



# iWHITESTACK CHALLENGE!

Whitestack Challenge #4

Julio 2024

# Contenidos

---

Contenidos

Introducción

[Bienvenidos a nuestro cuarto Whitestack Challenge!](#)

[El stack de monitoreo “más popular”](#)

Desafío

[0. Probar accesos](#)

[1. Adaptar la aplicación web](#)

[2. Desplegar la aplicación](#)

[3. Crear serviceMonitor para obtención de métricas](#)

[4. Instalar y configurar Prometheus Adapter](#)

[5. Crear HPA usando la métrica externa](#)

[6. Generar carga y analizar](#)

[7. Documentación de proceso](#)

Criterios de evaluación

Participación del Challenge

Entrega del Challenge



## Introducción

### ¡Bienvenidos a nuestro cuarto Whitestack Challenge!

Whitestack les da la bienvenida y les agradece por participar en la cuarta edición de los Whitestack Challenges.

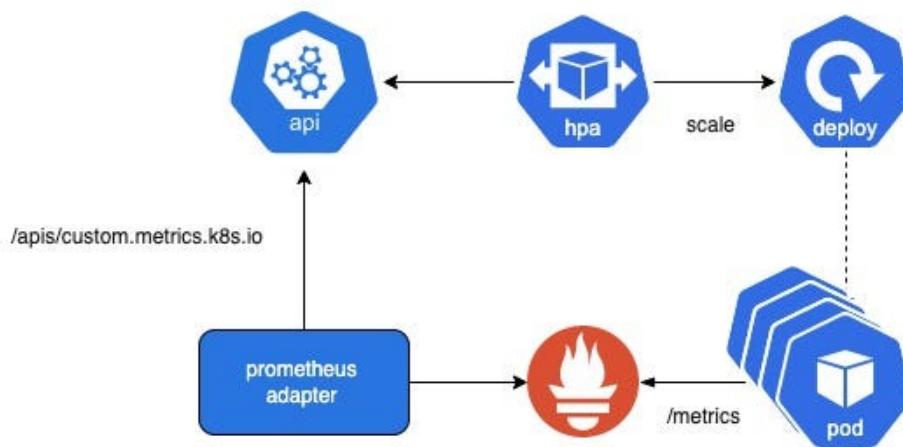
Estos últimos consisten en una serie de desafíos tecnológicos centrados alrededor del emocionante mundo del *Cloud Computing*, que pondrán a prueba sus habilidades como desarrolladores y les darán la oportunidad de desarrollar sus conocimientos con experiencias de primera mano.

Adicionalmente, serán acompañados por un equipo de profesionales dedicados a estas tecnologías, quienes les brindarán apoyo durante el desafío.

¡Esperamos que esta pueda ser una experiencia enriquecedora y que juntos sigamos creciendo en la comunidad del Cloud en Latinoamérica!

### El stack de monitoreo “más popular”

En esta oportunidad, nuestro Whitestack Challenge se enfoca en la flexibilidad y automatización del escalamiento para aplicaciones desplegadas sobre Kubernetes, el cual a su vez reside sobre una arquitectura de nube privada provista por Whitestack. En específico, se pretende utilizar métricas generadas por una app, prometheus y el HPA de kubernetes.



## Desafío

Primero, el participante deberá desplegar una aplicación web básica (Nginx) en su namespace asignado. Posteriormente, deberá configurar los objetos necesarios para que kubernetes pueda escalar la aplicación con el incremento de tráfico.

Las herramientas de monitoreo a utilizar serán:

- *Prometheus Operator*, el cual ya se encuentra instalado en el cluster de Kubernetes provisto por Whitestack (en caso de usar una infraestructura propia, será necesario instalarlo).
- *Prometheus Adapter*, el cual ya se encuentra instalado en el cluster de Kubernetes provisto por Whitestack (en caso de usar una infraestructura propia, será necesario instalarlo).

A continuación se presentan los pasos a seguir para completar el segundo Whitestack Challenge.

## 0. Probar accesos

Antes de comenzar el desafío, es importante probar el acceso al ambiente en el que vamos a desempeñar nuestras tareas.

Las interacciones con el cluster provisto se harán a través de la herramienta *kubectl*, por lo que esta debe estar instalada en su ambiente local. Los pasos de instalación de *kubectl* pueden ser vistos en la [documentación oficial de Kubernetes](#).

Para configurar *kubectl* para la comunicación con el cluster remoto usaremos un archivo configurador *kubeconfig*, el cual será entregado a cada participante con los datos específicos para su ambiente designado. Estas configuraciones le darán acceso al participante a un *namespace* del cluster. De esta manera se asegura que el trabajo de los participantes no afecte al resto.

El archivo tendrá la siguiente forma:

```
apiVersion: v1
kind: Config
clusters:
- name: "whitestackchallenge"
  cluster:
    server: "<dirección del cluster>"

users:
- name: "<usuario asignado al participante>"
```

```
user:  
  token: "<token del usuario>"  
  
contexts:  
- name: "whitestackchallenge"  
  context:  
    user: "<usuario asignado al participante>"  
    cluster: "whitestackchallenge"  
  
current-context: "whitestackchallenge"
```

Para configurar *kubectl* usando este archivo se debe exportar la variable de ambiente *KUBECONFIG* con su *path* como valor:

```
export KUBECONFIG="<path-al-archivo-de-config-provisto>"
```

Si las configuraciones fueron correctas, se debería ver el siguiente output al tratar de listar los pods del cluster:

```
$ kubectl get pods  
No resources found in <namespace-asignado-al-participante> namespace.
```

## 1. Adaptar la aplicación web

En el [repositorio del challenge](#) encontrarás una [aplicación web](#) básica, cuyos métodos y rutas no debes modificar.

Debes agregar un contador de requests solo para la ruta /heavywork, expuestos en /metrics en formato prometheus.

Hint: puedes usar la librería de python *prometheus\_client*.

## 2. Desplegar la aplicación

Debes crear una imagen de contenedor con la aplicación modificada.

Además, debes crear un helm chart que incluya los siguientes objetos:

- Deployment para la app
- Ingress
- Service

En este punto puedes probar acceder a tu aplicación, utilizando el Servicio del ingress-nginx-controller (NodePort o LoadBalancer)

### 3. Crear serviceMonitor para obtención de métricas

En tu namespace asignado, crear el *serviceMonitor* requerido para que Prometheus realice el scraping de las métricas personalizadas de la aplicación web.

### 4. Instalar y configurar Prometheus Adapter

En tu namespace asignado, instalar el prometheus-adapter para que prometheus pueda recolectar las métricas personalizadas.

La configuración de prometheus adapter está en un configmap, donde debes añadir las reglas necesarias para exponer estas métricas al API de kubernetes.

### 5. Crear HPA usando la métrica externa

En el namespace, crear un HPA con las siguientes características:

- Mínimo de réplicas: 1
- Máximo de réplicas: 5
- Tipo de target: Promedio
- Umbral: 10 peticiones por segundo

### 6. Generar carga y analizar

Para generar carga en tu aplicación, puedes utilizar el script [generate load.py](#) disponible en el [repositorio del challenge](#). De ser necesario, modificar los valores del script para aumentar o disminuir la carga.

Verificar que al usar la ruta */lightwork*, el deployment vuelve a tener una sola réplica, mientras que al usar la ruta */heavywork*, el deployment llegue hasta las 5 réplicas.

### 7. Documentación de proceso

Como parte de este challenge, se espera que los participantes documenten el trabajo realizado.

En este documento, y sin entrar en mucho detalle, se deben justificar las decisiones de diseño tomadas, como se resolvieron cada uno de los pasos y cualquier problema que se pueda presentar durante el challenge, junto con la forma en la que se soluciona.

## Criterios de evaluación

Los puntos que serán evaluados en este challenge serán los siguientes:

- La adaptación del código python: se evaluará la generación de métricas para el endpoint especificado
- La prueba de carga: se considerará tanto la representatividad de la prueba, como sus resultados obtenidos.
- Documentación: Se debe haber documentado el trabajo realizado durante el challenge de manera clara y concisa
- Creatividad: Se considerará el nivel de innovación y originalidad al momento de desarrollar el challenge.

## Participación del Challenge

Recuerda que para participar en el Challenge debes ratificar tu inscripción, siguiendo las instrucciones indicadas en:

<https://whitestack.com/es/whitestack-challenge/>

## Entrega del Challenge

El desafío puede ser desarrollado dentro de la Infraestructura que Whitestack ha dispuesto para este propósito. O bien, en caso que el participante lo desee, puede ser desarrollado dentro de un ambiente que la misma persona levante para el desarrollo del reto.

En caso que realices el desafío dentro de la infraestructura dispuesta por Whitestack, recomendamos que no elimines los recursos utilizados. Así, una vez concluido el plazo, podremos ver allí el resultado de tu trabajo.

Como entrega, generar un repositorio, y escribir a [whitestackchallenge@whitestack.com](mailto:whitestackchallenge@whitestack.com), indicando que has finalizado el Challenge #4 e incluir el link al repositorio GIT para revisar tu trabajo y lo que hayas documentado del mismo.

**SI TIENES CUALQUIER DUDA NOS PUEDES ESCRIBIR  
A [WHITESTACKCHALLENGE@WHITESTACK.COM](mailto:WHITESTACKCHALLENGE@WHITESTACK.COM)**

**iWHITESTACK  
CHALLENGE!**

**iWHITESTACK  
CHALLENGE!**