

# Deploy Your Website on Cloud Run

**GSP659**



# Overview

Running websites can be difficult with all of the overhead of creating and managing VMs, clusters, pods, services, etc. This is fine for larger, multi-tiered applications, but if you are just trying to get your website deployed and visible, it's a lot of overhead.

With [Cloud Run](#), Google Cloud's implementation of [Google's KNative framework](#), you can manage and deploy your website without any of the infrastructure overhead you experience with a VM or pure Kubernetes-based deployments. Not only is this a simpler approach from a management perspective, it also gives you the ability to "scale to zero" when there are no requests coming into your website.

Cloud Run brings "serverless" development to containers and can be run either on your own Google Kubernetes Engine (GKE) clusters or on a fully managed PaaS solution provided by Cloud Run. You will be running the latter scenario in this lab.

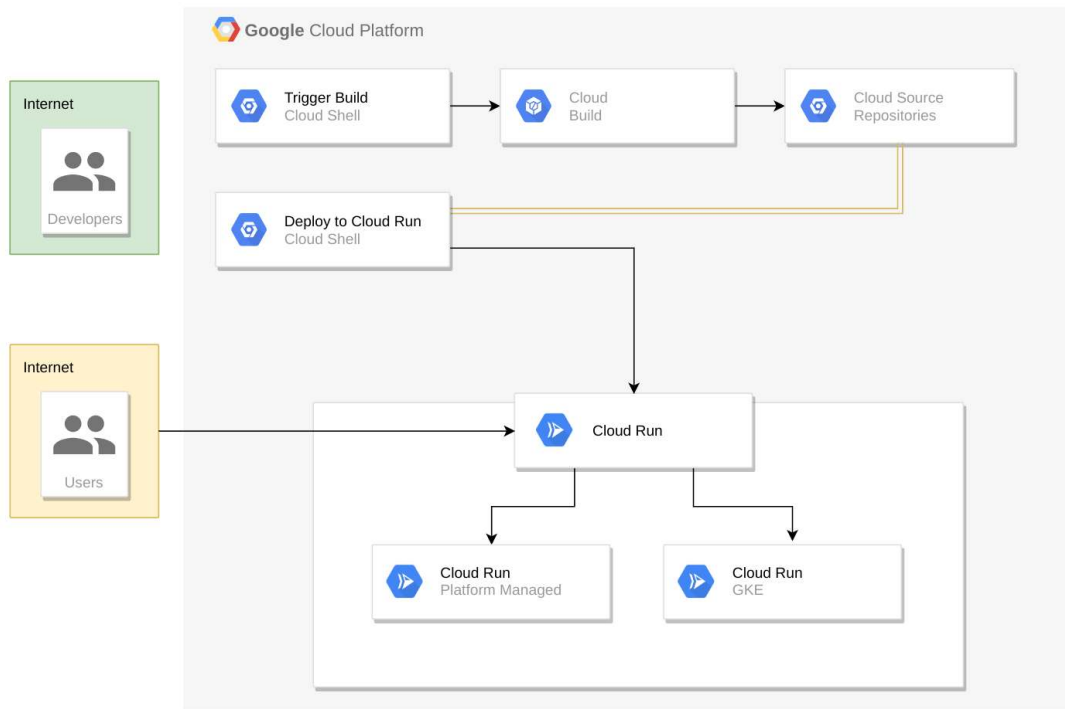
The exercises are ordered to reflect a common cloud developer experience:

1. Create a Docker container from your application
2. Deploy the container to Cloud Run
3. Modify the website
4. Roll out a new version with zero downtime

## Architecture diagram

Below you can see the flow of the deployment and Cloud Run hosting.

Begin with a Docker image created via Cloud Build, which gets triggered from Cloud Shell, then deploy the image out to Cloud Run from a command in Cloud Shell.



## What you'll learn

- How to build a Docker image using Cloud Build and upload it to gcr.io
- How to deploy Docker images to Cloud Run
- How to manage Cloud Run deployments
- How to setup an endpoint for an application on Cloud Run

# Environment Setup

## Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

## What you need

To complete this lab, you need:

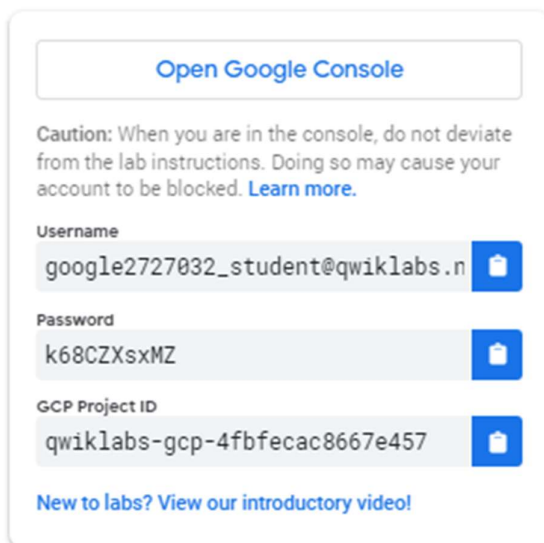
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

**Note:** If you already have your own personal Google Cloud account or project, do not use it for this lab.

**Note:** If you are using a Pixelbook, open an Incognito window to run this lab.

## How to start your lab and sign in to the Google Cloud Console

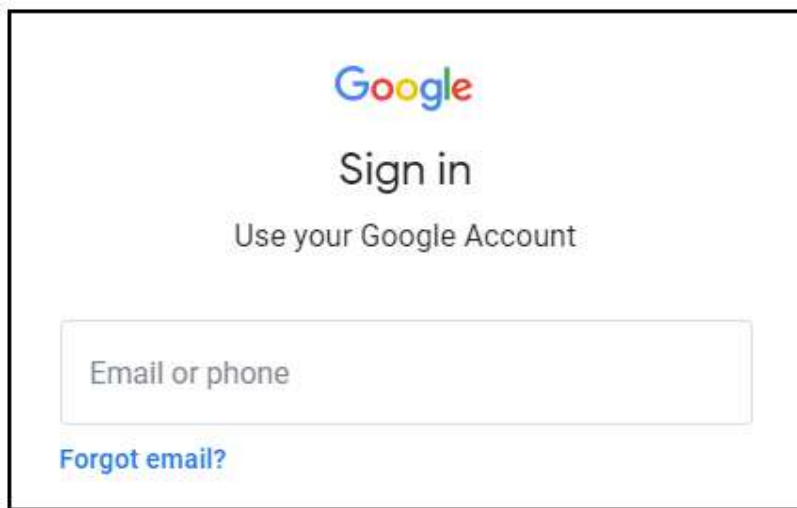
1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



The screenshot shows a sign-in panel with the following elements:

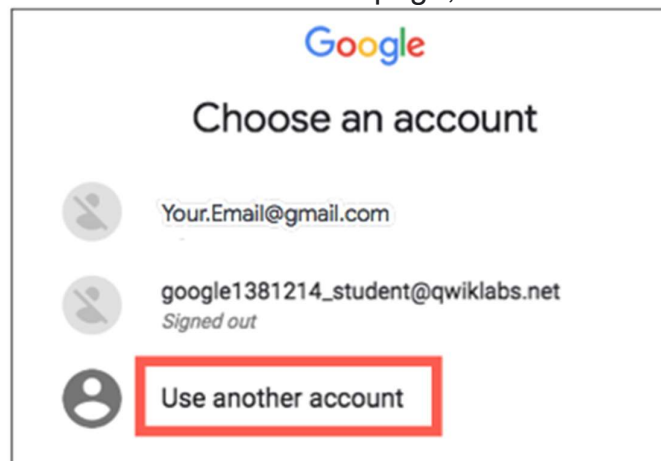
- A button at the top labeled "Open Google Console".
- A caution message: "Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)"
- Three input fields, each with a copy icon to its right:
  - Username:** google2727032\_student@qwiklabs.n
  - Password:** k68CZxsxMZ
  - GCP Project ID:** qwiklabs-gcp-4fbfecac8667e457
- A link at the bottom: "New to labs? View our introductory video!"

2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



**Tip:** Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another**



**Account.**

3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

**Important:** You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

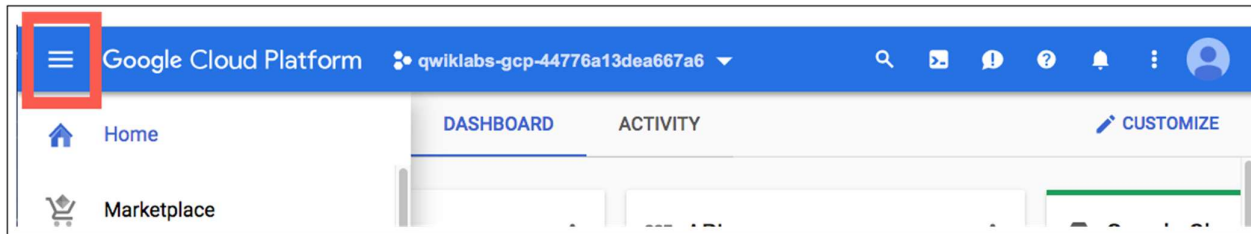
4. Click through the subsequent pages:

- Accept the terms and conditions.
- Do not add recovery options or two-factor authentication (because this is a temporary account).
- Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

**Note:** You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-

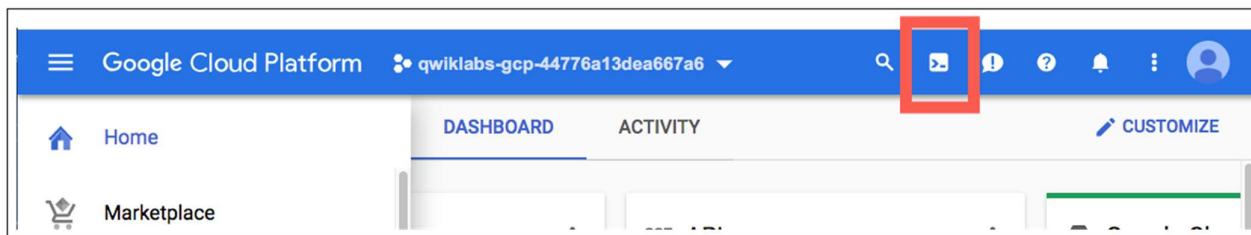
left.



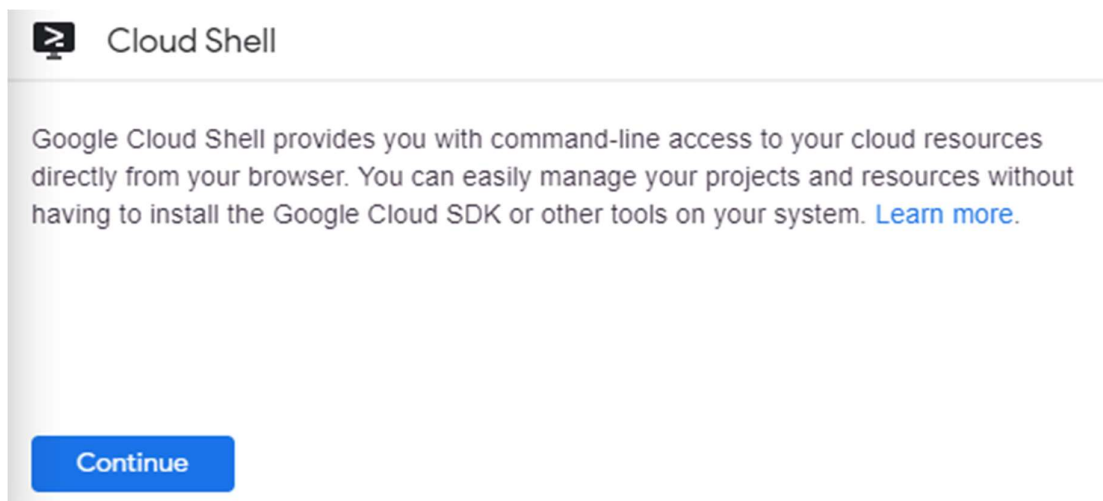
## Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

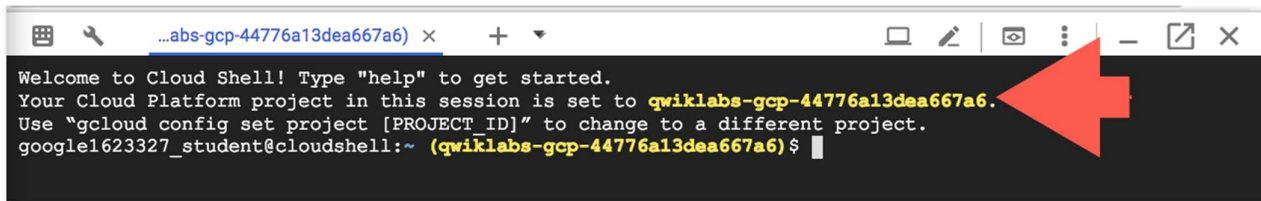
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT\_ID*. For example:



```
...abs-gcp-44776a13dea667a6) x + ▾  
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to qwiklabs-gcp-44776a13dea667a6.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
google1623327_student@cloudshell:~ (qwiklabs-gcp-44776a13dea667a6) $
```

`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

```
gcloud auth list
```

(Output)

```
Credentialed accounts:  
- <myaccount>@<mydomain>.com (active)
```

(Example output)

```
Credentialed accounts:  
- google1623327_student@qwiklabs.net
```

You can list the project ID with this command:

```
gcloud config list project
```

(Output)

```
[core]  
project = <project ID>
```

(Example output)

```
[core]  
project = qwiklabs-gcp-44776a13dea667a6
```

For full documentation of `gcloud` see the [gcloud command-line tool overview](#).

# Clone Source Repository

Since you are deploying an existing website, you just need to clone the source, so you can focus on creating Docker images and deploying to Cloud Run.

Run the following commands to clone the git repo to your Cloud Shell instance and change to the appropriate directory. You will also install the NodeJS dependencies so you can test the application before deploying:

```
git clone https://github.com/googlecodelabs/monolith-to-microservices.git
cd ~/monolith-to-microservices
./setup.sh
```

This will take a few minutes to run. You will see a success message when it finishes.

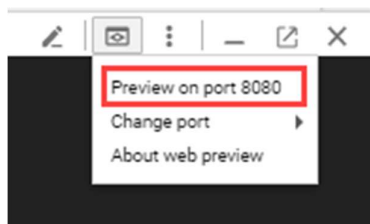
Test your application by running the following command to start the web server:

```
cd ~/monolith-to-microservices/monolith
npm start
```

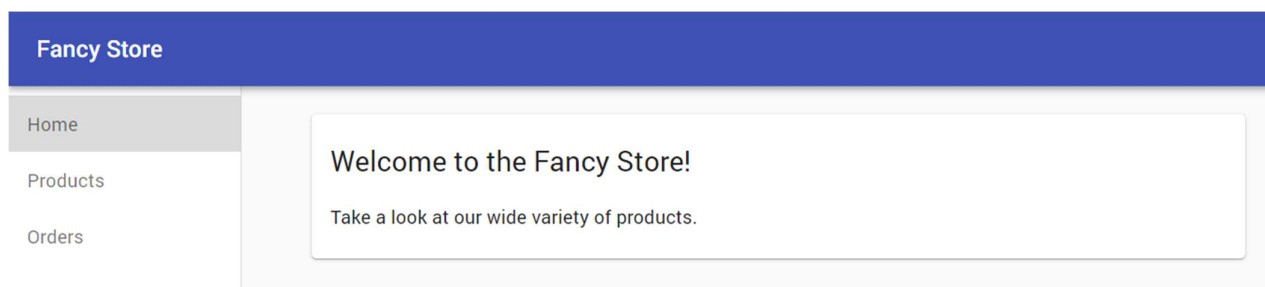
Output:

```
Monolith listening on port 8080!
```

Preview your application by clicking the web preview icon and selecting **Preview on port 8080**.



This should open a new window where you can see your Fancy Store web page in action.



Close this window after viewing the website, and stop the web server process by pressing `CTRL+C` in Cloud Shell.



# Create Docker Container with Cloud Build

Now that you have the source files ready to go, it is time to Dockerize your application!

Normally you would have to take a two step approach that entails building a docker container and pushing it to a registry to store the image for GKE to pull from. Make life easier and use Cloud Build to build the Docker container and put the image in Container Registry with a single command! To view the manual process of creating a Docker file and pushing it, go [here](#).

Cloud Build will compress the files from the directory and move them to a Cloud Storage bucket. The build process will then take all the files from the bucket and use the Dockerfile, which is present in the same directory, to run the Docker build process. Since you specified the `--tag` flag with the host as `gcr.io` for the Docker image, the resulting Docker image will be pushed to Container Registry.

First run the following command to enable the Cloud Build API:

```
gcloud services enable cloudbuild.googleapis.com
```

After the API is enabled, run the following command to start the build process:

```
gcloud builds submit --tag gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:1.0.0 .
```

This process will take a few minutes, but after it is completed, there will be output in the terminal similar to the following:

```
-----
ID                                     CREATE_TIME                               DURATION  SOURCE
IMAGES                               STATUS
1ae295d9-63cb-482c-959b-bc52e9644d53 2019-08-29T01:56:35+00:00 33S
gs://<PROJECT_ID>_cloudbuild/source/1567043793.94-abfd382011724422bf49af1558b894aa.tgz
gcr.io/<PROJECT_ID>/monolith:1.0.0 SUCCESS
```

To view your build history, or watch the process in real time, in the Console, from the **Navigation menu** click **Cloud Build > History**. Here you can see a list of all your builds; there should only be 1 that you just created.

Build History						
<input type="text" value="Filter builds"/>						
Build	Source	Git commit	Trigger	Started	Image target	Image tag
985ce3e5-afba...	gs://gcb-docs-project_cloudbuild/source/1508159187.8-b0d8841d51674a30aebd1e55bb99486f.tgz	-	-	9:06 AM	gcr.io/gcb-docs-project/quickstart-image	latest
d2e89f1b-cbf9...	gs://gcb-docs-project_cloudbuild/source/1508158566.55-725755714baa4b7e9e99984c422ec4e2.tgz	-	-	8:56 AM	gcr.io/gcb-docs-project/quickstart-image	latest

If you click on the Build ID, you can see all the details for that build including the log output.

From the Build Details page you can view the container image that was created by clicking the **Execution Details** tab, then clicking on on the image link.

[←](#) Build details [REBUILD](#) [COPY URL](#)

✓ Successful: 43acc906

Started on Feb 20, 2020, 3:00:50 PM

Source

<gs://qwiklabs-gcp-03-b1646832ec74.cloudbuild/source/1582228848.53-3592e66f101e4d67a6c16871ac58a585.tgz>

Steps	Duration	BUILD LOG	EXECUTION DETAILS	BUILD ARTIFACTS
✓ Build Summary	00:00:22			
1 Step				
0: gcr.io/cloud-builders/docker	00:00:10			
build --network cloudbuild --no-cache -t gcr.io/qwiklabs-gcp-0...				

Build id	43acc906-a7ee-4c7d-a8b6-a1cda987f874
Status	Successful
Timing	
Created	February 20, 2020 at 3:00:49 PM GMT-5
Started	February 20, 2020 at 3:00:50 PM GMT-5
Finished	February 20, 2020 at 3:01:13 PM GMT-5
Queued time	1 sec
Total build time	22 sec
Fetch source	4 sec
Build step(s)	11 sec
Push	5 sec
Source	<a href="gs://qwiklabs-gcp-03-b1646832ec74.cloudbuild/source/1582228848.53-3592e66f101e4d67a6c16871ac58a585.tgz">gs://qwiklabs-gcp-03-b1646832ec74.cloudbuild/source/1582228848.53-3592e66f101e4d67a6c16871ac58a585.tgz</a>
Image	<a href="gcr.io/qwiklabs-gcp-03-b1646832ec74/monolith:1.0.0">gcr.io/qwiklabs-gcp-03-b1646832ec74/monolith:1.0.0</a>

Click *Check my progress* to verify the objective.

# Deploy Container To Cloud Run

Now that you have containerized your website and pushed the container to Container Registry, it is time to deploy to Cloud Run!

There are two approaches for deploying to Cloud Run:

- **Managed Cloud Run:** The Platform as a Service model where all container lifecycle is managed by the Cloud Run product itself. You'll be using this approach in this lab.
- **Cloud Run on GKE:** Cloud Run with an additional layer of control which allows you to bring your own clusters & pods from GKE. [You can read more about it here.](#)

## Command-Line

You will deploy the image that was built earlier, and choose the managed version of Cloud Run by specifying `--platform managed`.

First you need to enable the Cloud Run API. Run the following command to enable it:

```
gcloud services enable run.googleapis.com
```

Run the following command to deploy your application:

```
gcloud run deploy --image=gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:1.0.0 --platform managed
```

Specify which region you'd like to run in. Type the number for the region closest to you.

```
Please specify a region:
[1] asia-northeast1
[2] europe-west1
[3] us-central1
[4] us-east1
[5] cancel
Please enter your numeric choice:
```

Accept the default suggested service name (it will be "monolith") by pressing **Enter**.

For this lab, allow unauthenticated requests into the application. Type "Y" at the prompt:

```
Allow unauthenticated invocations to [monolith] (y/N)? y
```

Click *Check my progress* to verify the objective.

## Verify deployment

To verify the deployment was created successfully, run the following command:

```
gcloud run services list
```

It may take a few moments for the pod status to be Running.

Type "1" to choose the first option: [1] Cloud Run (fully managed)

Output:

```
SERVICE    REGION    URL    LAST DEPLOYED BY    LAST DEPLOYED AT
✓ monolith  us-east1  <your url>  <your email>  2019-09-16T21:07:38.267Z
```

This output shows several things. You can see the deployment, as well as the user that deployed it (your email) and the URL you can use to access the app. Looks like everything was created successfully!

Click on the URL provided in the list of services. You should see the same website you previewed locally.

**Helpful Hint:** You can also view your Cloud Run deployments via the Console if you navigate to **Cloud Run** in the **Navigation menu**.

# Create new revision with lower concurrency

Deploy your application again, but this time adjust one of the parameters.

By default, a Cloud Run application will have a concurrency value of 80, meaning that each container instance will serve up to 80 requests at a time. This is a big departure from the Functions-as-a-Service model, where one instance handles one request at a time.

Re-deploy the same container image with a concurrency value of 1 (just for testing), and see what happens:

```
gcloud run deploy --image=gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:1.0.0 --platform managed --concurrency 1
```

Answer the subsequent questions just as you did the first time. Once the command is successful, check the Console to see the result.

To see the details, from the **Navigation menu**, click on **\_Cloud Run**, then click on the **monolith** service :

Each Cloud Run service has a unique endpoint and autoscales deployed containers. [Learn more](#)

Filter services

<input type="checkbox"/>	<input checked="" type="radio"/>	Name <span>↑</span>	Req/sec <span>?</span>	Location	Authentication <span>?</span>	Connectivity <span>?</span>
<input type="checkbox"/>	<input checked="" type="radio"/>	monolith	0	us-east1	Allow unauthenticated	External

On the Service Details page, click on the **Revisions** tab. You should now see 2 revisions created.

The most recent deployment has Details on the right hand side.

Cloud Run

Service details

DEPLOY NEW REVISION

monolith

Region: us-east1

URL: <https://monolith-k3wa45c54a-ue.a.run.app>

METRICS

REVISIONS

LOGS

DETAILS

YAML

PERMISSIONS

Revisions 

MANAGE TRAFFIC

Filter revisions

<input checked="" type="radio"/>	Name	Traffic	Deployed	Actions
<input checked="" type="radio"/>	monolith-00002-rip	100%	3 minutes ago	<div></div>
<input type="radio"/>	monolith-00001-kec	0%	7 minutes ago	<div></div>

You will see that the concurrency value has been reduced to "1".

DETAILS	
YAML	
Container image URL	<a href="https://gcr.io/qwiklabs-gcp-03-b1646832ec74/monolith@sha256-1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef">gcr.io/qwiklabs-gcp-03-b1646832ec74/monolith@sha...</a>
Container port	8080
Container command and args	(container entrypoint)
Autoscaling	Up to 1,000 container instances
CPU allocated	1
Memory allocated	256M
Concurrency	1
Request timeout	900 seconds
Service account	Default Compute service account

Although this configuration is sufficient for testing, in most production scenarios you will have containers supporting multiple concurrent requests.

Click *Check my progress* to verify the objective.

### Create new revision with lower concurrency

[Check my progress](#)

Next, restore the original concurrency without re-deploying.

You could set the concurrency value back to the default of "80", or you could just set the value to "0", which will remove any concurrency restrictions and set it to the default max (which happens to be 80).

Run the following command from Cloud Shell to update the current revision:

```
gcloud run deploy --image=gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:1.0.0 --platform
managed --concurrency 80
```

Answer the subsequent questions just as you have before.

You will notice that another revision has been created, that traffic has now been redirected, and that the concurrency is back up to 80.

You may need to leave the **Revisions** tab and then return to it to see the most up to date information.

# Make Changes To The Website

**Scenario:** Your marketing team has asked you to change the homepage for your site. They think it should be more informative of who your company is and what you actually sell.

**Task:** You will add some text to the homepage to make the marketing team happy! It looks like one of our developers already created the changes with the file name `index.js.new`. You can just copy this file to `index.js` and your changes should be reflected. Follow the instructions below to make the appropriate changes.

Run the following commands to copy the updated file to the correct file name and then print its contents to verify the changes:

```
cd ~/monolith-to-microservices/react-app/src/pages/Home
mv index.js.new index.js
cat ~/monolith-to-microservices/react-app/src/pages/Home/index.js
```

The resulting code should look like this:

```
/*
Copyright 2019 Google LLC

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    https://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
*/

import React from "react";
import { makeStyles } from "@material-ui/core/styles";
import Paper from "@material-ui/core/Paper";
import Typography from "@material-ui/core/Typography";
const useStyles = makeStyles(theme => ({
  root: {
    flexGrow: 1
  },
  paper: {
    width: "800px",
    margin: "0 auto",
    padding: theme.spacing(3, 2)
  }
}));
export default function Home() {
  const classes = useStyles();
  return (
    <div className={classes.root}>
      <Paper className={classes.paper}>
        <Typography variant="h5">
          Fancy Fashion & Style Online
        </Typography>
        <br />
        <Typography variant="body1">
```

```

        Tired of mainstream fashion ideas, popular trends and societal norms?
        This line of lifestyle products will help you catch up with the Fancy trend
and express your personal style.
        Start shopping Fancy items now!
    </Typography>
  </Paper>
</div>
);
}

```

You updated the React components, but you need to build the React app to generate the static files.

Run the following command to build the React app and copy it into the monolith public directory:

```

cd ~/monolith-to-microservices/react-app
npm run build:monolith

```

Now that the code is updated, rebuild the Docker container and publish it to Container Registry. You can use the same command as before, except this time you will update the version label.

Run the following command to trigger a new Cloud Build with an updated image version of 2.0.0:

```

cd ~/monolith-to-microservices/monolith

```

Optional

```

#Feel free to test your application
npm start
gcloud builds submit --tag gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:2.0.0 .

```

In the next section you will use this image to update your application with zero downtime.

Click *Check my progress* to verify the objective.

Make Changes To The Website

Check my progress



# Update website with zero downtime

The changes are complete and the marketing team is happy with your updates! It is time to update the website without interruption to the users.

Cloud Run treats each deployment as a new *Revision* which will first be brought online, then have traffic redirected to it. By default the latest revision will be assigned 100% of the inbound traffic for a service. It is possible to use "Routes" to allocate different percentages of traffic to different revisions within a service.

Follow the instructions below to update your website.

From Cloud Shell, re-deploy the service to update the image to a new version with the following command:

```
gcloud run deploy --image=gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:2.0.0 --platform managed
```

Answer the subsequent questions just as you have before.

Click *Check my progress* to verify the objective.

## Verify Deployment

Validate that your deployment updated by running the following command:

```
gcloud run services describe monolith --platform managed
```

Type in the number for the region you've been using.

Output:

```
✓ Service monolith in region us-east1
https://monolith-k3wa45c54a-ue.a.run.app

Traffic:
  100%                LATEST (currently monolith-00003-ten)

Last updated on 2020-02-20T20:26:56.049Z by student-03-e32055b0be3a@qwiklabs.net:
Revision monolith-00003-ten
Image:          gcr.io/qwiklabs-gcp-03-b1646832ec74/monolith:2.0.0
Port:           8080
Memory:         256M
CPU:            1000m
Concurrency:    1
Timeout:        900s
```

Here you will see that the Service is now using the latest version of your image, deployed in a new revision.

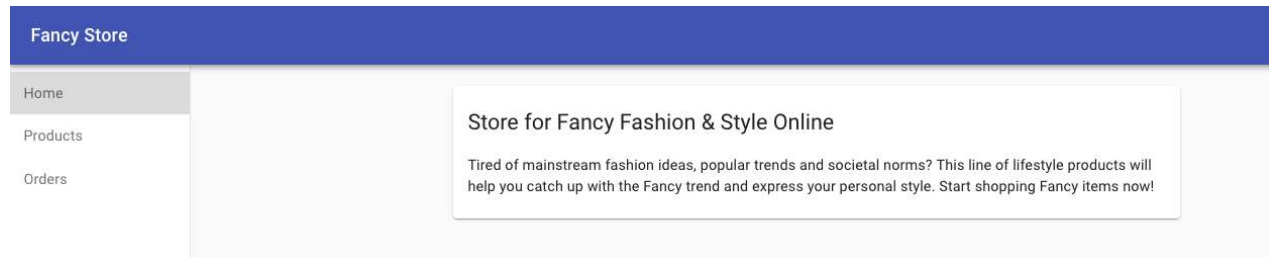
To verify the changes, navigate to the external URL of the Cloud Run service, refresh the page, and notice that the application title has been updated.

Run the following command to list the services and view the IP address:

```
gcloud beta run services list
```

Choose the first option, **Cloud Run (fully managed)** and press **Enter**.

Click on the URL of the service. Your web site should now be displaying the text you just added to the homepage component!



# Cleanup

When you end this lab all of the resources you used will be destroyed. It's good to know how to delete resources so you can do so in your own environment.

Run the following to delete Container Registry images:

```
# Delete the container image for version 1.0.0 of our monolith
gcloud container images delete gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:1.0.0 --quiet

# Delete the container image for version 2.0.0 of our monolith
gcloud container images delete gcr.io/${GOOGLE_CLOUD_PROJECT}/monolith:2.0.0 --quiet
```

Run the following to delete Cloud Build artifacts from Cloud Storage:

```
# The following command will take all source archives from all builds and delete them
from cloud storage

# Run this command to print all sources:
# gcloud builds list | awk 'NR > 1 {print $4}'

gcloud builds list | awk 'NR > 1 {print $4}' | while read line; do gsutil rm $line;
done
```

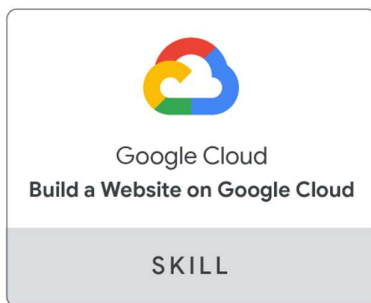
Finally, delete Cloud Run service:

```
gcloud beta run services delete monolith --platform managed
```

Choose the region where you've been running the service for this lab, then confirm that you want `monolith` deleted.

# Congratulations!

You successfully deployed, de-scaled, re-scaled, and updated your website on Cloud Run.



## Finish Your Quest

This self-paced lab is part of the Qwiklabs [Deploying Applications](#), [Cloud SQL](#), [Websites and Web Apps](#) and [Website on Google Cloud](#) Quests. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in a Quest and get immediate completion credit if you've taken this lab. [See other available Qwiklabs Quests](#).

## Take Your Next Lab

Continue your learning with [Build and Launch an ASP.NET Core App from Google Cloud Shell](#) or check out these suggestions:

- [Deploy ASP.NET Core App to App Engine](#)
- [Deploy Refinery CMS to App Engine Flexible Environment](#)

## Next Steps / Learn More

If you're not familiar with these products, here are links to learn more:

- [Docker](#), [Docker's "Getting Started" page](#)
- Kubernetes Engine - <https://cloud.google.com/kubernetes-engine/docs/>
- Cloud Build - <https://cloud.google.com/cloud-build/docs/>
- Container Registry - <https://cloud.google.com/container-registry/docs/>

## End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Manual Last Updated: May 26, 2020

Lab Last Tested: February 20, 2020

Copyright 2021 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.