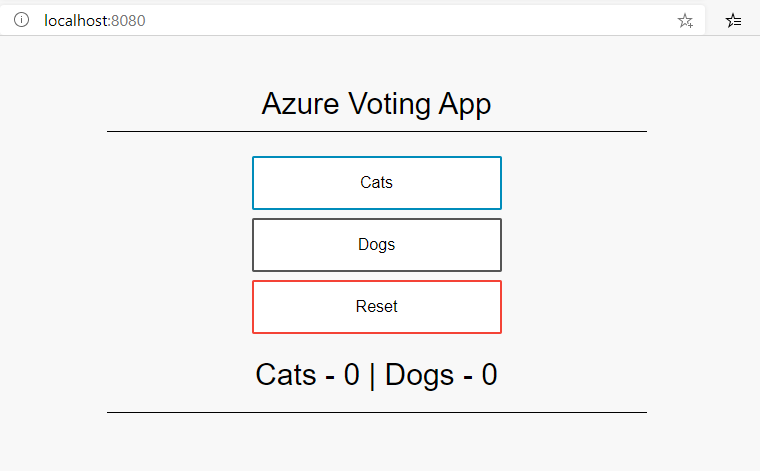
# **Tutorial: preparar una aplicación para Azure Kubernetes Service (AKS)**

* Artículo

En este tutorial, la primera parte de siete, se prepara una aplicación de varios contenedores para su uso en Kubernetes. Las herramientas de desarrollo existentes, como Docker Compose, se utilizan para compilar y probar una aplicación localmente. Tu aprendes como:

* Clonar una fuente de aplicación de muestra de GitHub
* Cree una imagen de contenedor a partir de la fuente de la aplicación de muestra
* Pruebe la aplicación de varios contenedores en un entorno Docker local

Una vez completada, la siguiente aplicación se ejecuta en su entorno de desarrollo local:



En tutoriales posteriores, la imagen del contenedor se carga en Azure Container Registry y luego se implementa en un clúster de AKS.

## **Antes de que empieces**

Este tutorial asume una comprensión básica de los conceptos básicos de Docker, como contenedores, imágenes de contenedores y dockercomandos. Para obtener información básica sobre los conceptos básicos de los contenedores, consulte [Primeros pasos con Docker](https://docs.docker.com/get-started/) .

Para completar este tutorial, necesita un entorno de desarrollo de Docker local que ejecute contenedores de Linux. Docker proporciona paquetes que configuran Docker en un [sistema Mac](https://docs.docker.com/docker-for-mac/) , [Windows](https://docs.docker.com/docker-for-windows/) o [Linux](https://docs.docker.com/engine/installation/#supported-platforms) .

Azure Cloud Shell no incluye los componentes de Docker necesarios para completar todos los pasos de estos tutoriales. Por lo tanto, recomendamos utilizar un entorno de desarrollo completo de Docker.

## **Obtener código de aplicación**

La [aplicación de muestra](https://github.com/Azure-Samples/azure-voting-app-redis) utilizada en este tutorial es una aplicación de votación básica que consta de un componente web front-end y una instancia de Redis de back-end. El componente web se empaqueta en una imagen de contenedor personalizada. La instancia de Redis usa una imagen sin modificar de Docker Hub.

Use [git](https://git-scm.com/downloads) para clonar la aplicación de muestra en su entorno de desarrollo:

git clone https://github.com/Azure-Samples/azure-voting-app-redis.git

Cambie al directorio clonado.

cd azure-voting-app-redis

Dentro del directorio se encuentra el código fuente de la aplicación, un archivo de composición de Docker creado previamente y un archivo de manifiesto de Kubernetes. Estos archivos se utilizan en todo el conjunto de tutoriales. El contenido y la estructura del directorio son los siguientes:

azure-voting-app-redis

**│ azure-vote-all-in-one-redis.yaml**

**│ docker-compose.yaml**

**│ LICENSE**

**│ README.md**

**│**

**├───azure-vote**

**│ │ app\_init.supervisord.conf**

**│ │ Dockerfile**

**│ │ Dockerfile-for-app-service**

**│ │ sshd\_config**

**│ │**

**│ └───azure-vote**

**│ │ config\_file.cfg**

**│ │ main.py**

**│ │**

**│ ├───static**

**│ │ default.css**

**│ │**

**│ └───templates**

**│ index.html**

**│**

**└───jenkins-tutorial**

**config-jenkins.sh**

**deploy-jenkins-vm.sh**

## **Crear imágenes de contenedores**

[Docker Compose](https://docs.docker.com/compose/) se puede utilizar para automatizar la creación de imágenes de contenedores y la implementación de aplicaciones de varios contenedores.

Utilice el archivo de muestra docker-compose.yamlpara crear la imagen del contenedor, descargue la imagen de Redis e inicie la aplicación:

Consola

Dupdo

docker-compose up -d

Cuando haya terminado, use el comando de [imágenes acoplables](https://docs.docker.com/engine/reference/commandline/images/) para ver las imágenes creadas. Se han descargado o creado tres imágenes. La imagen *de azure-vote-front* contiene la aplicación front-end y usa la imagen *de nginx-flask* como base. La imagen de *redis* se usa para iniciar una instancia de Redis.

$ docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

mcr.microsoft.com/azuredocs/azure-vote-front v1 84b41c268ad9 9 seconds ago 944MB

mcr.microsoft.com/oss/bitnami/redis 6.0.8 3a54a920bb6c 2 days ago 103MB

tiangolo/uwsgi-nginx-flask python3.6 a16ce562e863 6 weeks ago 944MB

Ejecute el comando [docker ps](https://docs.docker.com/engine/reference/commandline/ps/) para ver los contenedores en ejecución:

Dupdo

$ docker ps

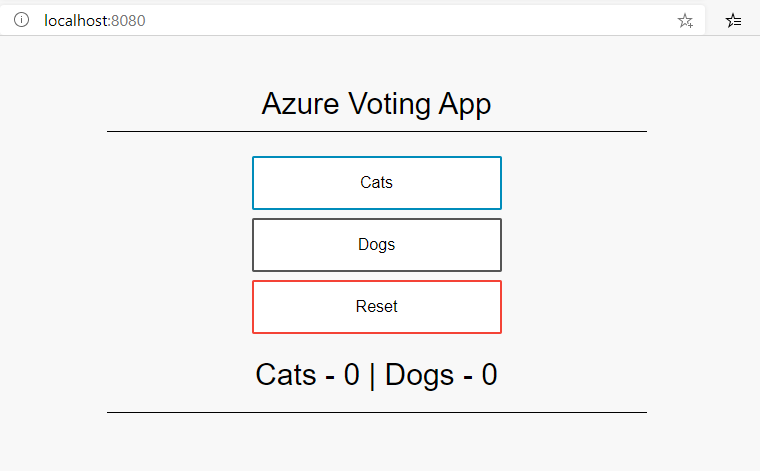
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

d10e5244f237 mcr.microsoft.com/azuredocs/azure-vote-front:v1 "/entrypoint.sh /sta…" 3 minutes ago Up 3 minutes 443/tcp, 0.0.0.0:8080->80/tcp azure-vote-front

21574cb38c1f mcr.microsoft.com/oss/bitnami/redis:6.0.8 "/opt/bitnami/script…" 3 minutes ago Up 3 minutes 0.0.0.0:6379->6379/tcp azure-vote-back

## **Probar la aplicación localmente**

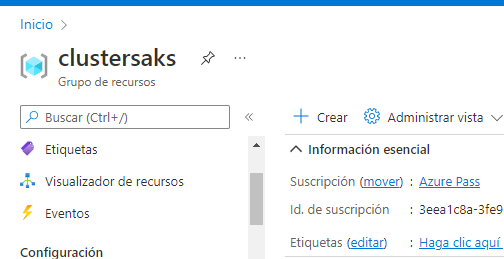
Para ver la aplicación en ejecución, ingrese http://localhost:8080en un navegador web local. La aplicación de muestra se carga, como se muestra en el siguiente ejemplo:



**Despliegue de Azure Voting App en un Cluster de Kubernetes**

**Creacion de un Cluster en AKS**

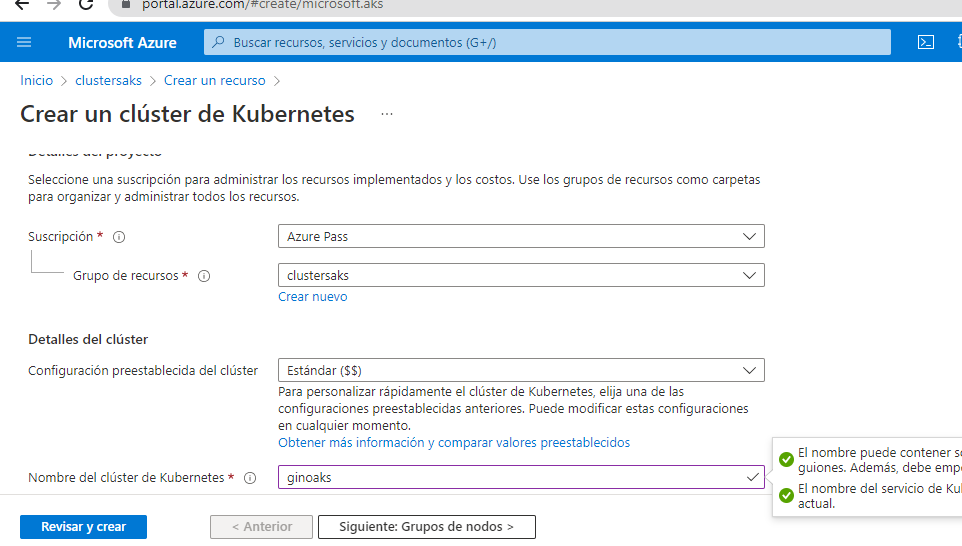
**dentro de un RG, seleccionamos “crear”**

****

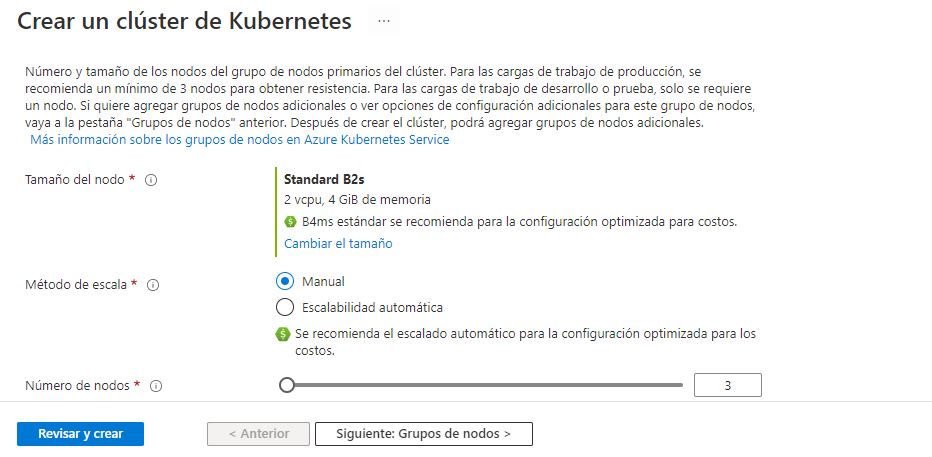
**Y seleccionamos “Crear un Cluster “**

****

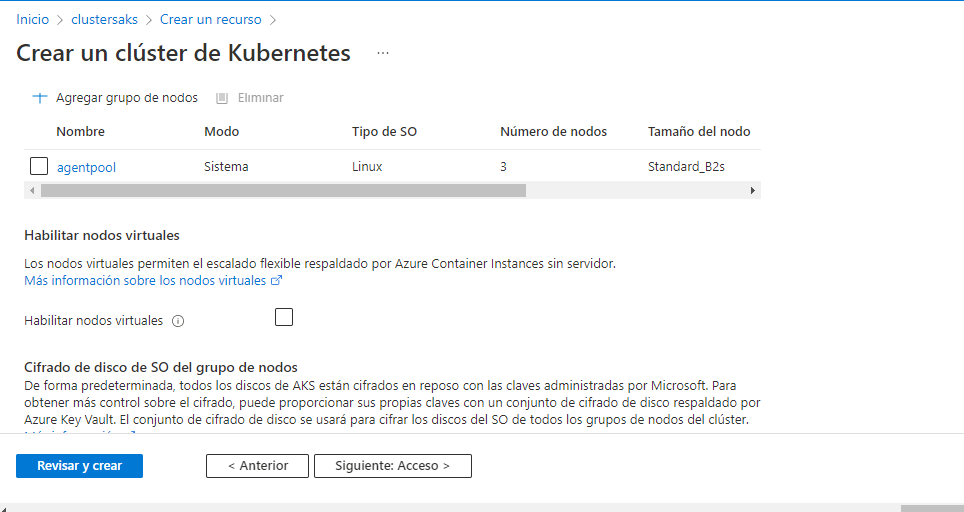
**Le brindo las siguientes configuraciones**

****

**Observen que cambie el tipo de instancia para los nodos**

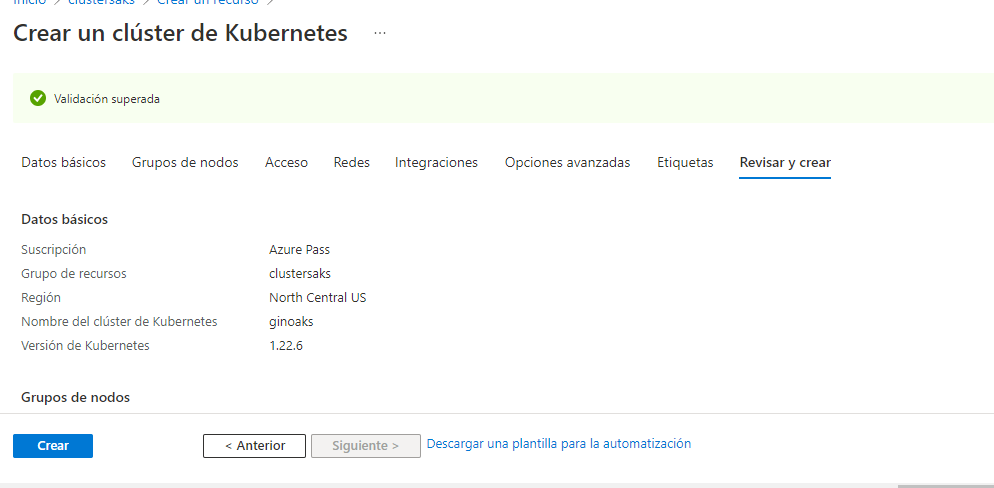
****

**vamos a Siguiente : Grupos de Nodos**

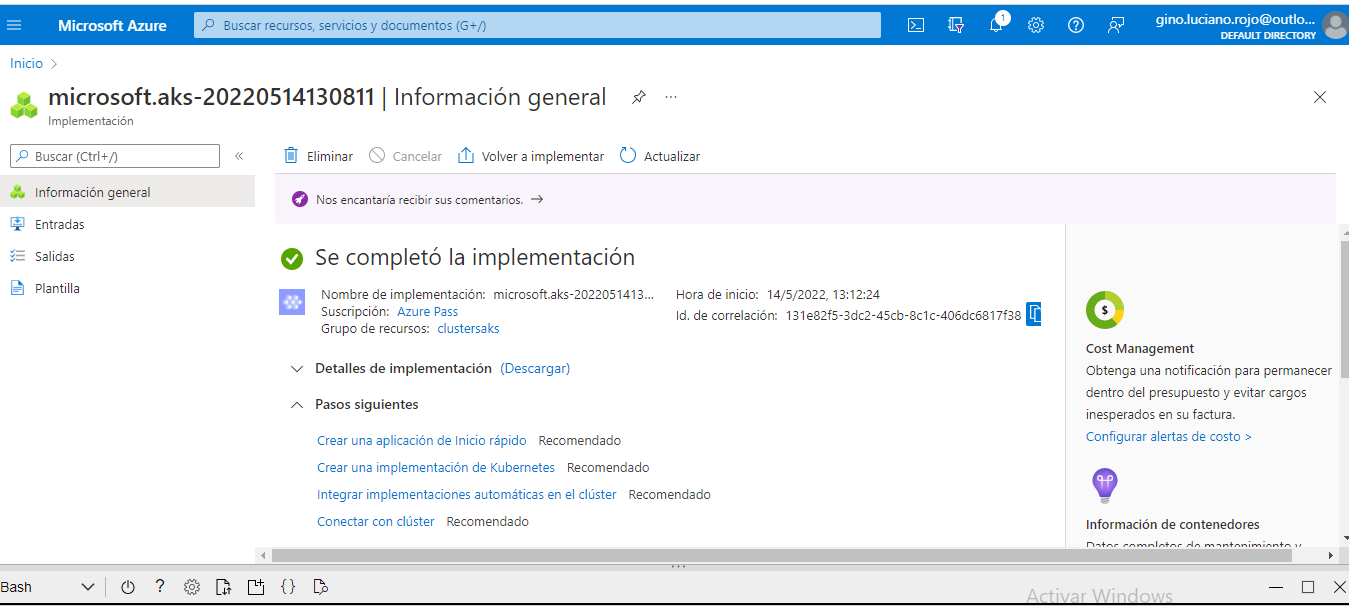
****

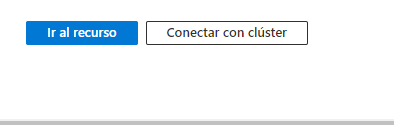
**Aqui dejamos todo tal cual esta hasta “Revisar y Crear”**

**Esperamos a que valide, y le damos a “Crear”**

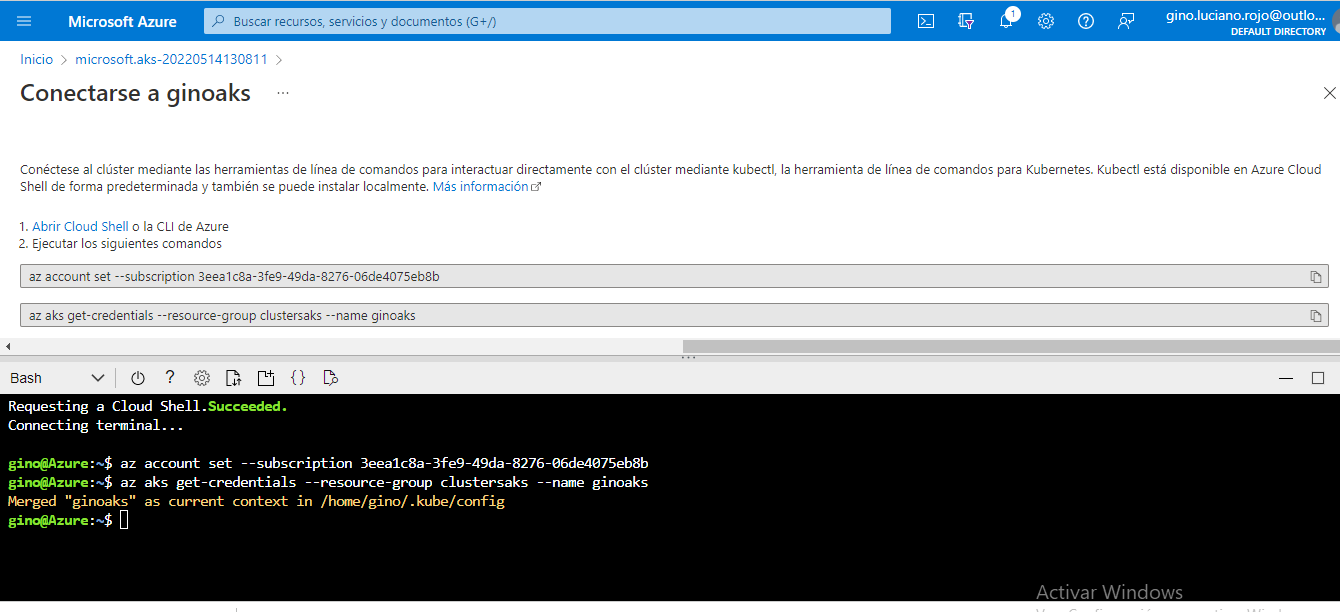
****

**hagamos click en “Conetctar con cluster”**

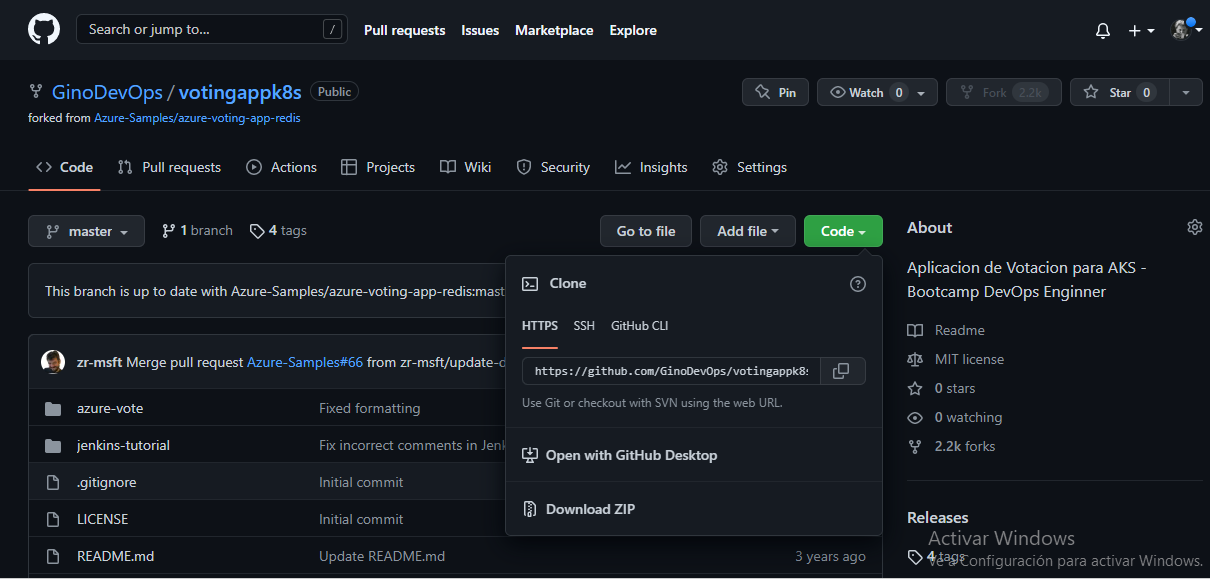
****

****

**y se vera asi**

****

**Donde ejecutamos**

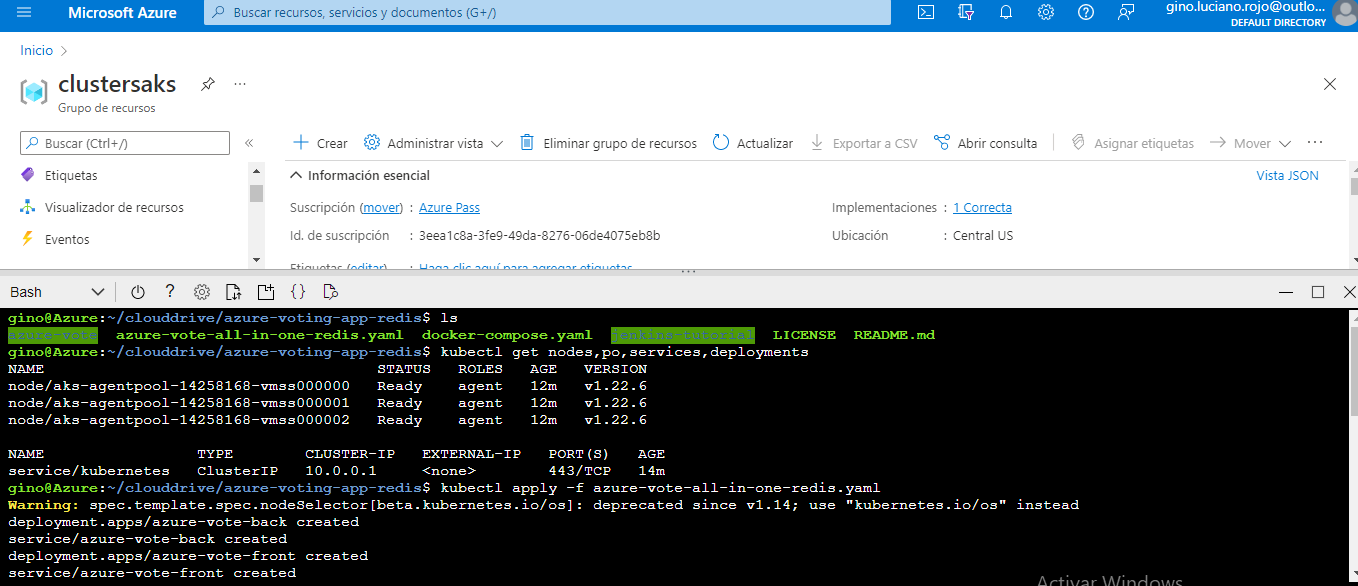
****

**git clone**

**git clone https://github.com/GinoDevOps/votingappk8s.git**

**y luego**

kubectl apply -f azure-vote-all-in-one-redis.yaml



kubectl apply -f azure-vote-all-in-one-redis.yaml

El siguiente resultado de ejemplo muestra los recursos creados correctamente en el clúster de AKS:

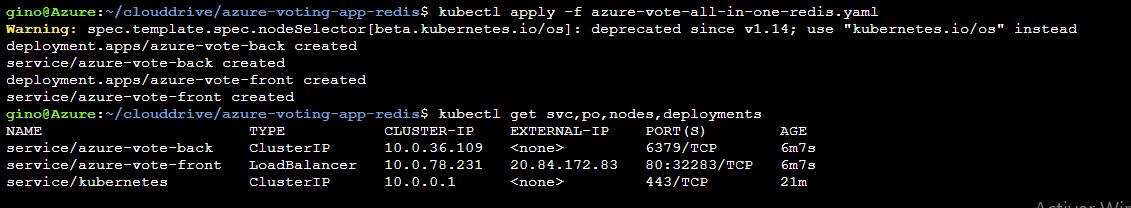
$ kubectl apply -f azure-vote-all-in-one-redis.yaml

deployment "azure-vote-back" created

service "azure-vote-back" created

deployment "azure-vote-front" created

service "azure-vote-front" created



## **Probar la aplicación**

Cuando se ejecuta la aplicación, un servicio de Kubernetes expone el front-end de la aplicación a Internet. Este proceso puede tardar unos minutos en completarse.

Para monitorear el progreso, use el [comando kubectl get service](https://kubernetes.io/docs/reference/generated/kubectl/kubectl-commands#get) con el --watch

kubectl get service azure-vote-front --watch

Inicialmente, la *IP EXTERNA* para el servicio *Azure-vote-front* se muestra como *pendiente* :

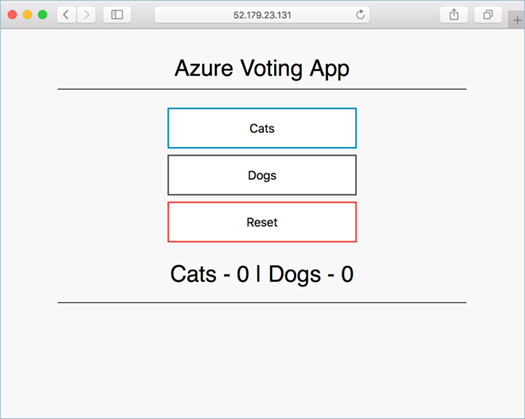
azure-vote-front LoadBalancer 10.0.34.242 <pending> 80:30676/TCP 5s

Cuando la dirección *IP EXTERNA* cambia de *pendiente* a una dirección IP pública real, use CTRL-Cpara detener la espera.

El siguiente ejemplo de salida muestra una dirección IP pública válida asignada al servicio:

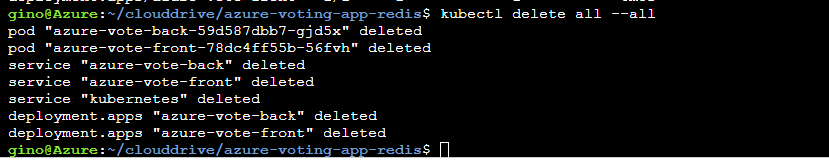
azure-vote-front LoadBalancer 10.0.34.242 52.179.23.131 80:30676/TCP 67s

Para ver la aplicación en acción, abra un navegador web a la dirección IP externa de su servicio:



Si la aplicación no se cargó, es posible que se deba a un problema de autorización con el registro de imágenes. Para ver el estado de sus contenedores, use el kubectl get pods

Eliminar todos los recursos



Ejecutamos kubectl delete all --all

## 

Dupdo

docker-compose do