada día (Temp. Mínima y Máxima, Probabilidad de lluvia, Humedad, etc.)

1. Historial de ciudades consultadas

* **Como** usuario, **quiero** que la app recuerde todas las ciudades que he buscado **para** averiguar su clima cuando lo necesite sin volver a buscarlas manualmente

Criterios de aplicación:

* + - La aplicación muestra una lista de las ciudades previamente consultadas
    - El usuario puede seleccionar una ciudad de la lista para ver su clima actual
    - Opción para limpiar el historial de ciudades si el usuario así lo requiere

1. Actualizar clima en tiempo real

* **Como** usuario, **quiero** se actualice automáticamente **para** estar siempre informado a cada momento del clima de la ciudad donde esté

Criterios de aplicación:

* + - El clima de la ciudad seleccionada se actualiza cada cierto periodo de tiempo (ejemplo, cada 10/15 minutos)
    - El usuario puede actualizar manualmente si así lo requiere en el momento.

1. Interfaz amigable y visualización del clima con iconos

* **Como** usuario, **quiero** que la app tenga iconos y graficos que representen visualmente las condiciones climáticas actuales **para** entender el estado del clima más sencillo.

Criterios de aplicación:

* + - El sistema utiliza iconos para representar condiciones meteorológicas (sol, nubes, lluvia, nieve, etc.)
    - El diseño debe ser limpio y fácil de navegar

Product Backlog

* + - 1. **Funcionalidad básica de consulta de clima por ciudad**
* Historia de usuario: **Como** usuario, quiero ingresar el nombre de una ciudad para consultar el clima actual.
  + Tareas:
    - Crear interfaz de búsqueda de ciudad.
    - Implementar integración con la API externa de clima.
    - Mostrar datos meteorológicos básicos (temperatura, estado del clima).
    - Validar si la ciudad ingresada es válida o no.
    - Mostrar mensajes de error si la ciudad no existe o no es válida.
  + **Prioridad**: Alta
    - 1. **Mostrar información meteorológica detallada**
* Historia de usuario: **Como** usuario, **quiero** ver detalles adicionales sobre el clima **para** tener información completa del estado meteorológico actual.
  + Tareas:
    - Añadir información de temperatura mínima y máxima.
    - Mostrar datos de humedad.
    - Incluir sensación térmica y estado del cielo.
    - Diseñar interfaz para mostrar estos detalles de manera clara y organizada.
  + **Prioridad**: Alta
    - 1. **Selección de unidades de medida**
* Historia de usuario: **Como** usuario, **quiero** poder elegir entre diferentes unidades de medida (Celsius/Fahrenheit) **para** visualizar la información meteorológica según mis preferencias.
  + Tareas:
    - Crear opciones para alternar entre Celsius/Fahrenheit.
    - Crear opciones para cambiar entre kilómetros/h y millas/h.
    - Guardar las preferencias del usuario localmente para que se mantengan en futuras consultas.
  + **Prioridad**: Media
    - 1. **Pronóstico de varios días**
* Historia de usuario: **Como** usuario, **quiero** ver el pronóstico meteorológico de los próximos días **para** planificar mejor mis actividades.
  + Tareas:
    - Implementar la funcionalidad para obtener el pronóstico extendido (al menos 5 días).
    - Mostrar información básica del clima para cada día (temperatura, estado del clima).
    - Habilitar la opción de ver detalles adicionales de cada día.
    - Diseño de interfaz para mostrar el pronóstico de varios días.
  + **Prioridad**: Media
    - 1. **Historial de ciudades consultadas**
* Historia de usuario: **Como** usuario, **quiero** que la app recuerde las ciudades que he buscado previamente **para** acceder rápidamente a su clima sin tener que escribirlas de nuevo.
  + Tareas:
    - Crear una base de datos local para almacenar el historial de búsquedas de ciudades.
    - Mostrar las ciudades consultadas previamente en una lista.
    - Implementar la opción de seleccionar una ciudad del historial para ver su clima.
    - Añadir opción para borrar el historial.
  + **Prioridad**: Baja
    - 1. **Actualizar clima en tiempo real**
* Historia de usuario: **Como** usuario, **quiero** que la información meteorológica se actualice automáticamente **para** obtener el estado más reciente del clima en la ciudad seleccionada.
  + Tareas:
    - Programar la actualización automática del clima cada 15 minutos.
    - Implementar opción de actualización manual.
    - Notificar al usuario si hay problemas al intentar actualizar los datos.
  + **Prioridad**: Media
    - 1. **Interfaz con iconos y gráficos del clima**
* Historia de usuario: **Como** usuario, **quiero** que la aplicación tenga iconos y gráficos que representen visualmente las condiciones climáticas **para** entender el estado del clima de un vistazo.
  + Tareas:
    - Diseñar e integrar iconos representativos (sol, nubes, lluvia, etc.).
    - Implementar gráficos para mostrar tendencias (por ejemplo, gráfico de temperatura).
    - Asegurar que los iconos se ajusten al pronóstico del clima consultado.
  + **Prioridad**: Media

Sprint Backlog

**Sprint 1: Funcionalidad básica de consulta de clima y manejo de errores**

El objetivo del primer sprint es desarrollar la funcionalidad central de la app: la capacidad de consultar el clima de una ciudad, recibir datos desde una API externa y mostrar la información básica del clima.

* **Objetivo del sprint**: Tener una versión inicial de la aplicación que permita buscar el clima de una ciudad y visualizar la información básica.
* **Historias de usuario incluidas**:
  1. **Consulta básica del clima por ciudad**
     + Crear interfaz de búsqueda de ciudad.
     + Implementar integración con la API externa de clima.
     + Mostrar datos meteorológicos básicos (temperatura y estado del clima).
     + Validar si la ciudad ingresada es válida o no.
     + Mostrar mensajes de error si la ciudad no existe o no es válida.
* **Tareas**:
  1. Configurar la arquitectura de la app.
  2. Desarrollar la interfaz de búsqueda de ciudad (input de texto).
  3. Implementar conexión a la API externa para obtener datos meteorológicos.
  4. Manejar errores y respuestas de la API para mostrar mensajes adecuados.
  5. Pruebas básicas de funcionalidad (manuales o automatizadas).
* **Resultados esperados**:
  1. El usuario puede buscar una ciudad y ver el clima básico.
  2. Manejo adecuado de errores (ciudad no encontrada o error de conexión).

**Sprint 2: Información detallada y detección automática de ubicación**

Este sprint se enfocará en ampliar la funcionalidad básica añadiendo más detalles sobre el clima y la posibilidad de detectar automáticamente la ubicación del usuario, ofreciendo una experiencia más completa.

* **Objetivo del sprint**: Proporcionar información meteorológica detallada.
* **Historias de usuario incluidas**:
  1. **Información meteorológica detallada**:
     + Añadir detalles: temperatura mínima y máxima, humedad, sensación térmica.
     + Mostrar los datos adicionales de forma clara.
* **Tareas**:
  1. Ampliar la interfaz para mostrar detalles adicionales (temperatura mínima y máxima, humedad, etc.).
  2. Conectar con las propiedades adicionales de la API meteorológica para obtener esos datos.
  3. Pruebas y correcciones de errores.
* **Resultados esperados**:
  1. El usuario puede ver el clima detallado de una ciudad.

**Sprint 3: Selección de unidades de medida, pronóstico extendido y actualización en tiempo real**

En este sprint se busca agregar características avanzadas, como la selección de unidades de medida, la visualización del pronóstico del clima para varios días y la actualización automática de la información meteorológica.

* **Objetivo del sprint**: Permitir al usuario elegir unidades de medida, ver el pronóstico extendido y actualizar el clima en tiempo real.
* **Historias de usuario incluidas**:
  1. **Selección de unidades de medida**:
     + Crear opciones para alternar entre Celsius/Fahrenheit.
     + Guardar las preferencias del usuario localmente.
  2. **Pronóstico de varios días**:
     + Implementar la funcionalidad para obtener y mostrar el pronóstico para al menos 5 días.
     + Mostrar información básica para cada día (estado del clima, temperatura).
  3. **Actualización automática del clima**:
     + Implementar actualización automática del clima cada 15 minutos.
     + Crear opción de actualización manual.
* **Tareas**:
  1. Desarrollar opciones de configuración para cambiar entre Celsius y Fahrenheit.
  2. Implementar la lógica para obtener el pronóstico del clima para los próximos días desde la API.
  3. Diseñar la interfaz para mostrar el pronóstico diario en un formato fácil de leer.
  4. Implementar la actualización automática cada cierto intervalo de tiempo.
  5. Añadir la opción manual de actualización.
  6. Guardar las preferencias del usuario (unidades de medida).
  7. Pruebas de integración.
* **Resultados esperados**:
  1. El usuario puede cambiar entre diferentes unidades de medida.
  2. El usuario puede ver el pronóstico para los próximos días.
  3. El clima se actualiza automáticamente o manualmente cuando el usuario lo desee.

**Backlog del Product Owner :**

