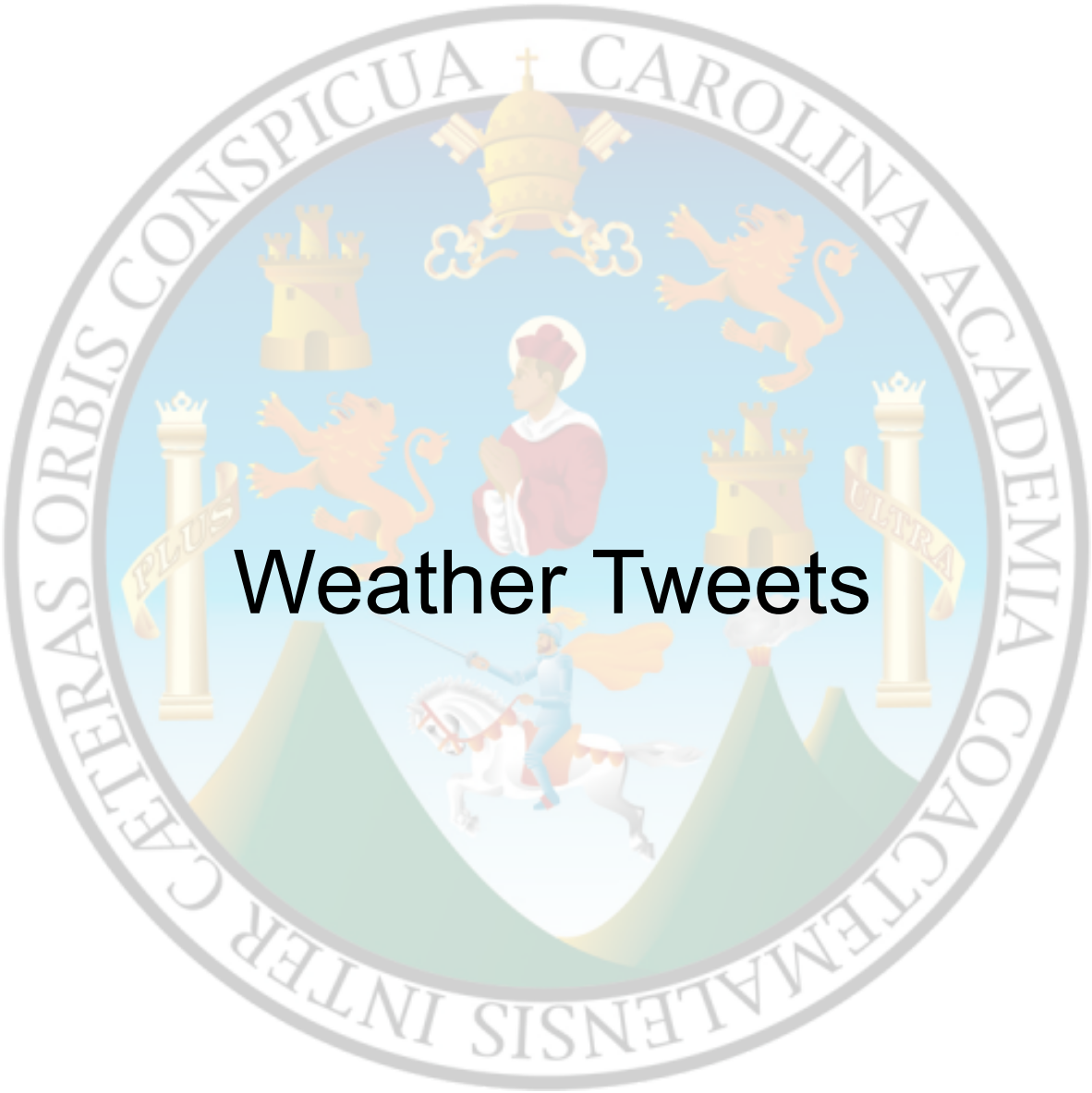
MANUAL
USUARIO



Weather Tweets

Tabla de Contenido

Introducción

El proyecto "Weather Tweets" es una aplicación distribuida diseñada para recolectar y mostrar tweets relacionados con el clima en tiempo real desde todo el mundo. Utiliza una arquitectura moderna y escalable basada en tecnologías como Kubernetes, Kafka, gRPC, Redis y MongoDB. El sistema está desarrollado para manejar altos niveles de concurrencia y permitir un monitoreo detallado del consumo de energía y las emisiones de CO2, alineado con objetivos de sostenibilidad ambiental.

Este manual está destinado a proporcionar a los usuarios las instrucciones necesarias para desplegar, configurar y utilizar la aplicación "Weather Tweets". Aquí encontrará detalles sobre los componentes del sistema, el proceso de despliegue, la interacción con la API, y las herramientas de monitoreo disponibles.

Objetivos

- **Optimizar el rendimiento del sistema** para manejar grandes volúmenes de solicitudes, asegurando que la arquitectura pueda escalar eficientemente.
- **Utilizar Docker y Kubernetes** para la gestión de contenedores y despliegue de aplicaciones, proporcionando un entorno consistente y fácil de gestionar.
- **Emplear gRPC** para la comunicación eficiente entre microservicios, aprovechando sus capacidades de alto rendimiento y baja latencia.
- **Establecer un sistema de procesamiento de mensajes de alto rendimiento** utilizando Kafka, permitiendo una transmisión de datos rápida y confiable.

Resumen

El proyecto "Weather Tweets" busca demostrar cómo una arquitectura distribuida puede manejar la recopilación y visualización de tweets sobre el clima a escala global. A través del uso de tecnologías avanzadas como Kubernetes, Kafka y gRPC, el sistema asegura la capacidad de manejar altas cargas de trabajo y ofrece un monitoreo detallado del consumo de recursos y el impacto ambiental.

La aplicación se despliega en un entorno Kubernetes, utilizando Strimzi para gestionar el clúster de Kafka, y emplea herramientas como Locust para generar tráfico de prueba. Los datos se almacenan en Redis y MongoDB, mientras que Grafana y Kepler se utilizan para la visualización y monitoreo del sistema y sus métricas ambientales.

Requisitos Previos

Antes de comenzar, asegúrese de tener lo siguiente:

- **Acceso a un clúster de Kubernetes (GKE recomendado):** Un clúster de Kubernetes bien configurado es esencial para desplegar y gestionar los diferentes servicios del sistema.
- **Docker instalado:** Docker se utiliza para construir y ejecutar los contenedores que componen la aplicación.
- **Acceso a un repositorio Git con el código fuente del proyecto:** Esto le permitirá clonar el repositorio y desplegar la aplicación.
- **Herramientas de línea de comandos como kubectl, helm y docker:** Estas herramientas son necesarias para gestionar los despliegues en Kubernetes y trabajar con contenedores Docker.

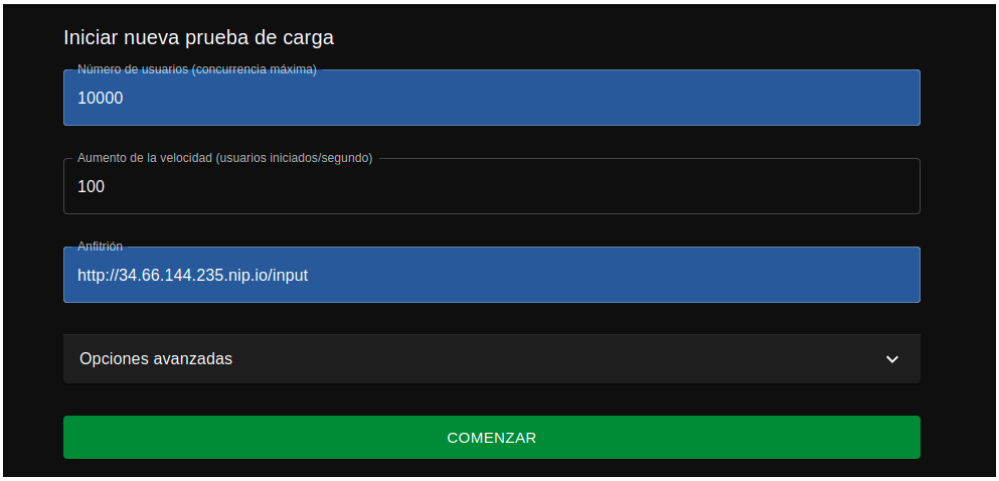
Uso de la Aplicación

La API de "Weather Tweets" permite la recolección de tweets relacionados con el clima y su almacenamiento en el sistema. Puede interactuar con la API utilizando herramientas como cURL, Postman, o integrando la API en sus propias aplicaciones.

Locust

Locust es una herramienta esencial en el proyecto "Weather Tweets" para garantizar que la arquitectura distribuida pueda manejar eficientemente la carga esperada y ofrecer un rendimiento óptimo en condiciones reales.

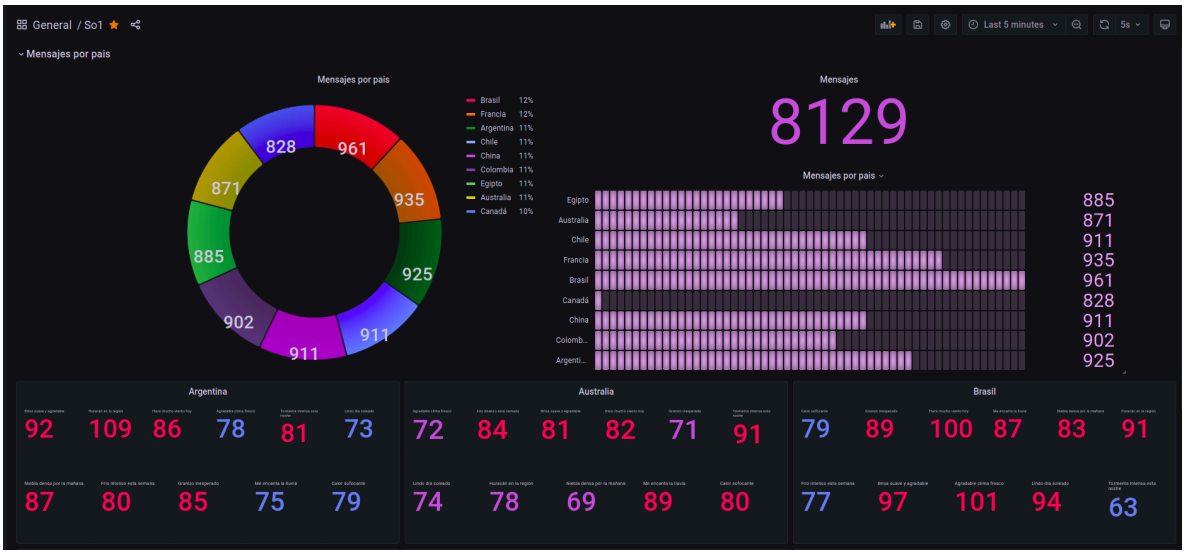
- **Instalación:** Para instalar Locust, siga las instrucciones de la documentación oficial en Locust.io.
- **Configuración:** Configure los scripts de prueba de carga para simular el tráfico de usuarios hacia la API de "Weather Tweets".
- **Ejecución:** Ejecute las pruebas de carga y monitoree el rendimiento del sistema. Ajuste las configuraciones según sea necesario para mejorar el rendimiento para su inicio ejecute el comando locust -f traffic.py, abra la url proporcionada y comience las pruebas.



Grafana

Grafana es una plataforma de código abierto para la analítica y monitorización interactiva. En el contexto del proyecto "Weather Tweets", Grafana se utiliza para visualizar las métricas y los datos recolectados por el sistema, proporcionando una interfaz gráfica que facilita el seguimiento del rendimiento y el estado del sistema.

- **Configuración:** Configure Grafana para conectarse a las bases de datos de Redis y MongoDB, así como a otras fuentes de datos necesarias.
- **Creación de Paneles:** Cree paneles personalizados para visualizar las métricas clave del sistema, como el rendimiento de la API, la latencia de los servicios, y el consumo de recursos.
- **Monitoreo en Tiempo Real:** Utilice Grafana para monitorear el estado del sistema en tiempo real, identificando rápidamente cualquier problema de rendimiento o disponibilidad.
- **Acceso:** Para su acceso verifique la url proporcionada en el manel de kubernetes o ingres al enlace acontinuación: http://34.66.144.235.nip.io/d/p0f-x1QSz/so1?orgId=1&refresh=5s



Kepler

Kepler (Kubernetes-based Efficient Power Level Exporter) es una herramienta de código abierto diseñada para monitorizar el consumo de energía y las emisiones de carbono en despliegues de Kubernetes. En el proyecto "Weather Tweets", Kepler se utiliza para medir y visualizar el impacto ambiental del sistema.

- **Instalación:** Instale Kepler en su clúster de Kubernetes siguiendo las instrucciones de la documentación oficial.
- **Configuración:** Configure Kepler para recolectar datos de consumo de energía y emisiones de CO2 de los diferentes pods y nodos en su clúster.
- **Visualización:** Integre Kepler con Grafana para visualizar las métricas ambientales en tiempo real.
- **Acceso:** Ingrese el comando kubectl port-forward svc/prometheus-grafana -n monitoring 4000:80 luego en su navegador únicamente ingrese localhost:4000 la cual le dará acceso al grafana proporcionado por kepler.

Monitoreo de Consumo de Energía

Acceder a Kepler

Para acceder a Kepler y visualizar las métricas de consumo de energía y emisiones de CO2:

1. Obtenga la IP externa del servicio Kepler:

   kubectl get svc --namespace project kepler
2. Acceda a Kepler en http://<EXTERNAL_IP>.

Visualizar Consumo de Energíaa

Utilice Kepler para monitorear el consumo de energía y emisiones de CO2 por Pod. Configure los paneles en Grafana para mostrar estas métricas de manera clara y comprensible.

Consideraciones

- **Seguridad:** Asegúrese de configurar las políticas de seguridad adecuadas para sus clústeres y servicios. Esto incluye la implementación de controles de acceso, la encriptación de datos en tránsito y en reposo, y la monitorización continua de posibles amenazas.
- **Escalabilidad:** Puede escalar los componentes según la demanda ajustando las configuraciones de despliegue en Kubernetes. Utilice herramientas de autoescalado para ajustar automáticamente el número de réplicas de sus servicios en función de la carga.
- **Mantenimiento:** Realice monitoreo continuo y actualizaciones periódicas para asegurar el correcto funcionamiento del sistema. Mantenga sus contenedores y dependencias actualizados para evitar vulnerabilidades de seguridad y problemas de compatibilidad.

Despliegue Detallado

Despliegue de Kafka

Instalación de Strimzi: Utilice Strimzi para gestionar el clúster de Kafka en Kubernetes. Siga las instrucciones en strimzi.io.

Configuración de Kafka: Configure los topics y brokers de Kafka según las necesidades de la aplicación.

Despliegue: Utilice los archivos YAML proporcionados para desplegar Kafka en su clúster.

Despliegue de Redis

Instalación de Redis: Utilice Helm para instalar Redis en su clúster de Kubernetes con el comando: helm install redis stable/redis

Configuración: Ajuste los parámetros de configuración para optimizar el rendimiento de Redis según sus necesidades.

Despliegue: Aplique los archivos YAML correspondientes para desplegar Redis en Kubernetes.

Despliegue de MongoDB

Instalación de MongoDB: Utilice Helm para instalar MongoDB en su clúster de Kubernetes utilizando el comando siguiente: helm install mongo stable/mongodb

Configuración: Configure las réplicas y los recursos asignados a MongoDB para asegurar un rendimiento adecuado.

Despliegue: Aplique los archivos YAML para desplegar MongoDB en Kubernetes.

Despliegue de Grafana

Instalación de Grafana: Utilice Helm para instalar Grafana en su clúster de Kubernetes usando el comando siguiente: helm install grafana stable/grafana

Configuración: Configure las fuentes de datos y los paneles en Grafana.

Despliegue: Aplique los archivos YAML para desplegar Grafana en Kubernetes.

Despliegue de Servicios gRPC

Configuración de gRPC: Desarrolle y configure los servicios gRPC necesarios para la comunicación entre microservicios.

Despliegue: Utilice los archivos YAML proporcionados para desplegar los servicios gRPC en Kubernetes.

Resolución de Problemas Comunes

- **Problemas de Conectividad:** Verifique las configuraciones de red y asegúrese de que los servicios estén correctamente expuestos.
- **Fallas en los Despliegues:** Revise los logs de los pods y servicios para identificar y solucionar errores.
- **Problemas de Rendimiento:** Utilice herramientas como Locust y Grafana para identificar cuellos de botella y ajustar las configuraciones de recursos.

Mantenimiento y Actualizaciones

- **Actualizaciones de Contenedores:** Mantenga los contenedores de Docker actualizados con las últimas versiones de las dependencias y librerías.
- **Monitoreo Continuo:** Utilice Grafana y Kepler para monitorear continuamente el rendimiento y el impacto ambiental del sistema.
- **Gestión de Seguridad:** Realice auditorías de seguridad regulares y aplique parches de seguridad a los componentes del sistema.