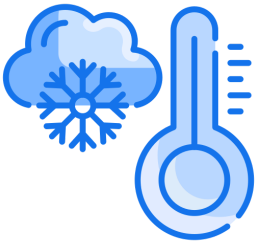
Condiciones de aprobación: Para aprobar debe sumar como mínimo 60 puntos y no menos del 50% en cada sección. Se puede recuperar por partes (teoría, objetos, arquitectura), pero no para la aprobación directa; en ese caso se deberá rendir el parcial completo nuevamente. Arquitectura se recuperará en conjunto primer y segundo parcial (Arq. parte I, Arq. parte II). Se podrá recuperar dos veces cada eje de la asignatura (objetos, datos, arquitectura).

BA Weather Alert



Contexto general

Estamos diseñando y desarrollando BA Weather Alert, una plataforma web y móvil impulsada por el Gobierno de la Ciudad de Buenos Aires. Su propósito es anticiparse a eventos climáticos adversos (heladas, nevadas, tormentas, etc.), y ofrecer a los ciudadanos información precisa, recomendaciones personalizadas y medios para reportar su experiencia.

La solución está compuesta por múltiples servicios distribuidos. En este examen, **nos enfocaremos en el diseño de los siguientes servicios:**

- Servicio de Alertas y Recomendaciones (a desarrollar)
- Servicio de Feedback Ciudadano (a desarrollar)

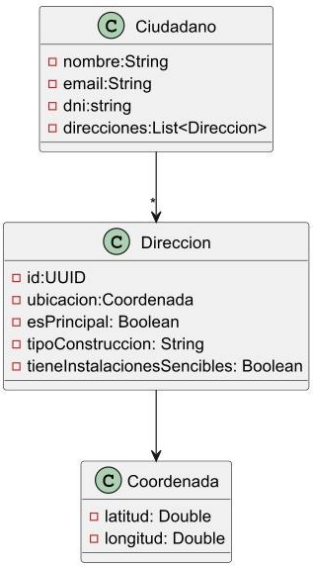
El resto de los componentes son servicios preexistentes o que están en proceso de desarrollo por otros equipos. En ambos casos, debemos integrarnos a ellos cuando lo necesitemos.

Servicio de Ciudadanos y Domicilios *(en desarrollo)*

El servicio de Ciudadanos gestiona los datos personales, domicilios principales y alternativos, así como detalles constructivos relevantes para emergencias climáticas.

Este servicio no debe ser modelado ni desarrollado pues ya lo está. Debemos utilizar su interfaz pública para realizar consultas cuando sea necesario.

El diseño de este servicio se encuentra documentado en el siguiente modelo de clases (ver al costado).



Servicio de Clima – Clima Proxy *(ya desarrollado)*

Este servicio se encarga de consultar a múltiples proveedores de clima, como AccuWeather y SMN, mediante integraciones pull-based (REST). Su responsabilidad es consolidar los datos y publicar eventos climáticos unificados.

El Clima Proxy expone un endpoint para que otros servicios se suscriban a novedades por tópico mediante webhooks. Es decir, otros servicios pueden registrar una URL que será invocada cuando ocurra un nuevo evento.

- Ejemplo de tópicos disponibles: */eventos/nevada, /eventos/helada, /eventos/tormenta*
- Ejemplo de notificación enviada vía webhook:

```
{
  "tipoEvento": "nevada",
  "fechaInicio": "2025-07-28T03:00:00Z",
  "fechaFin": "2025-07-28T12:00:00Z",
  "nivelSeveridad": "alto",
  "zonasAfectadas": [
    { "lat": -34.6037, "lng": -58.3816, "radio_km": 3 },
    { "barrio": "Villa Urquiza" }
  ]
}
```

Servicio de Alertas y Recomendaciones *(a desarrollar)*

Este servicio debe:

- Suscribirse a los eventos climáticos relevantes vía webhook (del Clima Proxy).
- Consultar al Servicio de Ciudadanos para obtener los domicilios afectados.
- Generar recomendaciones personalizadas para los eventos que ocurrirán en 24 hs con severidad *"alta", "crítica"* o *"extrema"*, basadas en:
    a. Tipo de evento
    b. Domicilio y condiciones del lugar (tipo de construcción, instalaciones sensibles, etc.)
- Para lo anterior, debemos consultar a un modelo externo *GPT4-Invierno*[1] (mediante POST) para enriquecer el texto de las recomendaciones.
- Enviar las alertas instantáneas frente a eventos con severidad *"extrema"* por:
    a. Notificación push[2] (a través de un servicio de mensajería interno)
    b. Email (vía servicio SMTP)

  Cada alerta debería dar aviso de la situación actual al usuario.
- Disparar webhooks de emergencia hacia organismos externos que se hayan registrado, como SAME o Defensa Civil, si la severidad del evento es "*extrema*".
- Activar una alarma física para la estación inteligente más cercana (ver más abajo) cuando el nivel del evento es *"crítico"* o *"extremo"* (debe ser configurable).

Servicio de Feedback Ciudadano *(a desarrollar)*

Este servicio debe permitir que los ciudadanos brinden su opinión posterior a que ocurra un evento climático. Debe incluir:

- Calificación de 1 a 5 estrellas.
- Respuestas a preguntas dinámicas (por ejemplo: "¿Las alertas fueron útiles?").
- Registro de acciones realizadas (por ejemplo: "Revisé mi caldera").

Debe almacenar esta información y ponerla a disposición del servicio de Estadísticas (por ejemplo, mediante una API REST). Todavía no está definido el medio persistente.

Además, debe permitir que los administradores modifiquen el conjunto de preguntas disponibles.

Servicio de Estaciones Inteligentes *(ya desarrollado)*

Existen actualmente unidades para monitorear datos climáticos y generar alertas, denominadas estaciones inteligentes. Este servicio fue desarrollado por otro proveedor y conecta cada estación inteligente con un broker de eventos, utilizando un modelo Pub/Sub. Cada estación emite eventos como:

- ***estacion.datos_climaticos*** (publicados periódicamente): brinda datos sobre la temperatura actual, la humedad y la presión atmosférica.
- ***estacion.alerta_manual***: Además de enviar datos meteorológicos en tiempo real, cada estación puede emitir eventos de alerta activados localmente, es decir, cuando una persona (personal de Defensa Civil o mantenimiento) en el lugar o desde una consola física activa una alarma (sirena, luces, etc.). Estos eventos pueden ser consumidos por otros servicios para, por ejemplo, registrar el evento, confirmar condiciones locales o realizar auditoría.

El Servicio de Alertas puede suscribirse a estos eventos si lo desea para mejorar la precisión de las recomendaciones.

Además, se ofrece una API REST para activar alarmas físicas en una estación determinada:

- Endpoint: POST /estaciones/{id}/activar-alarma

[1] GPT-4 es un modelo de inteligencia artificial desarrollado por OpenAI que utiliza técnicas avanzadas de aprendizaje profundo para comprender y generar texto con alta precisión y coherencia en una amplia variedad de tareas lingüísticas.

[2] Una notificación push es un mensaje emergente enviado desde una aplicación a un dispositivo móvil o computadora para alertar al usuario sobre actualizaciones, mensajes, o eventos importantes sin necesidad de que la aplicación esté abierta.

- Ejemplo de cuerpo JSON:

```
{
  "nivel": "critico",
  "duracion_segundos": 120,
  "componentes": ["sirena", "luz"]
}
```

- Valores posibles para nivel: `"preventivo","advertencia, "crítico"`
- Valores posibles para componentes: `"sirena", "luz", "pantalla" (mensaje de texto)`

Servicio de Estadísticas *(no desarrollado)*

Este servicio no debe ser implementado por el momento, pero sí debe ser tenido en cuenta en la arquitectura ya que recolectará información desde los servicios de Feedback y Alertas para generar reportes agregados.

Alcance y requerimientos

El Sistema deberá:

1. Permitir recibir eventos climáticos desde el servicio Clima Proxy mediante webhooks por tópico.
2. Consultar domicilios de ciudadanos a través del servicio ya desarrollado de Ciudadanos y Domicilios.
3. Generar recomendaciones personalizadas 24 horas antes de eventos climáticos.
4. Consultar a una API externa (GPT-invierno) para enriquecer el contenido de las recomendaciones.
5. Enviar alertas por webhook a organismos externos en caso de alertas extremas.
6. Integrar el servicio de Estaciones Inteligentes para recibir datos en tiempo real y activar alarmas.
7. Recolectar y guardar feedback de los ciudadanos luego de cada evento climático.
8. Permitir a administradores gestionar las preguntas del feedback.
9. Exponer información relevante al microservicio de Estadísticas.

Punto 1 - Modelo de Dominio (50 puntos)

1. ***(30 puntos) Servicio de Alertas y Recomendaciones; Servicio de Feedback Ciudadano***
   a. ***(7.5 puntos cada servicio) Para cada uno de los servicios:*** documentar la solución utilizando diagramas UML (diagrama de clases obligatorio), contemplando únicamente la capa de dominio.
   b. ***(7.5 puntos cada servicio) Para cada uno de los servicios:*** detallar las decisiones de diseño tomadas en el punto anterior. Mencione y haga referencia explícita a los siguientes conceptos que haya tenido en cuenta para diseñar su solución: Cualidades de Diseño y Atributos de Calidad de Software; Principios SOLID; Code Smells; Utilización de patrones de diseño (en caso que haya utilizado alguno) y comparación contra otra propuesta.
2. ***(10 puntos)*** Muestre mediante código o pseudocódigo cómo recibiría y guardaría un nuevo feedback de un Ciudadano. Muestre todas las capas involucradas. No es necesario que detalle la implementación de los repositorios.
3. ***(10 puntos)*** Teniendo en cuenta la propuesta dada del **Servicio de Ciudadanos y Domicilios**, indique si las siguientes afirmaciones son Verdaderas o Falsas. Justifique cada respuesta en no más de dos renglones.

   ☐ Representar tipoConstruccion como un String puede generar inconsistencias en los datos si no se aplican restricciones adicionales.

   _____
   _____

- ☐ Si quisiéramos que el usuario indique qué detalles constructivos deben ser tenidos en cuenta frente a alertas, bastaría con agregar una clase DetalleConstructivo y asociarla directamente a Dirección.

  _____
  _____

- ☐ Si en el futuro queremos modelar distintos tipos de direcciones (fiscales, temporales, laborales), lo ideal sería utilizar herencia, creando subclases de dirección.

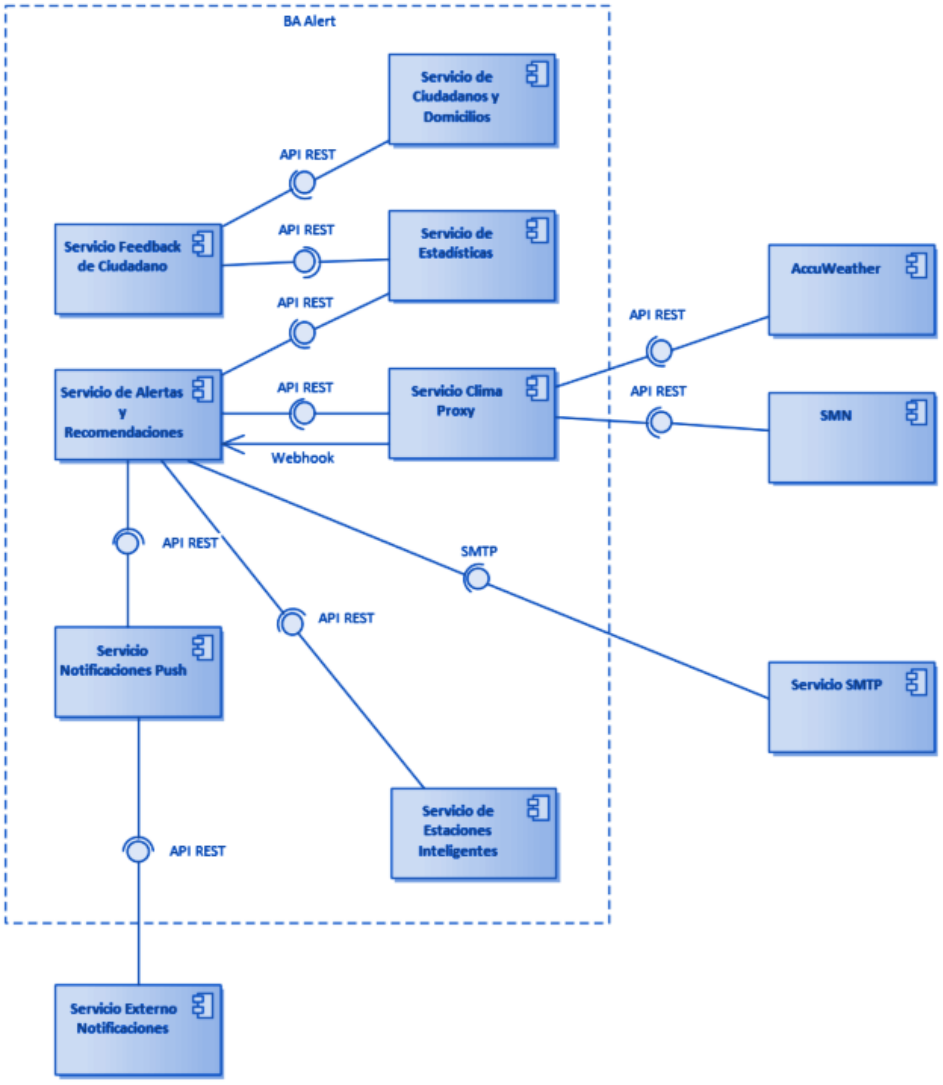
  _____
  _____

- ☐ El modelo actual no respeta el principio de responsabilidad única al incluir en la Dirección tanto información geográfica como lógica específica del dominio (tipo de construcción, instalaciones, etc.).

  _____
  _____

Punto 1 - Arquitectura parte I (50 puntos)

1) ***(20 puntos)*** Dado el siguiente Diagrama de Despliegue del Sistema, analícelo y describa mejoras, faltantes y puntos de mejora a considerar.



2) ***(20 puntos)*** ¿Cómo haría para que el servicio de Alertas sepa cuáles son las estaciones inteligentes disponibles a través del microservicio de Estaciones Inteligentes? ¿Cómo sabríamos cuál es la más cercana? Detalle, de forma completa y profunda, la interfaz que podría llegar a ofrecer el microservicio de Estaciones Inteligentes para tal fin.

3) ***(10 puntos)*** Indique si las siguientes afirmaciones son verdaderas. Justifique muy brevemente:

- ☐ Los webhooks utilizados por el Clima Proxy implican una integración push-based, donde los servicios suscriptos deben exponer una interfaz HTTP pública.

  _____

  _____

- ☐ Usar un broker para consumir datos de estaciones inteligentes permite mayor escalabilidad que hacer polling constante a su API.

  _____

  _____

- ☐ Programar una tarea periódica (cron task) para solicitar feedback a los usuarios luego de un evento climático es una alternativa razonable si no se cuenta con eventos disparadores.

  _____

  _____

- ☐ El microservicio de Alertas debería consumir eventos del Clima Proxy y al mismo tiempo exponer su propia API REST para que otras partes del sistema consulten alertas futuras.

  _____

  _____