Consulta y Visualización de Datos Meteorológicos con Python

Proyecto Grado II

Roger Vera
rogerjverag@gmail.com
Área de Ingeniería en Sistemas
Programa de Ingeniería en Informática

San Juan de los Morros - Febrero 2025

APROBACIÓN DEL TUTOR

En mi carácter de Tutor del Proyecto de Grado presentado por el ciudadano Roger Vera, para Optar al Título de Ingeniero en Informática, titulado: **Consulta y Visualización de datos meteorológicos con Python** confirmo que he revisado dicho proyecto en cada una de las partes, guiando y asesorando al autor de este; por lo que considero, reúne los requisitos y méritos suficientes para ser sometido a la presentación pública y evaluación por parte del jurado examinador que se designe en nuestra distinguida Universidad.

En la ciudad de San Juan de los Morros, a los 26 días del mes de junio del 2024.

Gabriel Díaz

C.I. N° 19.725.444

Índice

Índice de tablas

Índice de figuras

República Bolivariana de Venezuela
Ministerio del Poder Popular para la Educación Superior
Universidad Nacional Experimental Rómulo Gallegos de los Llanos Centrales
San Juan de los Morros. Estado Guárico
Área de Ingeniería en Sistemas
Unidad Curricular: Proyecto Grado II

Consulta y Visualización de Datos Meteorológicos Con Python

Autor: Roger Vera.
Tutor Académico: Gabriel Díaz.
Año: 2024.

3. RESUMEN

El proyecto propone desarrollar una herramienta de consulta y visualización de datos meteorológicos para la Universidad Nacional Experimental Rómulo Gallegos (UNERG), en respuesta a la necesidad de acceso a información climática precisa y oportuna en Venezuela, especialmente en áreas con alta diversidad geográfica. Basado en Python, el sistema integrará datos de diversas fuentes, almacenándolos en MongoDB y visualizándolos a través de una interfaz intuitiva desarrollada con el framework Flet. La herramienta promete ser accesible y fácil de usar, facilitando la toma de decisiones informadas sobre actividades diarias y preparación para eventos climáticos. La metodología SCRUM guiará el desarrollo, permitiendo flexibilidad y adaptabilidad. Este proyecto no solo beneficiará a estudiantes y residentes locales, sino que también contribuirá a la conciencia ambiental y a la adaptabilidad a los cambios climáticos, fortaleciendo la responsabilidad colectiva hacia el medio ambiente.

SUMMARY

The project proposes to develop a meteorological data visualization and query tool for the Universidad Nacional Experimental Rómulo Gallegos (UNERG), in response to the need for access to accurate and timely climate information in Venezuela, especially in areas with high geographic diversity. Based on Python, the system will integrate data from various sources, storing them in MongoDB and visualizing them through an intuitive interface developed with the Flet framework. The tool promises to be accessible and easy to use, facilitating informed decision making about daily activities and preparation for climatic events. The SCRUM methodology will guide the development, allowing flexibility and adaptability. This project will not only benefit students and local residents, but will also contribute to environmental awareness and adaptability to climate change, strengthening collective responsibility towards the environment.

4. Diagnóstico Situacional:

4.1 Descripción del contexto de la situación problemática planteada:

El acceso a información meteorológica precisa y oportuna se ha convertido en una necesidad global debido a los cambios climáticos y las variaciones meteorológicas cada vez más pronunciadas. Entendemos por meteorología como la ciencia interdisciplinaria que estudia la física de la atmósfera. Su nombre proviene de las voces griegas metéöron (“en lo alto del cielo”) y lógos (“conocimiento”). Su objetivo es entender y predecir los fenómenos atmosféricos, es decir, hacerse una idea respecto al tiempo atmosférico.[1] Herramientas como Weather Underground, AccuWeather, y OpenWeatherMap ofrecen servicios de pronóstico del tiempo y datos históricos, aunque su utilidad y accesibilidad varían considerablemente. Estas plataformas, aunque valiosas, a menudo están dirigidas a audiencias internacionales y pueden requerir conocimientos técnicos para su uso completo, limitando así su accesibilidad para el público general. Las aplicaciones móviles y sitios web especializados han facilitado el acceso a la información meteorológica, permitiendo que más personas puedan estar informadas sobre las condiciones climáticas actuales y futuras. Sin embargo, la precisión y la actualización de los datos siguen siendo un desafío constante para los proveedores de servicios meteorológicos.

En países como Venezuela, donde la diversidad geográfica y las condiciones climáticas son características predominantes, la necesidad de acceder a datos meteorológicos precisos se intensifica. Las herramientas disponibles a menudo enfrentan desafíos de infraestructura, conectividad y costos, lo que restringe aún más el acceso a la información climática para la población general. En este contexto, la Universidad Nacional Experimental Rómulo Gallegos (UNERG), ubicada en San Juan de los Morros, Estado Guárico, Venezuela, tiene la oportunidad única de abordar esta brecha mediante el desarrollo de una herramienta de consulta y visualización de datos meteorológicos accesible y fácil de usar. Esta iniciativa no solo beneficiaría a la comunidad académica sino proporciona información crucial para la planificación y la toma de decisiones en actividades organizadas por la UNERG.

La UNERG puede aprovechar su ubicación estratégica para ofrecer una herramienta invaluable para estudiantes, trabajadores y residentes locales. La falta de una plataforma local que integre y presente datos meteorológicos de manera accesible y comprensible representa una oportunidad significativa para mejorar la adaptabilidad y seguridad de la comunidad ante cambios climáticos imprevistos. Dado que la herramienta se centrará en la consulta de datos de otras fuentes, su diseño y desarrollo deben priorizar la facilidad de uso y la comprensión de la información por parte del usuario final. Además, es crucial que la herramienta sea capaz de actualizarse en tiempo real para proporcionar información meteorológica precisa y oportuna, lo que permitirá a los usuarios tomar decisiones informadas y prepararse adecuadamente para condiciones climáticas adversas.

Para abordar la necesidad de acceso a información meteorológica precisa y oportuna, se propone el desarrollo de una herramienta de consulta y visualización de datos meteorológicos. Esta herramienta se centrará en la extracción, almacenamiento y visualización de datos meteorológicos de diferentes fuentes mediante una aplicación de escritorio sencilla. Utilizando tecnologías como Python, MongoDB y el

[1] Editorial Etecé, 2.022, https://concepto.de/meteorologia/

framework Flet, se garantizará una solución robusta, modular y accesible. La herramienta permitirá la fácil comprensión y visualización de la información climática a través de tablas y gráficos, facilitando la toma de decisiones informadas y mejorando la adaptabilidad y seguridad de la comunidad ante cambios climáticos imprevistos. Además, la adopción de la metodología SCRUM permitirá una gestión eficiente del proyecto, asegurando la flexibilidad y la capacidad de incorporar nuevas funcionalidades o ajustes según sea necesario.

4.2 Justificación del proyecto:

Justificación Social

En el ámbito social, la herramienta va más allá de ser un simple instrumento de consulta; se transforma en un catalizador para la acción comunitaria y el cambio ambiental. Al proporcionar acceso a información climática precisa y oportuna, se empodera a la comunidad para tomar decisiones informadas que pueden beneficiar tanto a individuos como a la sociedad en su conjunto. Además, al fomentar la discusión y la educación sobre el cambio climático y la sostenibilidad, la herramienta contribuye a la formación de una cultura de respeto y cuidado del medio ambiente, promoviendo así una relación más armoniosa y responsable con nuestro planeta.

Justificación Tecnológica

El uso de Python, junto con bibliotecas específicas, no solo se justifica por su versatilidad y eficiencia en el manejo de datos, sino también por su capacidad para integrarse con otras tecnologías modernas. Por ejemplo, la conexión con MongoDB no solo garantiza un almacenamiento seguro y escalable, sino que también permite la realización de consultas complejas y la sincronización de datos en tiempo real. Además, el empleo del framework Flet no solo facilita el despliegue y mantenimiento de la aplicación, sino que también asegura su portabilidad y compatibilidad entre diferentes entornos, lo cual es crucial para una herramienta destinada a ser utilizada por una amplia audiencia.

Justificación Teórica

Este trabajo también tiene como propósito ser referencia para futuros trabajos que se planteen con un objetivo similar, ya sea en el uso de metodologías, tecnologías o cualquiera de las herramientas que se han descrito y serán descritas en las siguientes páginas.

4.3 Objetivos del proyecto:

Objetivo General

Implementar un sistema para consultar y visualizar datos meteorológicos de diferentes fuentes por medio de una aplicación de escritorio sencilla desarrollada en Python.

Objetivos Específicos

- Analizar API’s públicas que proveen consulta y visualización de datos meteorológicos.
- Diagnosticar los requisitos en términos de hardware y software para el desarrollo de la herramienta tecnológica, asegurando su compatibilidad y eficiencia operativa.
- Diseñar las principales funcionalidades del software de forma robusta y modular que facilite la extracción de datos de diversas fuentes y su almacenamiento seguro.
- Desarrollar las funciones necesarias para la visualización de la información climática a través de tablas y gráficos, usando una interfaz de usuario intuitiva y amigable con el usuario.

4.4 Procesos que se van a automatizar:

La herramienta propuesta automatizará varios procesos clave relacionados con la obtención, gestión y presentación de datos meteorológicos. En primer lugar, automatizará la extracción de datos de diferentes fuentes meteorológicas, lo que incluye la consulta a API's públicas. Posteriormente, estos datos serán almacenados de manera segura y eficiente en una base de datos MongoDB, garantizando su integridad y disponibilidad. La herramienta también automatizará el proceso de visualización de los datos, permitiendo la generación dinámica de tablas y gráficos que faciliten la comprensión y análisis de la información climática. Además, la solución utilizará el framework Flet, que asegura la portabilidad y consistencia de la interfaz de usuario, facilitando así el despliegue y mantenimiento de la aplicación en diferentes entornos. Con estos procesos automatizados, la herramienta ofrecerá una solución integral y accesible para la consulta y visualización de datos meteorológicos, beneficiando tanto a estudiantes como a residentes locales.

5. Determinación, Instalación y Configuración de las Herramientas de Desarrollo:

5.1 Plataforma de Desarrollo:

		Tipo de herramienta	Definición
Plataforma de desarrollo	**Software**	Sistema Operativo Ubuntu 22.04	Distribución GNU/Linux basada en Debian GNU/Linux.
		Visual Studio Code	Es un **editor de código fuente** desarrollado por **Microsoft**, diseñado para ser ligero, rápido y altamente personalizable.
		Python 3.10.12	Lenguaje de programación ampliamente utilizado en desarrollo de software.
		Mongo DB 6.0.20	Es un sistema de base de datos NoSQL, orientado a documentos y de código abierto
		Flet 0.26	Flet es un framework que permite crear aplicaciones web, de escritorio y móviles en Python.
	Hardware	Laptop Asus Vivobook F1500E	Una laptop con veinte gigas de RAM, un procesador intel core cinco de onceava generación con 2.40 GHz, una tarjeta gráfica integrada de 4 gigabytes y un disco duro de quinientos gigabytes.
	Otro	Conexión a internet	El sistema de enlace que permite a los dispositivos acceder a la red de internet y sus servicios

Tabla [1] Roger Vera 2.024

5.2 Arquitectura del sistema de información:

Sabemos que para el correcto desarrollo de un sistema debemos tener en cuenta diversos temas, entre ellos la estructura y organización para el desarrollo, entre ellos la arquitectura de sistemas o arquitectura de software.

La arquitectura de software se refiere a la estructura organizativa y conceptual que subyace en el diseño y desarrollo de sistemas de software. Es el proceso de definir una solución estructural que cumple con todos los requisitos funcionales y no funcionales, como la escalabilidad, el rendimiento y la seguridad. Su importancia radica en su capacidad para proporcionar un marco de trabajo que guía las decisiones de diseño y desarrollo. Aquí, se establece el enfoque, la estructura, y las tecnologías necesarias para construir una aplicación.[2]

[2] Miriam Martínez Canelo,2.024, https://profile.es/blog/que-es-la-arquitectura-de-software/

Para este sistema se pensó en la arquitectura de capas, la arquitectura mas usada debido a su gran versatilidad y con una distribución jerárquica que permite agregar una capa de seguridad extra a los sistemas desarrollados bajo dicha arquitectura. La arquitectura en capas es un patrón de arquitectura software usada en la gran mayoría de sistemas. Se centra en una distribución jerárquica de las roles y responsabilidades proporcionando una separación efectiva de las preocupaciones (cada cual se encarga de lo que le corresponde).[3]

Para esta arquitectura lo estándar es usar 3 capas, una capa de interfaz, una capa que contiene toda la lógica del sistema y una capa de datos donde se tiene la base de datos y su respectivo gestor, esta arquitectura no debe seguir esta estructura de 3 capas de forma obligatoria ya que existen sistemas con hasta 7 capas pero esto siempre dependerá de las necesidades del sistema.

5.3 Selección del entorno del sistema:

Elegir el sistema operativo adecuado es crucial para el éxito de cualquier proyecto de desarrollo de software. En el caso de un proyecto que involucra la creación de una herramienta para consultar y visualizar datos meteorológicos utilizando Python.

1. **Compatibilidad y soporte para Python**: Ubuntu 22.04 viene con Python 3.10 preinstalado, lo que facilita la configuración inicial del entorno de desarrollo. Además, Ubuntu tiene un excelente soporte para versiones anteriores y futuras de Python, lo que garantiza que las librerías y herramientas necesarias para el proyecto funcionarán sin problemas.
2. **Facilidad de instalación y configuración de herramientas**: Ubuntu 22.04 cuenta con un amplio repositorio de paquetes que facilita la instalación de herramientas como Flet, MongoDB, VS Code y Git. La gestión de paquetes a través de APT (Advanced Package Tool) es robusta y eficiente, permitiendo una configuración rápida y sin complicaciones.
3. **Compatibilidad con Flet**: Flet permite a los desarrolladores crear aplicaciones en Python que se ejecutan en múltiples plataformas, incluyendo Ubuntu, sin necesidad de experiencia en frontend. Su arquitectura monolítica simplifica el desarrollo y despliegue, lo que resulta en una experiencia más eficiente y accesible para los usuarios de Ubuntu.
4. **Creación de entornos virtuales:** El uso de entornos virtuales en Ubuntu permite crear espacios aislados para cada proyecto, lo que evita conflictos entre dependencias y facilita la gestión de versiones de bibliotecas. Esto se traduce en una administración más eficiente de los paquetes, ya que cada entorno puede tener sus propias dependencias sin afectar al sistema global. Además, Ubuntu es compatible con diversas herramientas de desarrollo, como venv y virtualenv, que simplifican la creación y gestión de estos entornos, permitiendo a los desarrolladores comenzar a trabajar en sus proyectos de manera rápida y sin complicaciones.

[3] Mauricio Costanzo, 2019,
https://platzi.com/tutoriales/1248-pro-arquitectura/5439-patron-arquitectonico-de-capas-layers/

5.4 Metodología para el desarrollo:

Se adoptará la metodología SCRUM, una estrategia ágil de gestión de proyectos que permite una gestión eficiente de los tiempos de entrega y la flexibilidad para incorporar nuevas funcionalidades o ajustes según sea necesario. Dado que el equipo de desarrollo está compuesto únicamente por una sola persona, se enfocará en la organización personal y la disciplina para seguir los ciclos de sprint de SCRUM, asegurando un flujo de trabajo constante y productivo.

Scrum es un marco de trabajo ágil a través del cual las personas pueden abordar problemas complejos adaptativos a la vez que se entregan productos de forma eficiente y creativa con el máximo valor. Así, Scrum es una metodología que ayuda a los equipos a colaborar y realizar un trabajo de alto impacto.[4]

Se utilizará un tablero digital para visualizar el progreso del proyecto, siguiendo la estructura típica de SCRUM con columnas para "Backlog", "Sprint", "In Progress" y "Done". Las tareas se moverán entre estas columnas a medida que se avanza, permitiendo una visión clara del estado actual del proyecto y facilitando la planificación y revisión de sprints.

Las fases de esta metodología son:

- **Elaboración del Product Backlog:**

La primera fase de la metodología Scrum es la elaboración del **Product Backlog**. En esta etapa, se desarrolla una lista exhaustiva de todo lo necesario para el proyecto, incluyendo características, mejoras y correcciones deseadas. Es fundamental crear y mantener este backlog, asignando prioridades a las tareas basadas en su relevancia y urgencia. La claridad en lo que se necesita y la planificación eficiente de los sprints son resultados clave de esta fase inicial.

- **Selección de Tareas para el Sprint Backlog:**

Tras la elaboración del Product Backlog, la siguiente fase implica seleccionar las tareas para el **Sprint Backlog**. Este backlog actúa como el plan de acción para el sprint actual, especificando exactamente qué tareas se trabajarán y cuándo se esperan completadas. La selección cuidadosa de las tareas permite una gestión efectiva del tiempo y recursos durante el sprint.

- **Realización del Sprint:**

Durante el **Sprint**, se enfoca en trabajar las tareas definidas en el Sprint Backlog. Es crucial mantener un registro del progreso y asegurarse de que se están realizando avances significativos hacia el objetivo del sprint. Esta fase requiere disciplina y compromiso por parte del equipo para cumplir con los objetivos establecidos.

[4] Julia Martins, 2.024, https://asana.com/es/resources/what-is-scrum

- **Evaluación en el Sprint Review:**

Al final del Sprint, se realiza el Sprint Review. Esta es una oportunidad para evaluar lo que se ha logrado durante el sprint, comparando el trabajo completado con los objetivos establecidos. Además, es un momento para realizar una autoevaluación del rendimiento individual y considerar posibles áreas de mejora.

- **Reflexión en el Sprint Retrospective:**

Finalmente, después del Sprint Review, se lleva a cabo el Sprint Retrospective. Esta fase ofrece una oportunidad para reflexionar sobre el sprint pasado, identificando tanto aspectos positivos como áreas de mejora. Se considera el proceso de trabajo, los desafíos enfrentados y cómo podrían modificarse las estrategias para futuros sprints.

6. Desarrollo del Sistema de Información:

6.1.1 Descripción:

El objetivo de este sistema es proporcionar información meteorológica de varias fuentes a los usuarios, mostrando tablas e información relevante sobre el clima en tiempo real.

Desarrollado en Python y con una base de datos MongoDB que garantiza la seguridad y escalabilidad de la información recopilada, además de las consultas a las diferentes API's como OpenWeatherMap, hacen de este sistema una herramienta de gran utilidad para la vida cotidiana de los usuarios ya que pueden enterarse en tiempo real del clima, lo cual es de gran ayuda en el contexto actual debido al impacto del cambio climático no solo a nivel local sino global. Contar con una interfaz amistosa con el usuario también se vuelve importante ya que estamos trabajando con un público general y no especialistas en el área.

El sistema incluye una barra de búsqueda que permite al usuario escribir la ciudad en donde se encuentra, esto para proporcionar la información en tiempo real del clima de dicha zona y mostrarla en pantalla. También cuenta con una tabla que permitirá ver el clima de los días anteriores y una tabla para el clima del día de hoy.

El guardado de la información de cada día es vital ya que esto se usará para las tablas de los días anteriores, esto aplica tanto para las API's a consultar como al resultado de la comparación de estos datos y para ello se tendrán distintas colecciones en la base de datos MongoDB que cumplan esa función.

Además, la solución utilizará el framework Flet, que asegura la portabilidad y consistencia de la interfaz de usuario, facilitando así el despliegue y mantenimiento de la aplicación en diferentes entornos. Con estos procesos automatizados, la herramienta ofrecerá una solución integral y accesible para la consulta y visualización de datos meteorológicos, beneficiando tanto a estudiantes como a residentes locales.

6.1.2 Requerimientos Funcionales del Proyecto:

Los requerimientos funcionales describen acciones específicas que el ingeniero de software debe ser capaz de realizar durante el desarrollo de software. Los requerimientos funcionales a menudo se dividen en reglas de negocio y casos de uso.[5]

- **Consulta de las API's**: La extracción de los datos meteorológicos es fundamental para el proyecto, ya que estos datos son la base para todas las operaciones y análisis posteriores.
- **Almacenamiento de la información en la base de datos**: Es esencial poder almacenar los datos de manera eficiente y segura para facilitar su acceso y manipulación en cualquier momento.
- **Implementar una funcionalidad que permita la exportación de los datos meteorológicos**: Esto facilitará a los usuarios la descarga y el uso de la información en otras aplicaciones o para análisis adicionales, garantizando que los datos sean accesibles y utilizables según sus necesidades específicas.
- **Visualización de la información**: La información debe ser presentada al usuario mediante tablas y vistas intuitivas, asegurando una experiencia de usuario amigable y eficiente.

6.1.3 Requerimientos No Funcionales del Proyecto:

Los requerimientos no funcionales describen características específicas que el software debe poseer durante el desarrollo de la aplicación. Algunas de ellas podrían ser la seguridad, la interfaz, compatibilidad y rendimiento.[5]

Interfaz de Usuario Amigable: La interfaz debe ser intuitiva y fácil de usar, garantizando la comodidad del usuario.

Compatibilidad y Rendimiento: Utilizar el framework Flet garantiza una mayor compatibilidad y proporciona interfaces más amigables y fáciles de usar. Es crucial que las consultas a la base de datos y a las API's sean rápidas y eficientes para asegurar una experiencia de usuario satisfactoria.

6.1.4. Restricciones:

El desarrollo de la herramienta de consulta y visualización de datos meteorológicos enfrenta ciertas limitaciones técnicas y operativas que deben ser consideradas para garantizar su viabilidad y sostenibilidad a largo plazo. Estas restricciones se derivan principalmente de las características inherentes de las tecnologías utilizadas, así como de las condiciones impuestas por los proveedores de servicios externos. A continuación, se detallan las principales restricciones identificadas:

1. **Limitaciones en las Consultas a APIs Públicas** : Las APIs gratuitas, como OpenWeatherMap y VisualCrossing, imponen un límite estricto de 1,000 consultas diarias. Esto restringe la capacidad del sistema para manejar grandes volúmenes de datos en tiempo real, especialmente ante un incremento significativo de usuarios.

[5]Northware,2.022
https://www.northware.mx/blog/requerimientos-en-el-desarrollo-de-software-y-aplicaciones/

2. **Disponibilidad Exclusiva como Aplicación de Escritorio** : El diseño actual está orientado exclusivamente a una aplicación de escritorio, excluyendo a una población considerable de usuarios que prefieren o requieren acceso a través de dispositivos móviles. Esta limitación reduce el alcance del sistema y su accesibilidad para un público más amplio.
3. **Dependencia de Conectividad a Internet** : La herramienta depende de una conexión a internet activa para realizar consultas a las APIs externas y actualizar la base de datos MongoDB. Esto puede ser problemático en regiones con infraestructura deficiente o intermitente, afectando la disponibilidad de datos en tiempo real.

6.2 Fase de diseño:

Diagrama de acuerdo a la metodología:

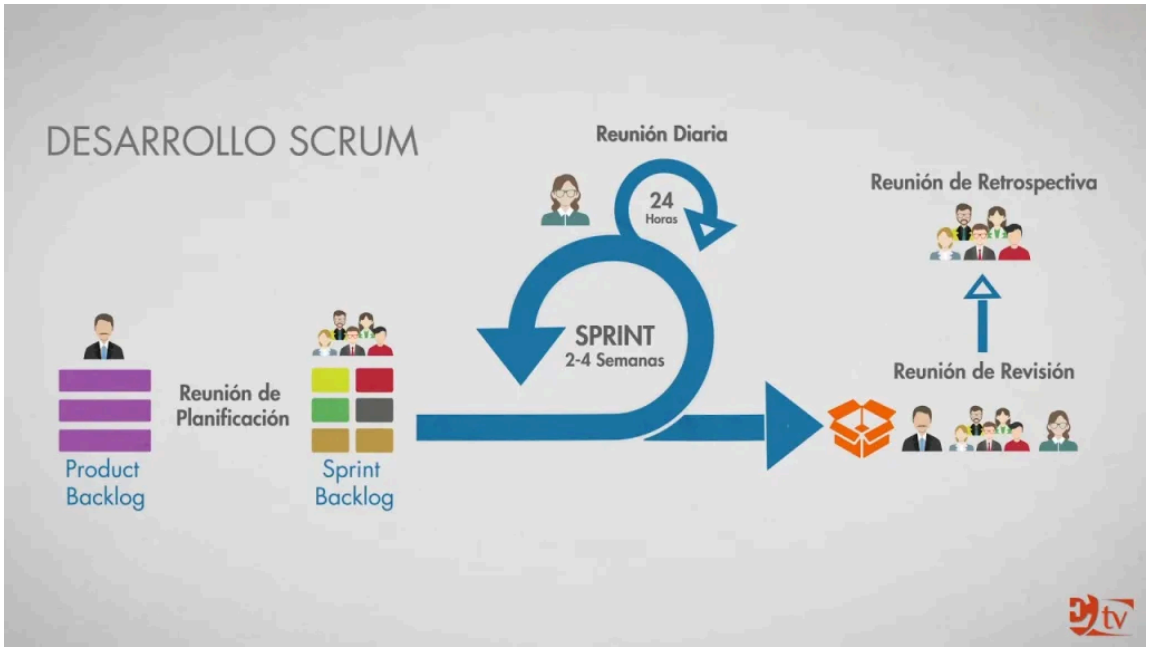
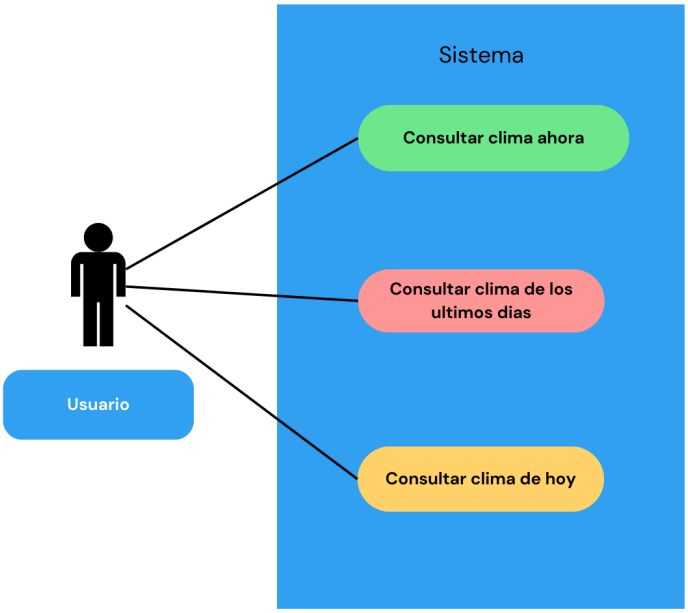


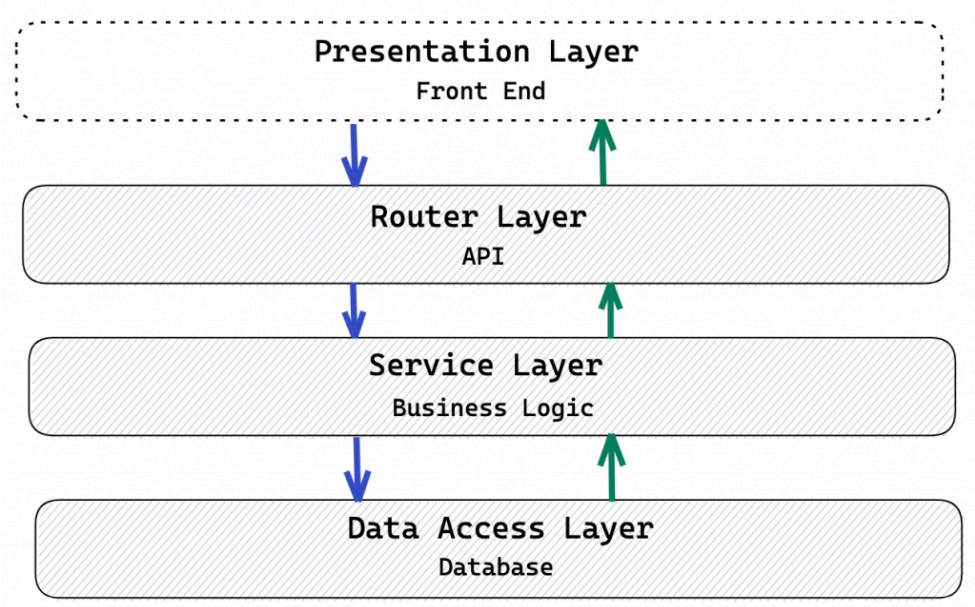
Figura [1]
Fuente:https://blog.iurlek.com/2020/02/como-convertirnos-en-una-empresa-agile-entendiendo-y-aplicando-scrum-de-una-forma-sencilla/, Febrero 2.020

Diagrama de casos de usos:



Figura[2] Roger Vera, 2.024

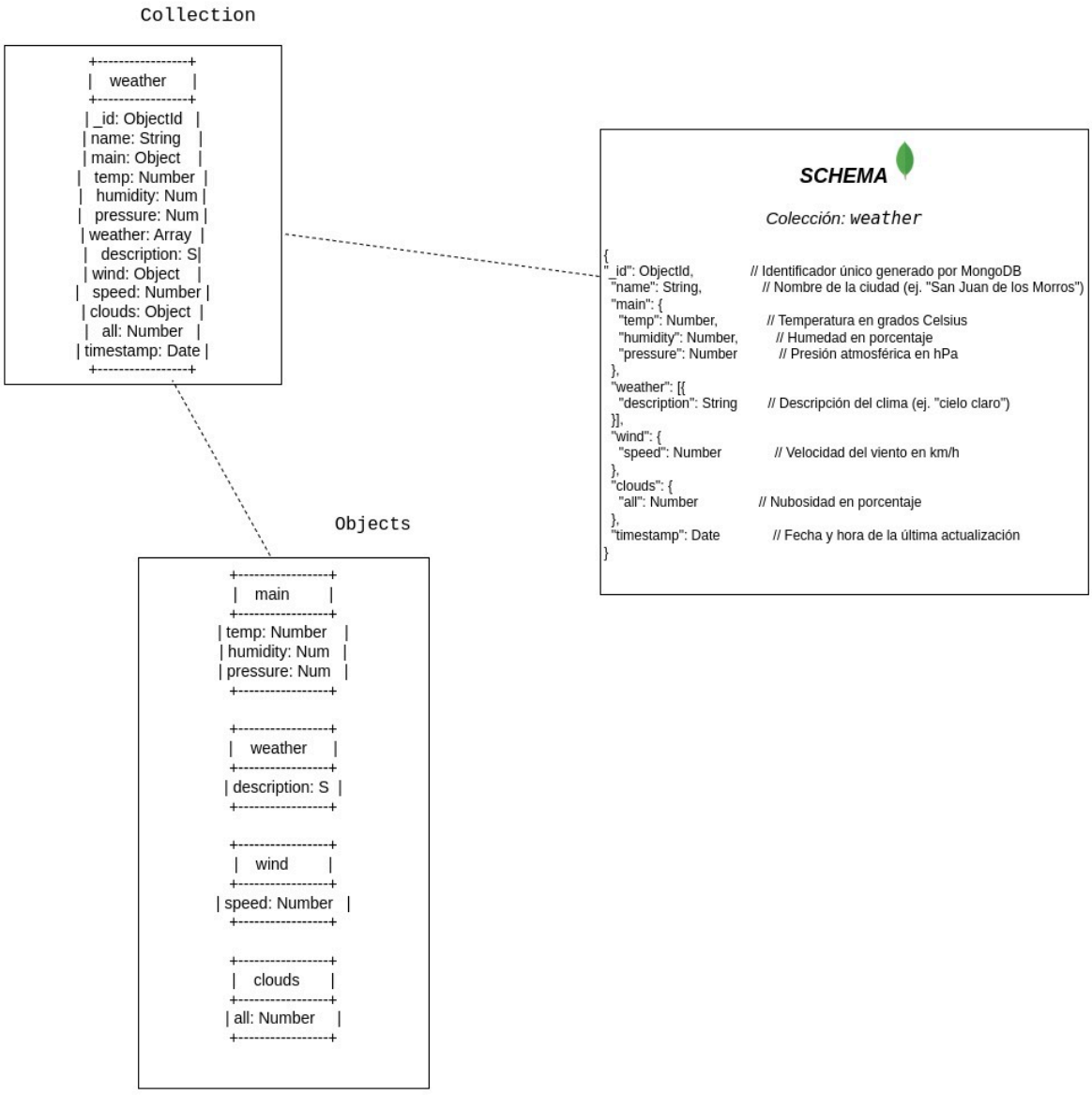
Diagrama de arquitectura de capas:



Figura[3].Yari Antonieta 2.021

https://ctrly.blog/nodejs-layered-architecture/

DIAGRAMA BASE DE DATOS



Figura[4].Roger Vera 2.024

Bocetos y desarrollo del sistema:

Figura[5] Roger Vera 2.024

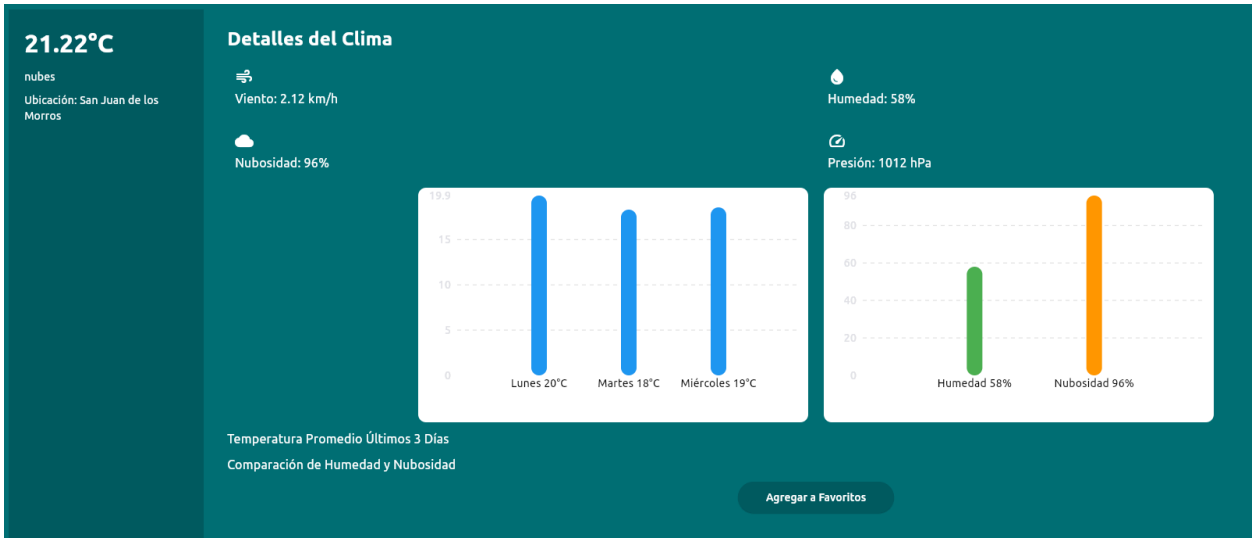
Vista del menú principal

```python
# Vista Principal
def main_view(page: ft.Page):
    page.clean()
    page.title = "Menú Principal"
    page.bgcolor = "#017075"
    page.padding = 40

    header = ft.Container(
        bgcolor="#015c63",
        padding=10,
        content=ft.Row([
            ft.Text("Smart Weather", size=24, color="white", weight=ft.FontWeight.BOLD),
            ft.IconButton(icon=ft.icons.SEARCH, on_click=lambda _: search_view(page), icon_color="white"),
        ], alignment=ft.MainAxisAlignment.SPACE_BETWEEN)
    )
```

Figura[6] Roger Vera 2.024

Código de la vista del menú principal.

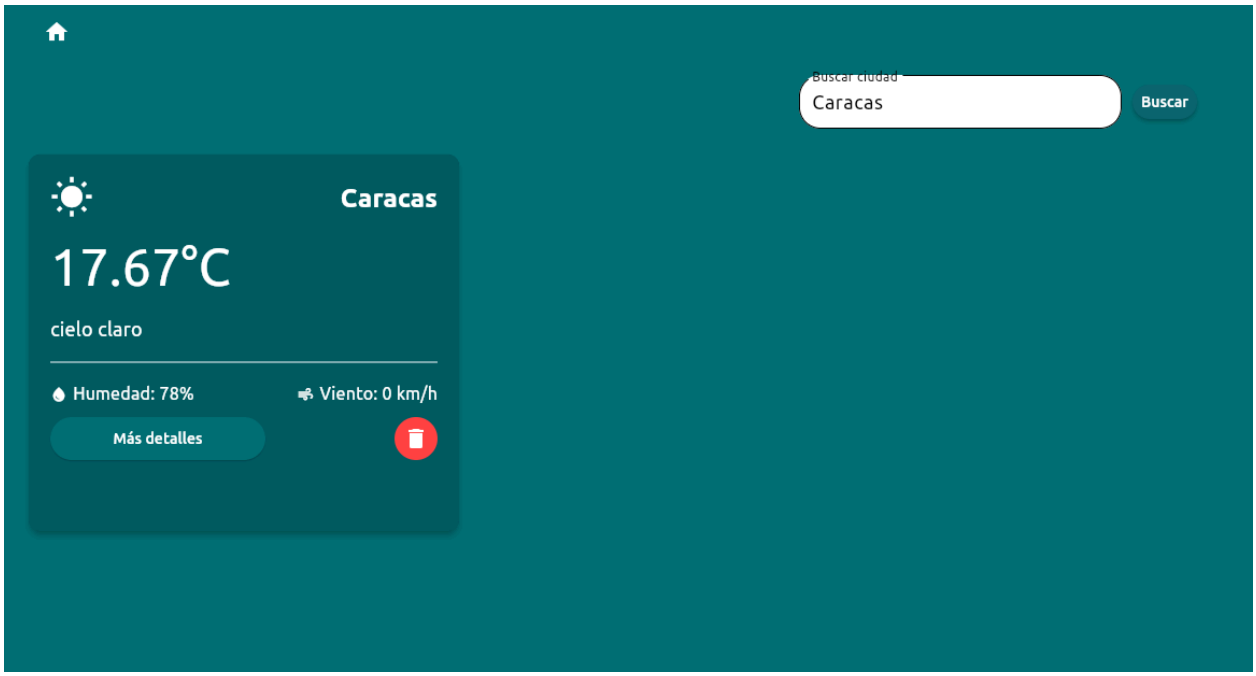


Figura[7] Roger Vera 2.024
Vista detallada de la tarjeta de la ciudad.

```
# Main Content con detalles adicionales, gráficos y botones
main_content = ft.Container(
    expand=True,
    bgcolor="#017075",
    padding=20,
    content=ft.Column([
        ft.Text("Detalles del Clima", size=24, color="#ffffff", weight=ft.FontWeight.BOLD),

        # Características del clima
        ft.Row([
            ft.Container(
                expand=True,
                bgcolor="#017075",
                border_radius=10,
                padding=10,
                content=ft.Column([
                    ft.Icon(ft.icons.AIR, size=24, color="white"),
                    ft.Text(f"Viento: {wind_speed} km/h", size=16, color="white"),
                ], spacing=5),
            ),
            ft.Container(
                expand=True,
                bgcolor="#017075",
                border_radius=10,
                padding=10,
                content=ft.Column([
                    ft.Icon(ft.icons.WATER_DROP, size=24, color="white"),
                    ft.Text(f"Humedad: {humidity}%", size=16, color="white"),
```

Figura[8] Roger Vera 2.024
Código de vista detallada



Figura[9] Roger Vera 2.024
Vista de búsqueda.

```python
# Contenedor para los resultados de búsqueda
results_container = ft.Column([], spacing=10)

# Función para manejar la búsqueda
def search_city(e):
    query = search_bar.value.strip()
    if query:
        results_container.controls.append(
            ft.Container(
                bgcolor="#015c63",
                padding=10,
                border_radius=10,
                content=ft.Column([
                    ft.Text(f"Ciudad: {query}", color="white", size=18, weight=ft.FontWeight.BOLD),
                    ft.Text("Temperatura: 25°C", color="white", size=16),
                    ft.ElevatedButton("Ver más detalles", on_click=lambda _: print(f"Detalles de {query}"), bgcolor="#017075", color="white")
                ], spacing=5)
            )
        )
        page.update()

search_button = ft.ElevatedButton("Buscar", on_click=search_city, bgcolor="#015c63", color="white")

# Layout principal
page.add(
    header,
    ft.Column([
        ft.Row([search_bar, search_button], alignment=ft.MainAxisAlignment.CENTER, spacing=10),
        results_container
    ], spacing=20, alignment=ft.MainAxisAlignment.CENTER)
)
```

Figura[10] Roger Vera 2.024
Código de menú de búsqueda.

6.3 Fase de Codificación:

6.3.1 Requerimientos de desarrollo:

-**Python**: Es el lenguaje de programación usado en el desarrollo del sistema, su uso en conjunto con Flet facilitó el desarrollo de las interfaces. Para el desarrollo de la lógica también fue usado junto a librerías como datetime y timedelta que ayudaron a la comparación de datos, las consultas a las APIs y las conexiones a la base de datos.

-**Flet**: Se implementó para el desarrollo de las interfaces del sistema, además de facilitar la creación de las interfaces también ofrece una ventaja para el desarrollo del potencial de esta herramienta ya que es posible crear tanto aplicaciones de escritorio, app web y aplicaciones móviles tanto para iOS como android.

-**MongoDB**: Se optó por una base de datos no relacional como MongoDB por el tipo de respuestas que ofrecen las APIs, ya que las mismas retornan un JSON es más fácil de manejar que una base de datos relacional. Además usando un modelo no relacional las consultas son más rápidas y permite manejar un gran volumen de datos con mayor facilidad.

6.3.2 Desarrollo de los módulos del sistema de información:

-**Módulo de consulta de datos:** En este módulo, se lleva a cabo la consulta de información mediante el uso de dos APIs externas especializadas en datos meteorológicos: OpenWeatherMap y VisualCrossing . Ambas interfaces proporcionan información detallada sobre condiciones climáticas actuales, pronósticos futuros y otros parámetros relevantes, los cuales son obtenidos a través de solicitudes HTTP utilizando endpoints específicos configurados según las necesidades del sistema. Una vez que los datos son recibidos en formato JSON, estos son procesados mediante algoritmos de transformación y validación para asegurar su integridad y consistencia.

Posteriormente, la información procesada es almacenada en una base de datos no relacional, donde se organizan en colecciones optimizadas para facilitar consultas eficientes. Este almacenamiento permite mantener un historial de los datos climáticos y garantiza que la información esté disponible para su uso en tiempo real o análisis posteriores. Finalmente, los datos son recuperados desde la base de datos mediante consultas estructuradas y son renderizados en las vistas correspondientes del sistema, ya sea en la vista detallada , que ofrece una representación exhaustiva de los datos meteorológicos, o en la vista de la página principal , que presenta un resumen visualmente accesible y adaptado para una experiencia de usuario más fluida.

-**Módulo de exportación de datos**: Este proceso se encarga de extraer la información almacenada en la base de datos para su posterior exportación a un archivo en formato PDF, garantizando así la disponibilidad de los datos meteorológicos de forma autónoma. La funcionalidad está diseñada para permitir al usuario acceder a un historial detallado de las condiciones climáticas registradas durante los

últimos 7 días, sin requerir una conexión activa a internet.

Para ello, se ejecutan consultas específicas sobre la base de datos, recuperando los registros relevantes que incluyen parámetros como temperatura, humedad, velocidad del viento, precipitaciones y otros indicadores meteorológicos. Estos datos son procesados y estructurados en un formato legible y organizado, optimizado para su representación visual en el archivo PDF.

El sistema utiliza bibliotecas o frameworks especializados en generación de documentos para formatear la información, asegurando que el archivo resultante sea compatible con diversos dispositivos y mantenga un diseño claro y profesional. Esta característica no solo facilita la consulta offline de los datos, sino que también proporciona una solución práctica para usuarios que necesitan acceder a la información en entornos con limitaciones de conectividad o para fines de auditoría y análisis retrospectivo.

7. Fase de Pruebas

7.1. Elaboración y Ejecución del Plan de Pruebas

TIPO DE PRUEBA	OBJETIVO	PARTICIPANTE	AMBIENTE	MÉTODO
UNITARIAS	Validar componentes individuales del sistema para asegurar su correcto funcionamiento.	Roger Vera	Entorno local.	Automatización con Selenium.
FUNCIONALES	Verificar que las funcionalidades principales cumplan con los requisitos especificados.	Roger Vera	Entorno local.	Uso de Selenium y Appium.
INTEGRACIÓN	Asegurar que los componentes del sistema interactúen correctamente entre sí.	Roger Vera	Entorno local.	Automatización con Selenium/Appium.

Tabla [2] Roger Vera 2.024

7.2. Análisis de Resultados:

TIPO DE PRUEBA	OBJETIVO	PARTICIPANTE	AMBIENTE	RESULTADO
UNITARIAS	Validar componentes individuales del sistema para asegurar su correcto funcionamiento.	Roger Vera	Entorno local.	No se detectaron errores en la lógica ni en la visualización del componente.
FUNCIONALES	Verificar que las funcionalidades principales cumplan con los requisitos especificados.	Roger Vera	Entorno local.	Los datos se muestran de forma precisa y consistente en la interfaz.
INTEGRACIÓN	Asegurar que los componentes del sistema interactúen correctamente entre sí.	Roger Vera	Entorno local.	No se detectaron fallos en la interacción entre los componentes durante las pruebas.

Tabla [3] Roger Vera 2.024

8. Conclusiones:

El desarrollo de la herramienta de consulta y visualización de datos meteorológicos representa un avance significativo para la comunidad académica de la UNERG y el público en general. Al integrar datos de diversas fuentes, almacenarlos en MongoDB y presentarlos mediante una interfaz intuitiva desarrollada con Flet, se logra una solución accesible y eficiente que facilita la toma de decisiones informadas sobre actividades diarias y preparación ante eventos climáticos.

La automatización de procesos como la extracción y visualización de datos permite superar las limitaciones de métodos manuales y fragmentados, ofreciendo información precisa y oportuna. Además, su diseño modular y escalable garantiza adaptabilidad para futuras mejoras, maximizando su impacto a largo plazo. Un aspecto particularmente innovador de esta herramienta es su funcionalidad de exportación de datos, que permite a los usuarios descargar la información meteorológica en formato PDF. Esta característica no solo facilita el acceso offline a los datos, sino que también proporciona una solución práctica para análisis retrospectivos, auditorías o uso en entornos con limitaciones de conectividad. Al democratizar el acceso a la información climática de esta manera, la herramienta se posiciona como una solución única y versátil en su categoría.

En conclusión, esta herramienta no solo contribuye a la eficiencia y transparencia en el acceso a datos meteorológicos, sino que también promueve una mayor conciencia ambiental y adaptabilidad ante los cambios climáticos, beneficiando tanto a la comunidad universitaria como al público general.

9. Recomendaciones:

Para abordar las restricciones mencionadas y maximizar el impacto y la funcionalidad de la herramienta, se proponen las siguientes recomendaciones técnicas y estratégicas:

1. **Optimización del Uso de APIs Públicas** :
   Para superar las limitaciones impuestas por las APIs gratuitas, se sugiere evaluar la migración a versiones pagas que ofrezcan mayores cuotas de consultas diarias y datos más detallados. Alternativamente, se podría implementar un mecanismo de caché en MongoDB para almacenar temporalmente los resultados de consultas frecuentes, reduciendo así la dependencia de llamadas externas.
2. **Desarrollo de Versiones Multiplataforma** :
   Dado que el framework Flet permite la creación de aplicaciones tanto de escritorio como móviles y web, se recomienda priorizar el desarrollo de una versión móvil y una interfaz web responsiva. Esto ampliará significativamente el alcance del sistema, permitiendo que una mayor cantidad de usuarios accedan a la información meteorológica desde sus dispositivos preferidos.
3. **Implementación de Mecanismos Offline y Sincronización** :
   Para mitigar los problemas asociados con la dependencia de conectividad a internet, se recomienda incorporar una funcionalidad de sincronización offline-online. Esto permitiría a los usuarios interactuar con la herramienta incluso en ausencia de conexión y sincronizar los datos

una vez que se restablezca la conectividad, mejorando así la experiencia del usuario en regiones con infraestructura limitada.

10. Referencias

[1] Editorial Etecé, 2.022, https://concepto.de/meteorologia/
[2] Miriam Martínez Canelo,2.024, https://profile.es/blog/que-es-la-arquitectura-de-software/
[3] Mauricio Costanzo, 2019,
https://platzi.com/tutoriales/1248-pro-arquitectura/5439-patron-arquitectonico-de-capas-layers/
[4] Julia Martins, 2.024, https://asana.com/es/resources/what-is-scrum
[5]Northware,2.022
https://www.northware.mx/blog/requerimientos-en-el-desarrollo-de-software-y-aplicaciones/