Ejecutar sitios web puede ser difícil debido a la creación y administración de máquinas virtuales, clústeres, pods, servicios, etc. Esto está bien para aplicaciones más grandes y de varios niveles, pero si solo intenta implementar y hacer visible su sitio web, implica una sobrecarga considerable.

Con Cloud Run , la implementación de Google Cloud del framework Knative de Google , puedes administrar e implementar tu sitio web sin la sobrecarga de infraestructura que experimentas con una máquina virtual o implementaciones basadas exclusivamente en Kubernetes. Este enfoque no solo es más sencillo desde el punto de vista de la administración, sino que también te permite escalar a cero cuando no hay solicitudes en tu sitio web.

Cloud Run permite el desarrollo sin servidor en contenedores y puede ejecutarse en sus propios clústeres de Google Kubernetes Engine (GKE) o en una solución PaaS totalmente administrada proporcionada por Cloud Run. En este laboratorio, se ejecutará este último escenario.

Los ejercicios están ordenados para reflejar una experiencia común de desarrollador en la nube:

1. Crea un contenedor Docker desde tu aplicación
2. Implementar el contenedor en Cloud Run
3. Modificar el sitio web
4. Implementar una nueva versión sin tiempo de inactividad

Lo que aprenderás

En este laboratorio aprenderá a:

- Cree una imagen de Docker con Cloud Build y cárguela en Artifact Registry
- Implementar imágenes de Docker en Cloud Run
- Administrar implementaciones de Cloud Run
- Configurar un punto final para una aplicación en Cloud Run

Tarea 1. Clonar el repositorio de origen

Dado que está implementando un sitio web existente, solo necesita clonar la fuente, para poder concentrarse en crear imágenes de Docker e implementar en Cloud Run.

1. En Cloud Shell, ejecute los siguientes comandos para clonar el repositorio git y cambiar al directorio apropiado:

```
git clone https://github.com/googlecodelabs/monolith-to-microservices.git

cd ~/monolith-to-microservices
```

2. Instale las dependencias de NodeJS para que pueda probar la aplicación antes de implementarla:

```
./setup.sh
```

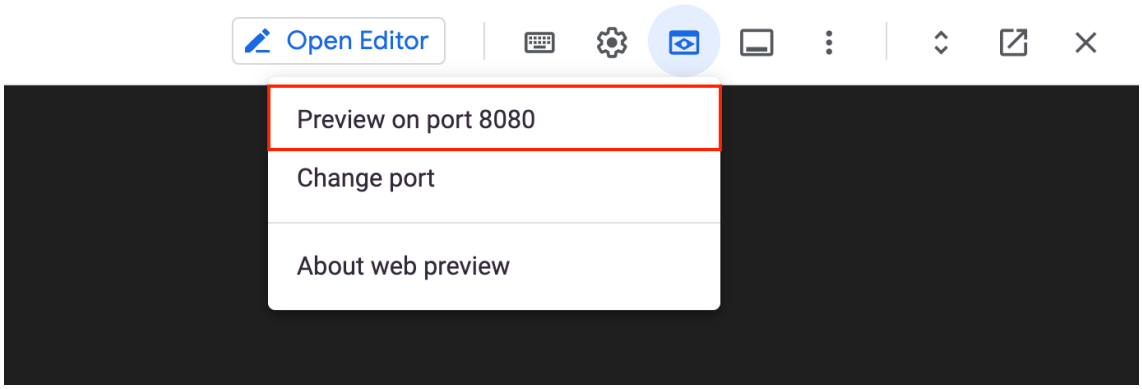
Tardará unos minutos en ejecutarse. Verá un mensaje de éxito al finalizar.

3. Pruebe su aplicación ejecutando el siguiente comando para iniciar el servidor web:

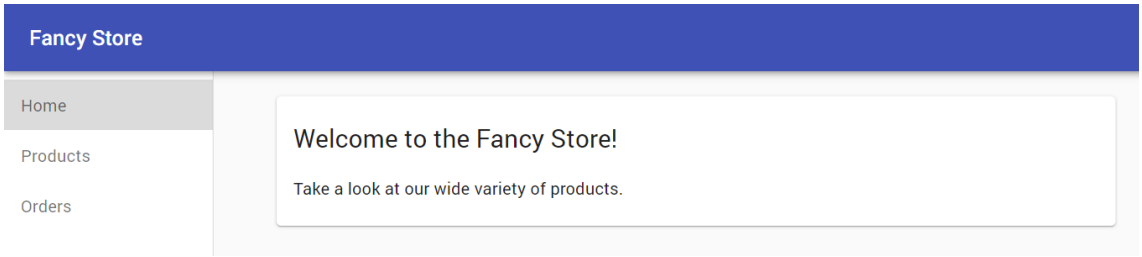
cd ~/monolith-to-microservices/monolith

npm start

4. Obtenga una vista previa de su aplicación haciendo clic en el ícono de vista previa web y seleccionando **Vista previa en el puerto 8080** .



Esto debería abrir una nueva ventana donde podrás ver tu página web de Fancy Store en acción.



5. Cierre esta ventana después de ver el sitio web y detenga el proceso del servidor web presionando **CTRL+C** en Cloud Shell.

Tarea 2. Crear un contenedor Docker con Cloud Build

¡Ahora que tienes los archivos fuente listos, es hora de dockerizar tu aplicación!

Normalmente, se necesitaría un enfoque de dos pasos: crear un contenedor Docker y enviarlo a un registro para almacenar la imagen y que GKE la extraiga. Cloud Build permite crear el contenedor Docker y enviar la imagen a Artifact Registry con un solo comando.

Cloud Build comprimirá los archivos del directorio y los moverá a un contenedor de Cloud Storage. El proceso de compilación tomará todos los archivos del contenedor y usará el Dockerfile, presente en el mismo directorio, para ejecutar el proceso de compilación de Docker.

Crear el repositorio Docker de destino

Debe crear un repositorio antes de poder enviar imágenes. Enviar una imagen no activa la creación de un repositorio, y la cuenta de servicio de Cloud Build no tiene permisos para crear repositorios.

1. En la consola, busque **Artifact Registry** en el campo de búsqueda, luego haga clic en el resultado **Artifact Registry** .
2. Haga clic en **Crear repositorio** .
3. Especifique monolith-demo como nombre del repositorio.
4. Elija **Docker** como formato.
5. En Tipo de ubicación, seleccione Región y luego elija la ubicación Región.
6. Haga clic en **Crear** .

Configurar la autenticación

Antes de poder enviar o recibir imágenes, configure Docker para usar la CLI de Google Cloud para autenticar solicitudes en Artifact Registry.

- Para configurar la autenticación en los repositorios de Docker en la región Región, ejecute el siguiente comando en Cloud Shell:

```
gcloud auth configure-docker Region-docker.pkg.dev
```

El comando actualiza tu configuración de Docker. Ahora puedes conectarte con Artifact Registry en tu proyecto de Google Cloud para enviar y recibir imágenes.

Implementar la imagen

Ahora implementará la imagen que se creó anteriormente.

1. Primero, debe habilitar las API de Cloud Build, Artifact Registry y Cloud Run. Ejecute el siguiente comando en Cloud Shell para habilitarlas:

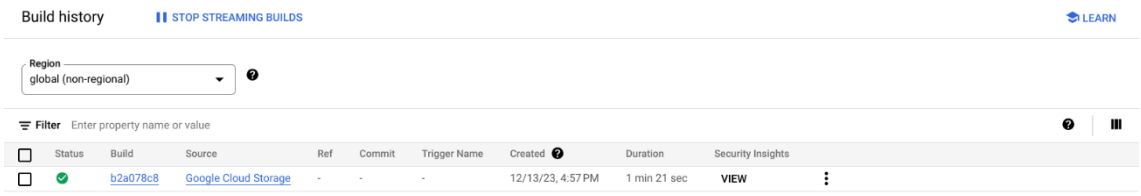
```
gcloud services enable artifactregistry.googleapis.com \
  cloudbuild.googleapis.com \
  run.googleapis.com
```

2. Una vez habilitadas las API, ejecute el siguiente comando para iniciar el proceso de compilación:

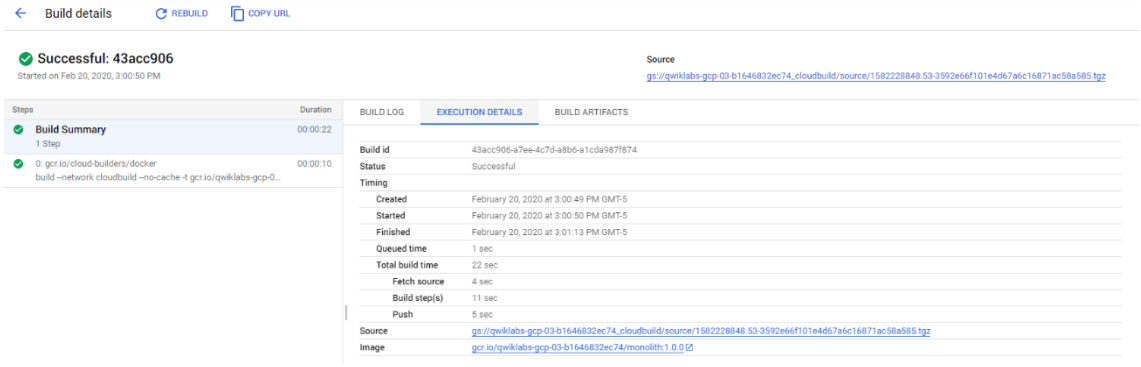
```
gcloud builds submit --tag Region-
docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/monolith-demo/monolith:1.0.0
```

Nota: Este proceso tardará unos minutos.

3. Para ver su historial de compilación o ver el proceso en tiempo real, en la consola, busque **Cloud Build** y luego haga clic en el resultado **de Cloud Build** .
4. En la página **Historial** puedes ver una lista de todas tus compilaciones; solo debería haber una que acabes de crear.



- Si hace clic en el ID de compilación, podrá ver todos los detalles de esa compilación, incluida la salida del registro.
- Desde la página Detalles de compilación, puede ver la imagen del contenedor que se creó haciendo clic en la pestaña **Detalles de ejecución** y luego en el enlace de la imagen.



Tarea 3. Implementar el contenedor en Cloud Run

Ahora que ha contenedorizado su sitio web y enviado el contenedor a Artifact Registry, ¡es momento de implementarlo en Cloud Run!

Hay dos enfoques para realizar la implementación en Cloud Run:

- **Cloud Run administrado** : El modelo de Plataforma como Servicio donde todo el ciclo de vida del contenedor es gestionado por el propio producto Cloud Run. Utilizará este enfoque en este laboratorio.
- **Cloud Run en GKE** : Cloud Run con una capa adicional de control que te permite incorporar tus propios clústeres y pods desde GKE. Puedes leer más aquí .

1. Ejecute el siguiente comando para implementar la imagen en Cloud Run:

gcloud run deploy monolith --image Región-docker.pkg.dev/ ${GOOGLE_CLOUD_PROJECT} /monolith-demo/monolith: 1 . 0 . 0 --region Región

2. Cuando se le solicita permitir invocaciones no autenticadas , [monolith]escriba **Y.**

Verificar la implementación

1. Para verificar que la implementación se creó correctamente, ejecute el siguiente comando:

gcloud run services list

Este resultado muestra varias cosas. Puedes ver la implementación, así como el usuario que la implementó (tu correo electrónico) y la URL que puedes usar para acceder a la aplicación. ¡Parece que todo se creó correctamente!

2. Haz clic en la URL proporcionada en la lista de servicios. Deberías ver el mismo sitio web que previsualizaste localmente.

Nota: También puede ver sus implementaciones de Cloud Run a través de la consola si navega a **Cloud Run** en el **menú de Navegación** .

Tarea 4. Crear una nueva revisión con menor concurrencia

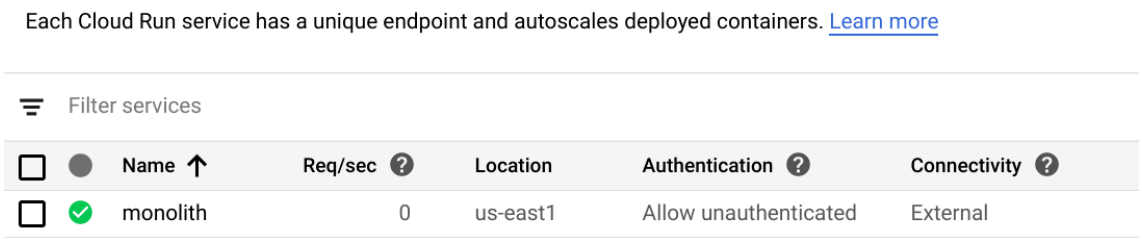
En esta sección volverás a desplegar tu aplicación, pero esta vez ajustando uno de los parámetros.

De forma predeterminada, una aplicación de Cloud Run tendrá un valor de concurrencia de 80, lo que significa que cada instancia de contenedor atenderá hasta 80 solicitudes a la vez. Esto supone una gran diferencia con el modelo de Funciones como Servicio, donde una instancia gestiona una solicitud a la vez.

1. Ejecute el siguiente comando para volver a implementar la misma imagen de contenedor con un valor de simultaneidad de 1 (solo para probar) y vea qué sucede:

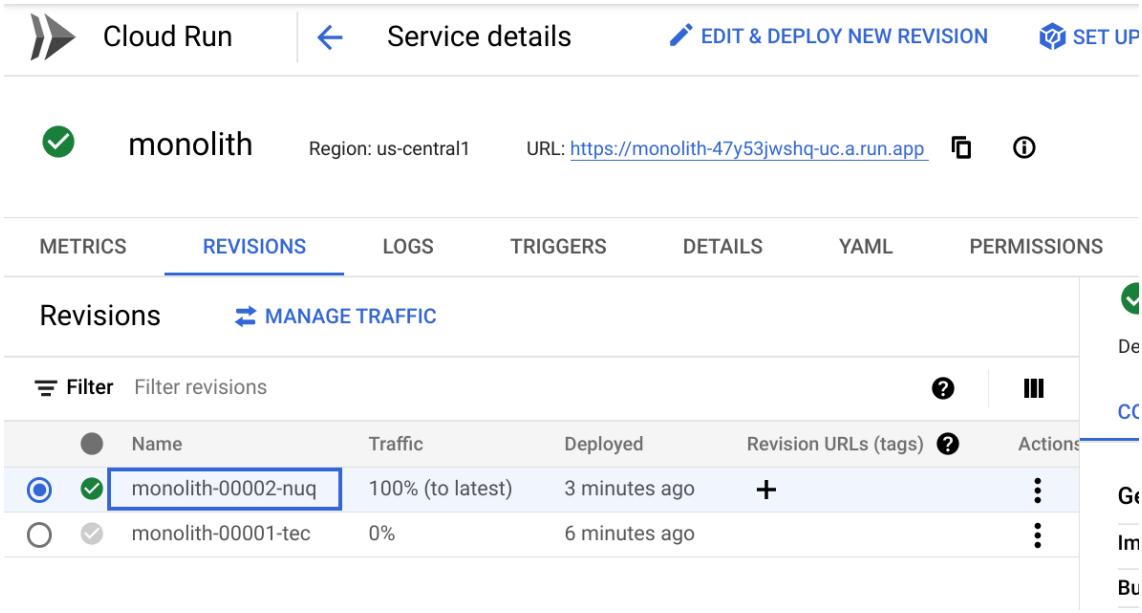
gcloud run deploy monolith --image Region-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/monolith-demo/monolith:1.0.0 --region Region --concurrency 1

2. Para ver los detalles, desde el **Menú de navegación** , haga clic en **Cloud Run** y luego haga clic en el servicio **monolito** :

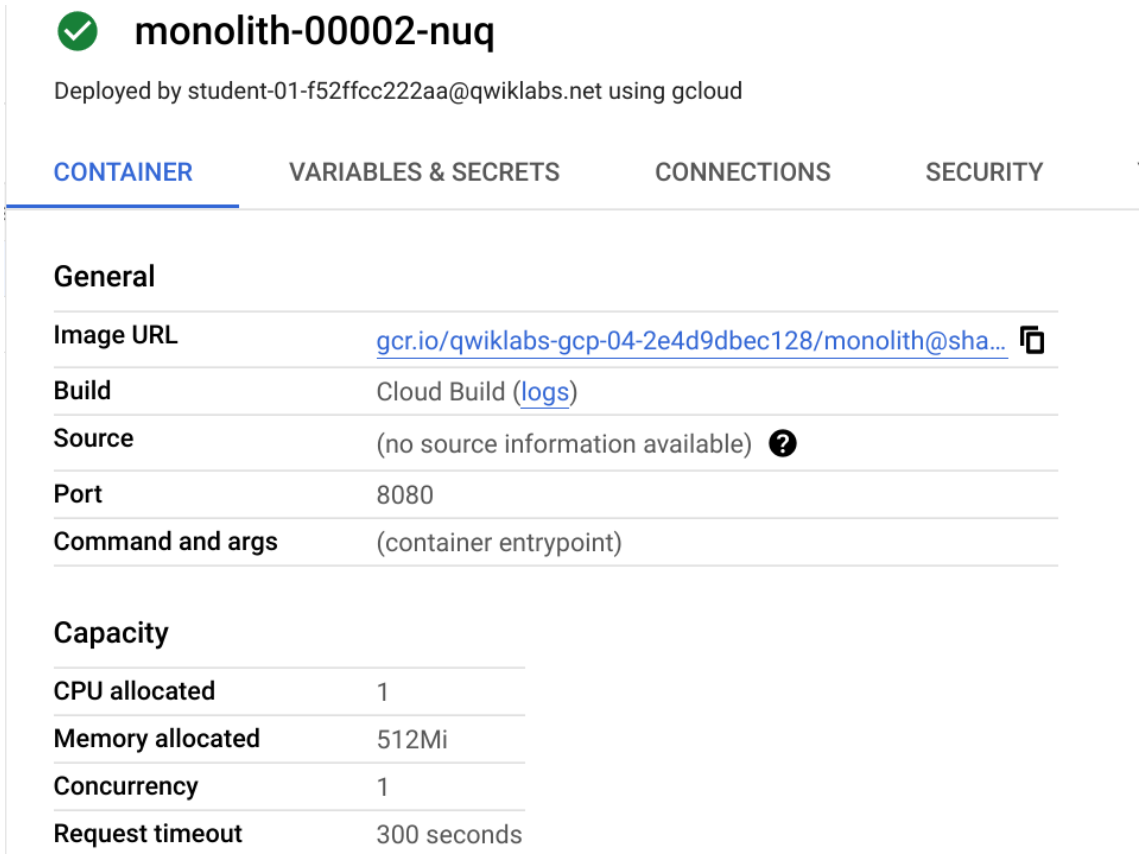


3. En la página "Detalles del servicio", haga clic en la pestaña **"Revisiones"** . Debería ver dos revisiones creadas.

La implementación más reciente tiene detalles en el lado derecho.



Verá que el valor de concurrencia se ha reducido a "1".



Si bien esta configuración es suficiente para realizar pruebas, en la mayoría de los escenarios de producción tendrá contenedores que admitan múltiples solicitudes simultáneas.

A continuación, puede restaurar la concurrencia original sin volver a implementar. Puede restablecer el valor de concurrencia predeterminado de "80" o simplemente establecerlo en "0", lo que eliminará cualquier restricción de concurrencia y lo establecerá en el máximo predeterminado (que es 80).

4. Ejecute el siguiente comando para actualizar la revisión actual, utilizando un valor de concurrencia de 80:

gcloud run deploy monolith --image Region-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/monolith-demo/monolith:1.0.0 --region Region --concurrency 80

Notarás que se ha creado otra revisión, que ahora se ha redirigido el tráfico y que la concurrencia ha vuelto a ser 80.

Nota: Es posible que tengas que salir de la pestaña **Revisiones** y luego regresar a ella para ver la información más actualizada.

Tarea 5. Realizar cambios en el sitio web

Escenario: Su equipo de marketing le ha pedido que cambie la página de inicio de su sitio web. Creen que debería ofrecer más información sobre su empresa y sus productos y servicios.

Tarea: ¡Añadirás texto a la página de inicio para que el equipo de marketing esté contento! Parece que uno de nuestros desarrolladores ya creó los cambios con el nombre de archivo index.js.new. Puedes copiar este archivo a [nombre del archivo index.js] y los cambios deberían reflejarse. Sigue las instrucciones a continuación para realizar los cambios necesarios.

1. Ejecute los siguientes comandos para copiar el archivo actualizado al nombre de archivo correcto:

   cd ~/monolith-to-microservices/react-app/src/pages/Home

   mv index.js.new index.js

2. Imprima su contenido para verificar los cambios:

cat ~/monolith-to-microservices/react-app/src/pages/Home/index.js

3. Ejecute el siguiente comando para compilar la aplicación React y copiarla en el directorio público del monolito:

cd ~/monolith-to-microservices/react-app

npm run build:monolith

Ahora que el código está actualizado, reconstruye el contenedor Docker y publícalo en Artifact Registry. Puedes usar el mismo comando que antes, solo que esta vez actualizarás la etiqueta de la versión.

4. Ejecute el siguiente comando para activar una nueva compilación en la nube con una versión de imagen actualizada de 2.0.0:

cd ~/monolith-to-microservices/monolith

gcloud builds submit --tag Region-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/monolith-demo/monolith:2.0.0

En la siguiente sección, utilizará esta imagen para actualizar su aplicación sin tiempo de inactividad.

Tarea 6. Actualizar el sitio web sin tiempo de inactividad

Los cambios están completos y el equipo de marketing está satisfecho con las actualizaciones. Es hora de actualizar el sitio web sin interrumpir a los usuarios. Cloud Run considera cada implementación como una nueva *revisión* que primero se pondrá en línea y luego se redirigirá el tráfico.

De forma predeterminada, se asignará el 100 % del tráfico entrante de un servicio a la última revisión. Es posible usar "Rutas" para asignar diferentes porcentajes de tráfico a diferentes revisiones dentro de un servicio. Siga las instrucciones a continuación para actualizar su sitio web.

- Ejecute el siguiente comando para volver a implementar el servicio para actualizar la imagen a una nueva versión con el siguiente comando:

gcloud run deploy monolith --image Region-docker.pkg.dev/${GOOGLE_CLOUD_PROJECT}/monolith-demo/monolith:2.0.0 --region Region

Verificar la implementación

1. Valide que su implementación se haya actualizado ejecutando el siguiente comando:

gcloud run services describe monolith --platform managed --region Region

Aquí verá que el Servicio ahora está utilizando la última versión de su imagen, implementada en una nueva revisión.

Para verificar los cambios, navegue a la URL externa del servicio Cloud Run, actualice la página y observe que se ha actualizado el título de la aplicación.

2. Ejecute el siguiente comando para enumerar los servicios y ver la URL del servicio:

   gcloud beta run services list

3. Haz clic en la URL del servicio. Tu sitio web debería mostrar ahora el texto que acabas de añadir a la página de inicio.