

APIs Explorer: App Engine

GSP422



Overview

Google APIs Explorer is a tool that lets you try out various Google APIs interactively. With the APIs Explorer, you can:

- Browse quickly through available APIs and versions.
- See methods available for each API and what parameters they support along with inline documentation.
- Execute requests for any method and see responses in real-time.
- Make authenticated and authorized API calls.
- Search across all services, methods, and your recent requests to quickly find what you are looking for.

[App Engine](#) lets you deploy applications on a fully managed platform. You can scale your applications seamlessly without having to worry about managing the underlying infrastructure. With zero server management and zero configuration deployments, developers can focus only on building great applications without the management overhead.

In this lab you will deploy a simple hello world application to App Engine and make updates to its configuration using the App Engine Admin API through the APIs Explorer tool.

Objectives

In this lab, you will:

- Build an App Engine application with the APIs Explorer tool.
- Deploy an App Engine instance from the hello world sample code.
- Configure App Engine firewall rules with the APIs Explorer tool.
- Make changes to your code base and create a new version of your application with the APIs Explorer tool.

Prerequisites

This is a **fundamental level** lab. You should be familiar with the basic functioning and architecture of APIs. Experience with Google Cloud Shell and command-line interface tools is recommended.

Familiarity with the APIs Explorer tool is also recommended. At a minimum, take the following labs before attempting this one:

- [Introduction to APIs in Google](#)
- [APIs Explorer: Qwik Start](#)

If you are unfamiliar with App Engine, the [App Engine: Qwik Start - Python](#) lab has valuable information that will orient you with the content of this lab. Once you're ready, scroll down and follow the steps below to set up your lab environment.

Setup and Requirements

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:

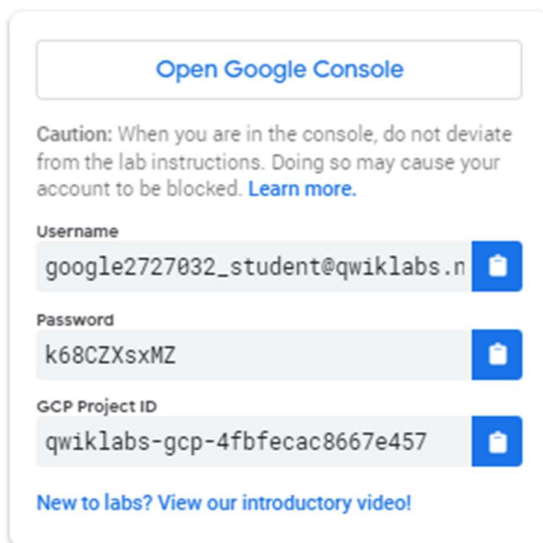
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

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 panel with the following content:

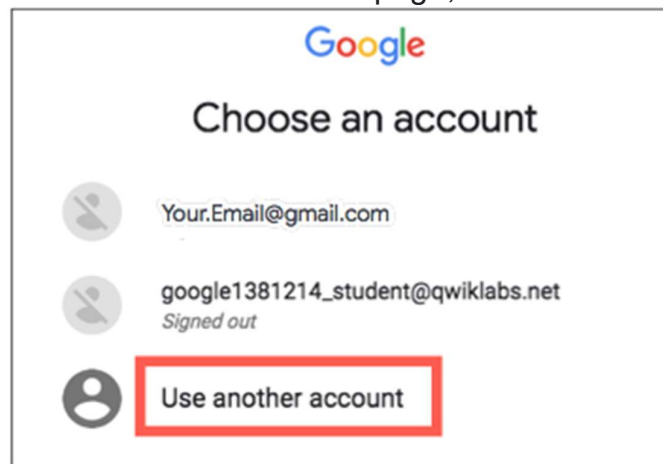
- A button at the top labeled "Open Google Console".
- A caution message: "Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)"
- Three input fields, each with a copy icon to its right:
 - Username: google2727032_student@qwiklabs.n
 - 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).

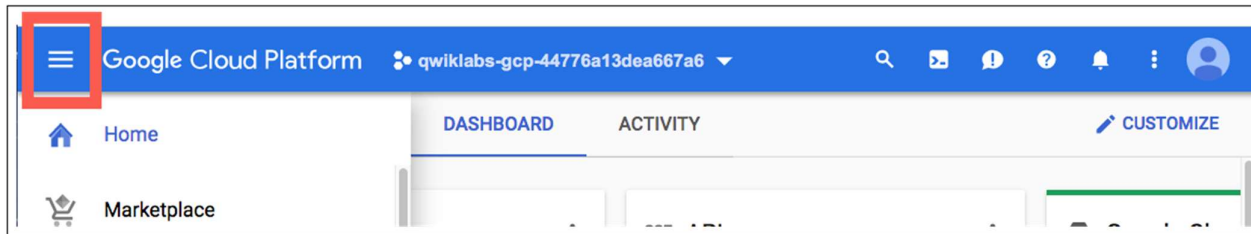
4. Click through the subsequent pages:

- 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-

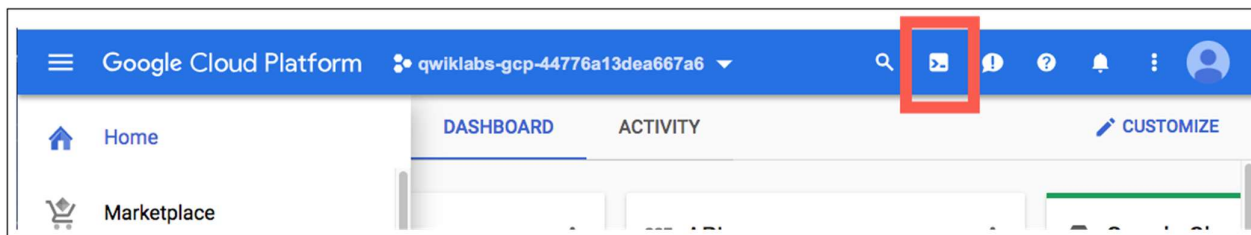
left.



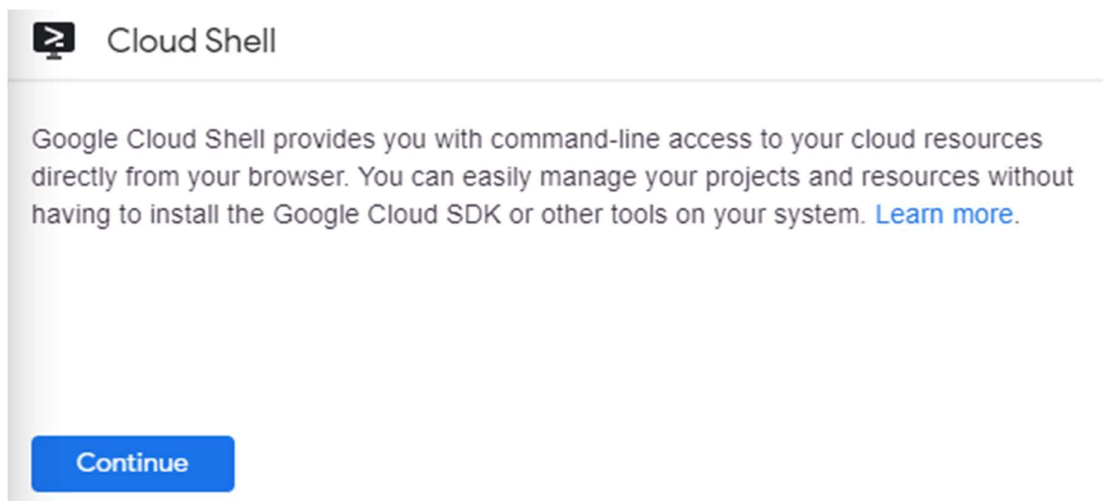
Activate Cloud Shell

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.

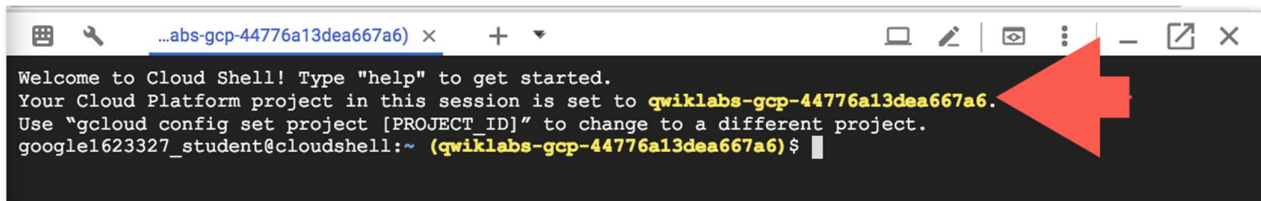
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



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](#).

Build an App Engine application with apps.create

You will now build an App Engine application with one of the methods found in the APIs Explorer.

To access the App Engine APIs Explorer tool, open up the navigation menu and select **APIs & Services > Library**.

In the search bar, enter in **App Engine** and select the **App Engine Admin API** from the results list. Make sure that API is enabled, if not click **Enable**.

Now that you have verified the API's enablement, open [this link](#) This will take you to the apps **create** method.

Under **Try this API** in the right panel, click in the **Request body** field and add:

- the **ID** property. Set it's value to your Project ID.
 - the **locationId** property. Set its value to **us-west2**. This required field tells Google Cloud where your App Engine resources will live.
- Make sure that there are no trailing spaces in any of the fields. Also, that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.

Credentials ?

☒ Google OAuth 2.0
OAuth 2.0 provides authenticated access to an API. [Show scopes](#) ▾

☒ API key
An API key is a unique string that lets you access an API.

Note: To view **Credentials FAQs**, click on question mark icon next to **Credentials** title. Click the **Execute** button.

Select the student account you started the lab with.

On the next screen, click **Allow** to give APIs Explorer access.

Your response should resemble the following:

```
{
  "name": "apps/qwiklabs-gcp-da84962e277c92a7/operations/193f576c-8791-4638-920e-b1ccb6305ae1",
  "metadata": {
    "@type": "type.googleapis.com/google.appengine.v1.OperationMetadataV1",
    "method": "google.appengine.v1.Applications.CreateApplication",
    "insertTime": "2019-10-16T12:37:36.743Z",
    "user": "gcpstaging92860_student@qwiklabs.net",
```

```
}
  "target": "apps/qwiklabs-gcp-da84962e277c92a7"
}
```

You have successfully built an App Engine application for a Google Cloud project.

Get application information with apps.get

Next you'll retrieve some information on your App Engine application to ensure that it has been properly created.

From the left APIs & Reference section navigate to **REST API > v1 > apps > get**. Or you can use [this direct link](#) to `apps.get` method.

For the **appId** field, enter `<PROJECT_ID>`. Make sure to replace `<PROJECT_ID>` with your lab Project ID.

Make sure that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.

Credentials ?

☒ Google OAuth 2.0

OAuth 2.0 provides authenticated access to an API. [Show scopes](#) ▾

☒ API key

An API key is a unique string that lets you access an API.

Click the **Execute** button. You may need to select the student account and click **Allow** again.

Your response should resemble the following:

```
{
  "name": "apps/qwiklabs-gcp-da84962e277c92a7",
  "id": "qwiklabs-gcp-da84962e277c92a7",
  "authDomain": "gmail.com",
  "locationId": "us-west2",
  "codeBucket": "staging.qwiklabs-gcp-da84962e277c92a7.appspot.com",
  "servingStatus": "SERVING",
  "defaultHostname": "qwiklabs-gcp-da84962e277c92a7.appspot.com",
  "defaultBucket": "qwiklabs-gcp-da84962e277c92a7.appspot.com",
  "gcrDomain": "us.gcr.io",
  "featureSettings": {
    "splitHealthChecks": true,
  }
}
```



```
"useContainerOptimizedOs": true
}
```

This method works as a sanity check and offers you useful information about your App Engine application, such as its default hostname, location, and serving status.

Download the starter code

Before you *deploy* an App Engine application, you will need to download some starter code so you have something to work with.

Return to the Cloud Console and in Cloud Shell run the following command to clone a repository that contains the codebase for a simple hello world application:

```
git clone https://github.com/GoogleCloudPlatform/python-docs-samples
```

Now change your current working directory:

```
cd ~/python-docs-samples/appengine/standard_python3/hello_world
```

The `hello_world` folder contains a simple Python application that uses the [Flask](#) web framework. This Python app responds to a request with an HTTP header and the message "Hello World!"

Deploy your App Engine application

For this step, remain in your Cloud Shell session. Run the following command to set your Project ID as an environment variable, replacing `[YOUR_PROJECT_ID]` with your Qwiklabs Project ID:

```
export PROJECT_ID=[YOUR_PROJECT_ID]
```

Now run the following gcloud command to deploy your hello world application:

```
gcloud app deploy app.yaml --project $PROJECT_ID
```

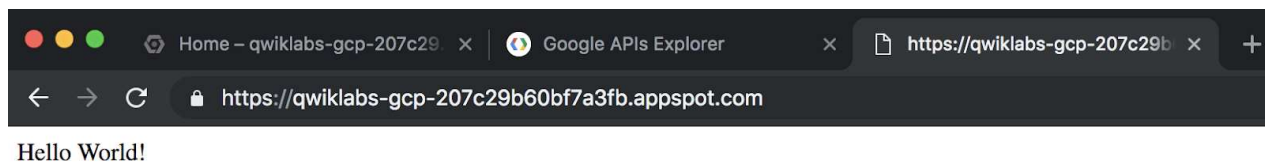
When prompted with the following, enter in **Y**:

```
Do you want to continue (Y/n)?
```

The deployment will take a couple minutes to complete. Once it has finished, you should receive a similar output:

```
File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://qwiklabs-gcp-b5d5fa242d334941.appspot.com]
```

Copy the deployed service link that resembles `https://qwiklabs-gcp-b5d5fa242d334941.appspot.com` and paste it in a new tab. This will open the hello world application. Your page should resemble the following:



Now that your application is deployed, you will make some changes to your App Engine configuration with the APIs Explorer. **Keep the hello world page open.**

Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully deployed an app engine application, you will see an assessment score.

Configure ingress firewall rules with `apps.firewall.ingressRules`

You will now create, list, and delete firewall rules that prescribe access to your hello world application.

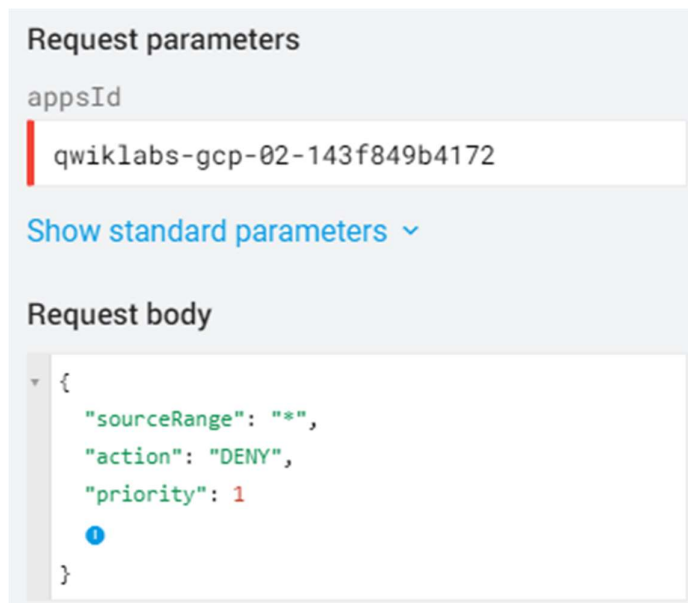
Create an ingress firewall rule

From the left APIs & Reference section navigate to **REST API > v1 > apps.firewall.ingressRules > create**. Or you can use [this direct link](#) to `apps.firewall.ingressRules.create` method. For the **appId** field, enter your Project ID.

Now click on the **Request body** and add:

- the **sourceRange** property. Set its value to `*`.
- the **action** property. Set its value to **deny**.
- the **priority** property and set its value to **1**.

Your method should resemble the following:



The screenshot shows a REST client interface. Under the 'Request parameters' section, the 'appId' field is filled with 'qwiklabs-gcp-02-143f849b4172'. Below this is a 'Show standard parameters' link with a dropdown arrow. The 'Request body' section is expanded, showing a JSON object:

```
{  "sourceRange": "*",  "action": "DENY",  "priority": 1}
```

Verify that your fields have no trailing spaces. Also, that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.

Credentials ?

☒ Google OAuth 2.0

OAuth 2.0 provides authenticated access to an API. [Show scopes](#) ▾

☒ API key

An API key is a unique string that lets you access an API.

Click the **Execute** button. Your response should resemble the following:

```
{
  "priority": 1,
  "action": "DENY",
  "sourceRange": "*"
}
```

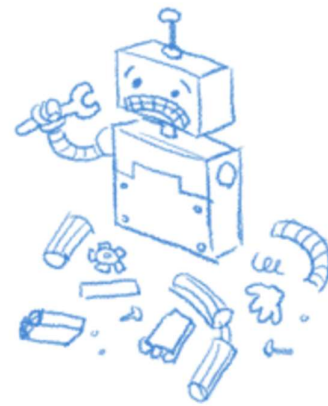
This firewall rule rejects all requests to your hello world application.

To see it in action, **Refresh** your hello world page in your browser. You should now see that access is now forbidden:



403. That's an error.

Access is forbidden. That's all we know.



Return to the APIs Explorer page for the next step.

List ingress firewall rules

From the left APIs & Reference section navigate to **REST API > v1 > apps.firewall.ingressRules > list** Or you can use [this direct link](#) to `apps.firewall.ingressRules.list` method. For the **projectId** field, enter your Project ID.

Request parameters

appsId

qwiklabs-gcp-01-a2b60d009631

matchingAddress

string

pageSize

integer

pageToken

string

Verify that the appsId field has no trailing spaces. Also, that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.

Credentials ?

☒ Google OAuth 2.0

OAuth 2.0 provides authenticated access to an API. [Show scopes](#) ▾

☒ API key

An API key is a unique string that lets you access an API.

Click the **Execute** button.

Your response should resemble the following:

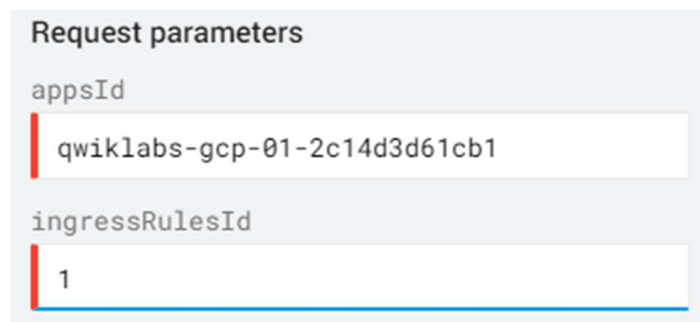
```
{
  "ingressRules": [
    {
      "priority": 1,
      "action": "DENY",
      "sourceRange": "*"
    },
    {
      "priority": 2147483647,
      "action": "ALLOW",
      "sourceRange": "*",
      "description": "The default action."
    }
  ]
}
```

You now see the two firewall rules: one that allows traffic and another that denies traffic to your application. Note the *priority* values for each ingress rule — these act as firewall rule IDs as well.

Delete an ingress firewall rule

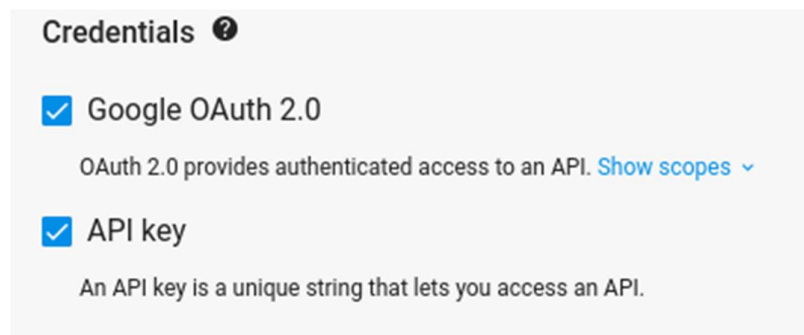
From the left APIs & Reference section navigate to **REST API > v1 > apps.firewall.ingressRules > delete** Or you can use [this direct link](#) to `apps.firewall.ingressRules.delete` method.

For the **appId** field, enter your Project ID. For the **ingressRulesId** field, enter **1**. Your method should resemble the following:



The screenshot shows the 'Request parameters' section of a REST API client. It contains two input fields. The first field is labeled 'appId' and contains the text 'qwiklabs-gcp-01-2c14d3d61cb1'. The second field is labeled 'ingressRulesId' and contains the text '1'.

Verify that your fields have no trailing spaces. Also, that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.



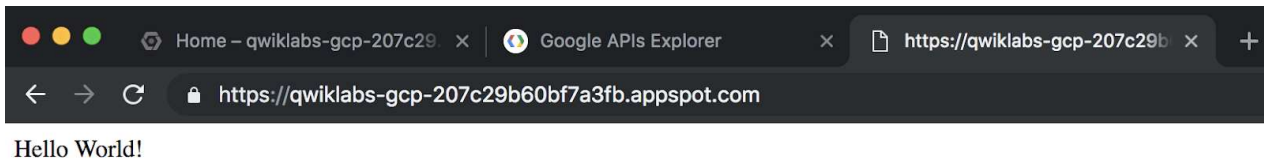
The screenshot shows the 'Credentials' section of the REST API client. It has a title 'Credentials' with a help icon. Below the title are two checked checkboxes. The first is 'Google OAuth 2.0' with a subtext 'OAuth 2.0 provides authenticated access to an API. [Show scopes](#)'. The second is 'API key' with a subtext 'An API key is a unique string that lets you access an API.'

Click the **Execute** button.

Your response should resemble the following:

```
{}
```

Now refresh your hello world page in your browser. You should now see that access been restored:



Now that you've gotten practice with ingress firewall rule configuration, take it to the next level by creating and deploying new versions of our application.

Update your application files

Now make a small change to your application's source code. For this step, return to the Cloud Shell. You should still be in the `hello_world` directory. If not, run the following command:

```
cd ~/python-docs-samples/appengine/standard_python3/hello_world
```

Now open the `main.py` file with the `nano` text editor:

```
nano main.py
```

Scroll down to the `hello` function and edit it so it says "Goodbye world!" instead:

```
@app.route('/')
def hello():
    """Return a friendly HTTP greeting."""
    return 'Goodbye World!'
```


Press **CNTRL +x** then **Y > ENTER** to save your changes and exit the `nano` editor.


Create a new version of your application with `apps.services.versions.create`


You will now create a new version of your application that uses your updated "Goodbye world!" codebase.


In the Cloud Console, from the **Navigation menu** select **Storage > Browser**. You should see a list of buckets that resemble the following:

Browser

 CREATE BUCKET





 REFRESH

 DELETE

 Filter by prefix...

Columns ▾

Buckets

<input type="checkbox"/>	Name	Default storage class 	Location	Public access 	Lifecycle 	Labels 
<input type="checkbox"/>	qwiklabs-gcp-b930bc8cd8e78415.appspot.com	Regional	US-WEST2	Per object	None	
<input type="checkbox"/>	staging.qwiklabs-gcp-b930bc8cd8e78415.appspot.com	Regional	US-WEST2	Per object	Enabled	
<input type="checkbox"/>	us.artifacts.qwiklabs-gcp-b930bc8cd8e78415.appspot.com	Multi-Regional	US	Per object	<u>None</u>	

Copy the `staging.qwiklabs-gcp-xxxx.appspot.com` bucket name and save it.

Now click on that bucket to view the files it contains.

Copy the name of the `application/json` file and save it:














staging.qwiklabs-gcp-b930bc8cd8e78415.appspot.com

[Objects](#) [Overview](#) [Permissions](#) [Bucket Lock](#)

[Upload files](#) [Upload folder](#) [Create folder](#) [Manage holds](#) [Delete](#)

Filter by prefix...

[Buckets](#) / staging.qwiklabs-gcp-b930bc8cd8e78415.appspot.com

 Name	Size	Type	Storage class
  0b48b923b0d7a0323b162a4161a4a891cec1aa93	13 B	text/plain	Regional
  3eeca6c25374c369ebfe19d86579b1515011d3f3	136 B	application/json	Regional
  7824eeacd13c187cfe57b939d5f840e4134fed33	801 B	text/x-python	Regional
  9a5b7abfb336fa296b988b912b5732faa0de6ea1	1.15 KB	text/x-python	Regional
  a019edb1f4e2a68ad6f1e573bf4e02953fc3a11b	18 B	application/octet-stream	Regional
  ae/	—	Folder	—

You now have the information needed to create a new version of your hello world application. Return to the APIs Explorer for the next step.

From the left APIs & Reference section navigate to **REST**

API > v1 > apps.services.versions > create. Or you can use [this direct link](#) to `apps.services.versions.create` method.

For the **appId** field, enter your Project ID. For the **servicesId** field, enter **default**.

Now click in the request body and add:

- the **id** property. Set its value to **v1**.
- the **runtime** property and set its value to **python37**.
- Add the **entrypoint** property—inside, then add the **shell** property and keep its value empty.

Your method should now resemble the following:

Request parameters

appsId

qwiklabs-gcp-01-a2b60d009631

servicesId

default

[Show standard parameters](#) ▾

Request body

```
{
  "id": "v1",
  "runtime": "python37",
  "entrypoint": {
    "shell": ""
  }
}
```

For suggestions, press control+space or click one of the blue "add" circles.

- Now add the **deployment** property.
 - Inside deployment, add a new property **files** and click the **add** link that comes underneath it. Give it the name **latest**.
 - Add the **sourceUrl** property in latest and set it to the following, replacing `<YOUR_BUCKET_NAME>` with the name of the staging Cloud Storage bucket and `<YOUR_JSON_FILE_NAME>` with the name of the JSON file you copied over:
`https://storage.googleapis.com/<YOUR_BUCKET_NAME>/<YOUR_JSON_FILE_NAME>`
- Your method should now resemble the following:

Request body

```
{
  "id": "v1",
  "runtime": "python37",
  "entrypoint": {
    "shell": ""
  },
  "deployment": {
    "files": {
      "latest": {
        "sourceUrl": "https://storage.googleapis.com/..."
      }
    }
  }
}
```

For suggestions, press control+space or click one of the blue "add" circles.

Make sure that **Google OAuth 2.0** and **API key** checkboxes are selected under **Credentials** section.

Credentials ?

☒ Google OAuth 2.0

OAuth 2.0 provides authenticated access to an API. [Show scopes](#) ▾

☒ API key

An API key is a unique string that lets you access an API.

Once that's filled out, click the **Execute** button.
You should receive the following output:

```
{
  "name": "apps/qwiklabs-gcp-da84962e277c92a7/operations/7ca371a7-3bf6-4215-871e-7f9aac815714",
  "metadata": {
    "@type": "type.googleapis.com/google.appengine.v1.OperationMetadataV1",
    "method": "google.appengine.v1.Versions.CreateVersion",
    "insertTime": "2019-10-16T14:39:19.231Z",
    "user": "gcpstaging92860_student@qwiklabs.net",
    "target": "apps/qwiklabs-gcp-da84962e277c92a7/services/default/versions/v1"
  }
}
```

There were many fields to fill out, but that is where the APIs Explorer shines. Being able to visualize all the parameters and see how they relate to one another is critical in successfully calling API methods.

Deploy the new version of your application

Return to the Cloud Console for this step. Open the navigation menu and select **App Engine > Versions**.

You should see that there are now two versions of your application available:

Versions							
<div>REFRESH DELETE STOP START MIGRATE TRAFFIC</div>							
Service default							
Filter versions							
<input type="checkbox"/> Version	Status	Traffic Allocation	Instances ?	Runtime	Environment		
<input type="checkbox"/> v1	Serving	0%	0	python37	Standard		
<input type="checkbox"/> 20181207t142901	Serving	100%	1	python37	Standard		

If you don't see two versions, it's possible that your updates haven't propagated yet. Click the **Refresh** button in the top left corner until your page resembles the above screenshot.

Return to your Cloud Shell session. You should still be in the `hello_world` directory. If not, run the following command:

```
cd ~/python-docs-samples/appengine/standard_python3/hello_world
```

You will now deploy the new version of your application.

Run the following command to deploy the new version, with "Goodbye world!" as the message:

```
gcloud app deploy -v v1
```

When prompted with the following, enter in **Y**:

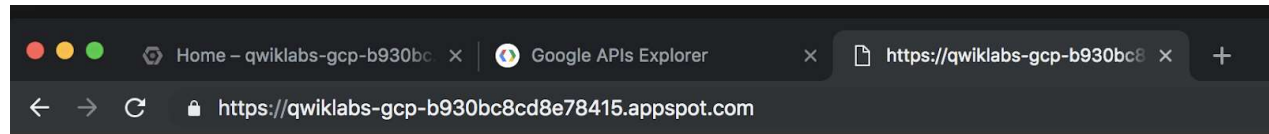
```
Do you want to continue (Y/n)?
```

The deployment will take a couple minutes to complete.

Once it has finished, you should receive a similar output:

```
File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://qwiklabs-gcp-b5d5fa242d334941.appspot.com]
```

Now copy the link or refresh the application page in your browser. You should see the following:



Goodbye World!

If you return to the Cloud Console and look at **App Engine > Versions** you will see that that v1 is being run:

Service **default** ▼

Filter versions							
<input type="checkbox"/> Version	Status	Traffic Allocation	Instances	Runtime	Environment		
<input type="checkbox"/> v1 ↗	Serving	<div><div></div></div> 100%	1	python37	Standard		
<input type="checkbox"/> 20181207t142901 ↗	Serving	<div><div></div></div> 0%	1	python37	Standard		

You have successfully created a new version of an application with the APIs Explorer and deployed it in Cloud Shell.

Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully created a new version of your app, you will see an assessment score.

Congratulations!

In this lab, you got hands-on practice with App Engine Admin API methods using the APIs Explorer. After building an App Engine application with the APIs Explorer tool, you deployed an instance from the hello world sample code. You then learned how to configure ingress firewall rules with the APIs Explorer tool. After making changes to the codebase, you used the APIs Explorer to create a new version of your application, which you deployed and successfully accessed. You are now ready to take more Exploring APIs labs.

Finish Your Quest



This self-paced lab is part of the Qwiklabs [Exploring APIs](#) Quest. 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 this Quest](#) and get immediate completion credit if you've taken this lab. [See other available Qwiklabs Quests](#).

Take your Next Lab

Be sure to check out the following labs for more practice with the APIs Explorer:

- [APIs Explorer: IoT](#)

Next Steps / Learn More

- Documentation for the [App Engine Admin API](#)

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. [Our classes](#) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](#) help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated: August 19, 2020

Lab Last Tested: July 7, 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.