Introduction

API Gateway enables you to provide secure access to your services through a well-defined REST API that is consistent across all of your services, regardless of service implementation. A consistent API:

•Makes it easy for app developers to consume your services
•Enables you to change the backend service implementation without affecting the public API
•Enables you to take advantage of the scaling, monitoring, and security features built into the Google Cloud
In this lab, you will deploy an API on API Gateway to secure traffic to a backend service.

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).

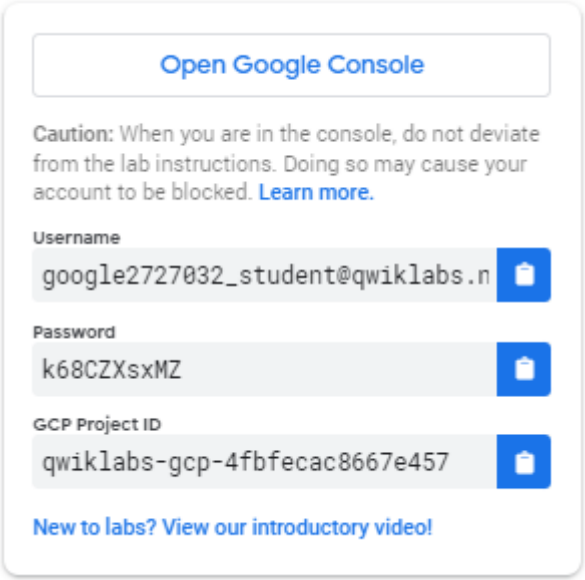
•Time to complete the lab.

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

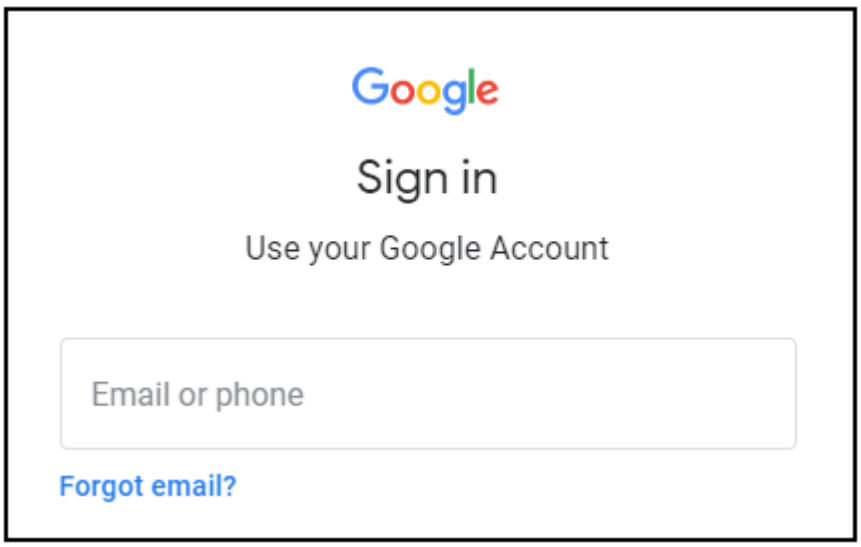
Note: If you are using a Chrome OS device, open an Incognito window to run this lab.

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.

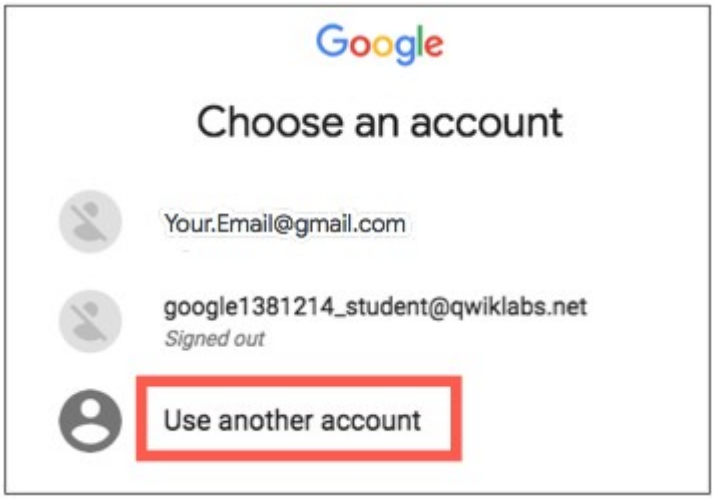


2.Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



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

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.

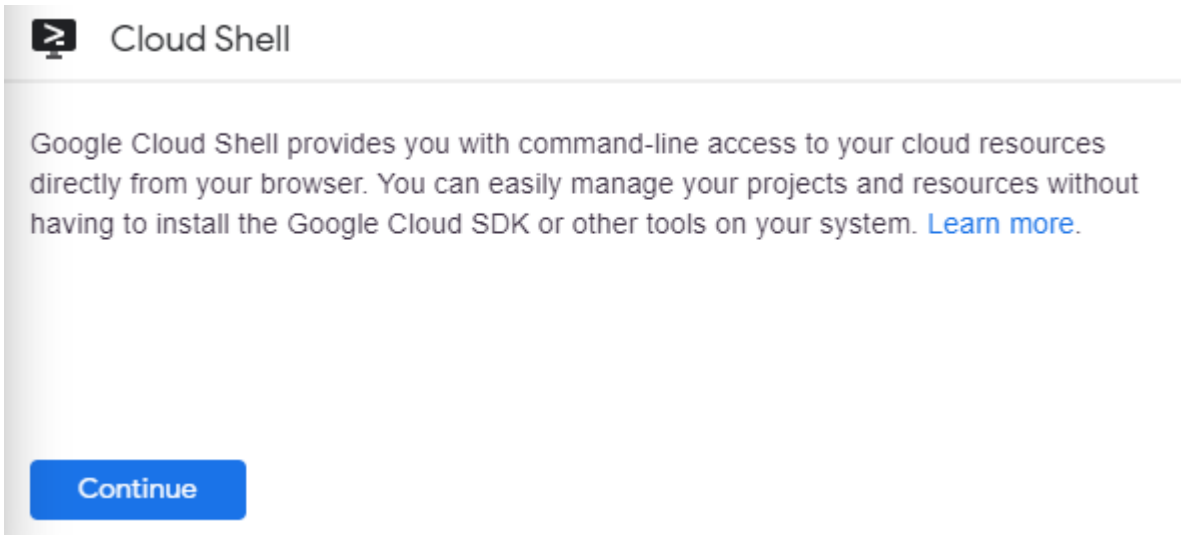
Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left.

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:

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
```
Copied!

(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
```
Copied!

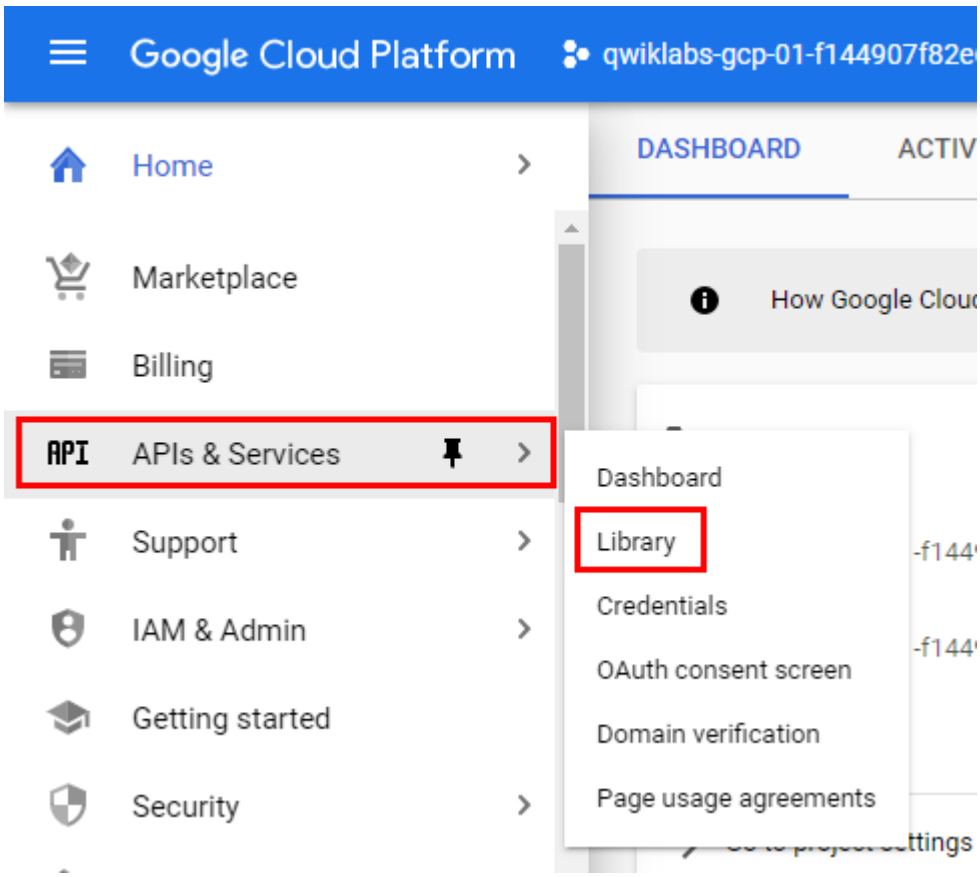
(Output)

```
[core]
project = <project_ID>
```
(Example output)

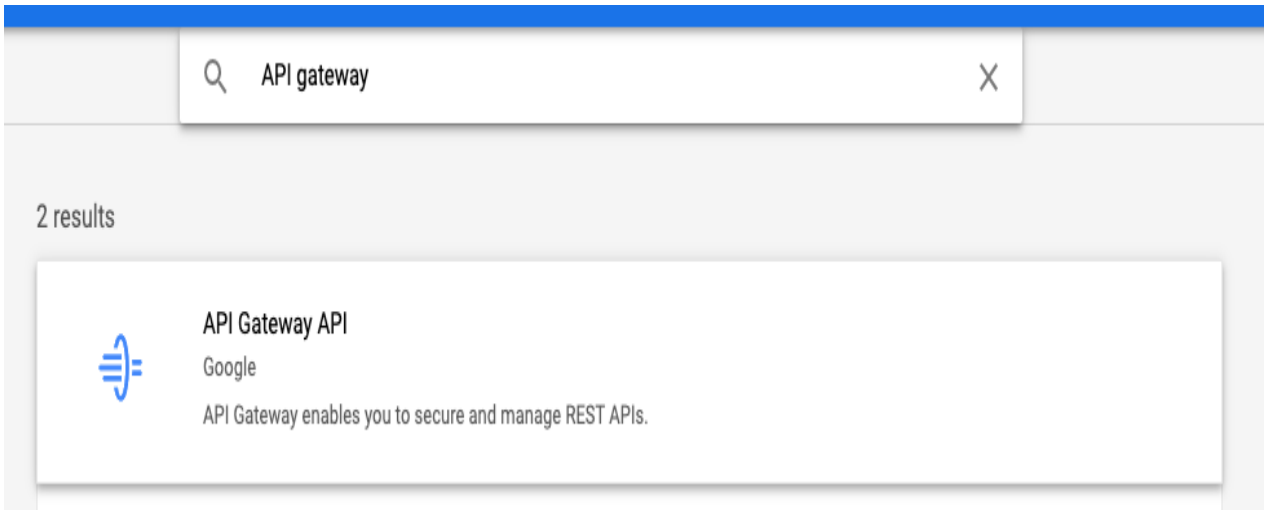
```
[core]
project = qwiklabs-gcp-44776a13dea667a6
```
gcloud see the gcloud command-line tool overview.

Enable the required APIs

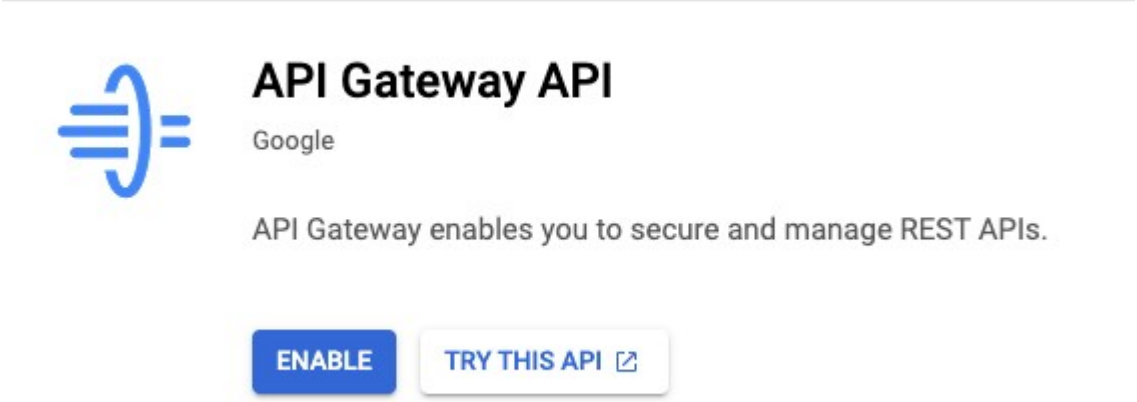
In the Cloud Console, navigate to **APIs & Services** > **Library**:



Start typing "api gateway" in the Search bar, then select the **API Gateway** tile:



Now click the **Enable** button on the next screen.



Deploying an API Backend

API Gateway sits in front of a deployed backend service and handles all incoming requests. In this lab, API Gateway routes incoming calls to a Cloud Function backend named **helloGET** that contains the function shown below:

```
/**
 * HTTP Cloud Function.
 * This function is exported by index.js, and is executed when
 * you make an HTTP request to the deployed function's endpoint.
 *
 * @param {Object} req Cloud Function request context.
 *  More info: https://expressjs.com/en/api.html#req
 * @param {Object} res Cloud Function response context.
 *  More info: https://expressjs.com/en/api.html#res
 */
exports.helloGET = (req, res) => {
      res.send('Hello World!');
};
```

Copied!

In Cloud Console, clone the Cloud Function sample repository.

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

Copied!

Change to the directory that contains the Cloud Functions sample code:

```
cd nodejs-docs-samples/functions/helloworld/
```
Copied!

To deploy the function with an HTTP trigger, run the following command in the directory containing your function:

```
gcloud functions deploy helloGET --runtime nodejs10 --trigger-http --allow-unauthenticated
```
Copied!

Authorize. It will take a few minutes to deploy the cloud function. Wait for the operation to complete before proceeding.

Click Check my progress to verify the objective.

Test the API Backend

When the function finishes deploying, take note of the httpsTrigger's url property or find it using the following command:

```
gcloud functions describe helloGET
```
Copied!

The output should look similar to the URL below where PROJECT_ID is a value specific to your project. To obtain your PROJECT_ID you can run the following command in the Cloud Shell console:

```
export PROJECT_ID=$(gcloud config get-value project)
```
Copied!

Visit the URL to invoke the Cloud Function. You should see the message Hello World! as the response:

```
curl -v https://us-central1-${PROJECT_ID}.cloudfunctions.net/helloGET
```
Copied!

Click Check my progress to verify the objective.

Create the API Definition

API Gateway uses an API definition to route calls to the backend service. You can use an OpenAPI spec that contains specialized annotations to define the desired API Gateway behavior. The OpenAPI spec for this quickstart contains routing instructions to the Cloud Function backend.

1.From Cloud Shell, navigate back to your home directory:

```
cd ~
```
Copied!

2.Create a new file named openapi2-functions.yaml:

```
touch openapi2-functions.yaml
```
Copied!

3.Copy and paste the contents of the OpenAPI spec shown below into the newly created file:

```
# openapi2-functions.yaml
swagger: '2.0'
info:
  title: API_ID description
  description: Sample API on API Gateway with a Google Cloud Functions backend
  version: 1.0.0
schemes:
  - https
produces:
  - application/json
paths:
  /hello:
    get:
      summary: Greet a user
      operationId: hello
      x-google-backend:
        address: https://us-central1-PROJECT_ID.cloudfunctions.net/helloGET
      responses:
        '200':
          description: A successful response
          schema:
            type: string
```
Copied!

4.Set the following environment variables:

```
export API_ID="hello-world-$(cat /dev/urandom | tr -dc 'a-z' | fold -w ${1:-8} | head -n 1)"
```

Copied!

5.Run the following commands to replace the variables set in the last step in the OpenAPI spec file:

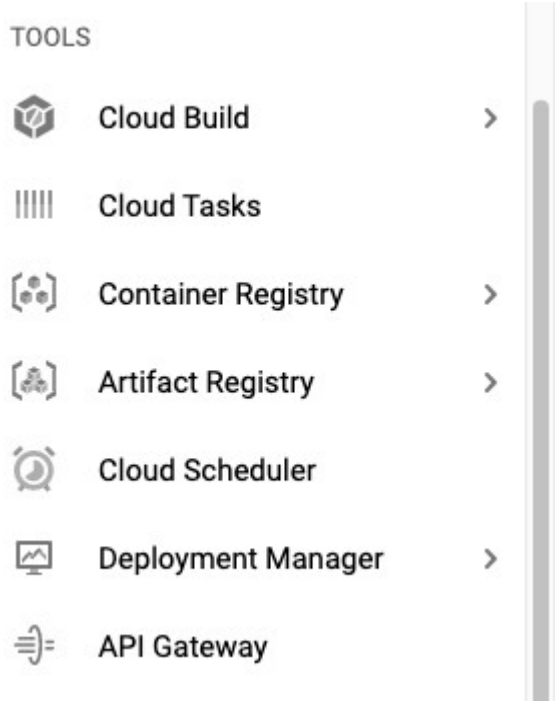
```
sed -i "s/API_ID/${API_ID}/g" openapi2-functions.yaml
sed -i "s/PROJECT_ID/$PROJECT_ID/g" openapi2-functions.yaml
```

Copied!

Creating a Gateway

Now you are ready to create and deploy a gateway on API Gateway.

1.From the **Navigation menu**, open the **API Gateway** page.



2.Click **Create Gateway**.

3.In the **API** section:

•Ensure the **Select an API** input is set to **Create new API**.

•For **Display Name** enter Hello World API

•For **API ID**, run the following command to once again obtain the API ID and enter it into the **API ID** field.

```
export API_ID="hello-world-$(cat /dev/urandom | tr -dc 'a-z' | fold -w ${1:-8} | head -n 1)"
echo $API_ID
```

Copied!

4.In the **API Config** section:

•Ensure the **Select a Config** input is set to **Create a new API config**.

•Do the following to upload the openapi2-functions.yaml file previously created. In Cloud Shell, run the following command.

```
cloudshell download $HOME/openapi2-functions.yaml
```

Copied!

openapi2-functions.yaml is now downloaded to your local machine.

Select **Browse** and select the file from the browser's download location:

•Enter Hello World Config in the **Display Name** field.

•Ensure the **Select a Service Account** input is set to **Compute Engine default service account**.

5.In the **Gateway details** Section:

•Enter Hello Gateway in the **Display Name** field.

•Set the **Location** drop down to us-central1.

6.Click **Create Gateway**.

Create gateway

API

Select an API
Create new API

Display Name *
Hello World API

API ID *
hello-world-wrbqqrqg

API ID can have lowercase letters, digits or hyphens. It must start with a lowercase letter and end with a letter or number.

Managed service

hello-world-wrbqqrqg.apigateway.qwiklabs-gcp-01-b199f5385da7.cloud.goog

Labels

+ ADD LABEL

API Config

Select a Config
Create new API config

Upload an API Spec *
openapi2-functions.yaml ✕ BROWSE

OpenAPI Spec 2.0 is supported. Learn more

Display Name *
Hello World Config

Select a Service Account *
Compute Engine default service account

Identity to be used by gateway.

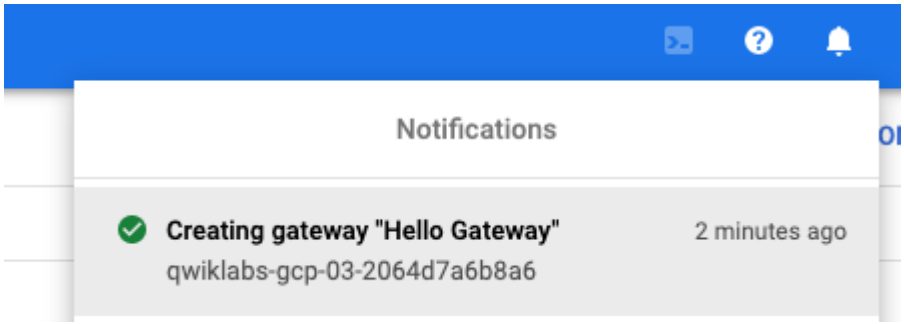
Labels

+ ADD LABEL

Gateway details

Display Name *

It will take several minutes (~10 minutes) for the Create Gateway operation to complete. To check the status of the creation and deployment process, you can click the Notification icon in the main navigation bar to display a status notification, as shown in the image below. Please ensure that the icon status has a green check next to it before proceeding.



Click Check my progress to verify the objective.

Testing your API Deployment

Now you can send requests to your API using the URL generated upon deployment of your gateway.

In Cloud Shell, enter the following command to retreive the GATEWAY_URL of the newly created API hosted by API Gateway:

```
export GATEWAY_URL=$(gcloud api-gateway gateways describe hello-gateway --location us-central1 --format json | jq -r .defaultHostname)
```
Copied!

Run the following command to ensure that the GATEWAY_URL environment variable is set. If it is not, that means you will need to wait longer for the API Gateway to be deployed.

```
echo $GATEWAY_URL
```
Copied!

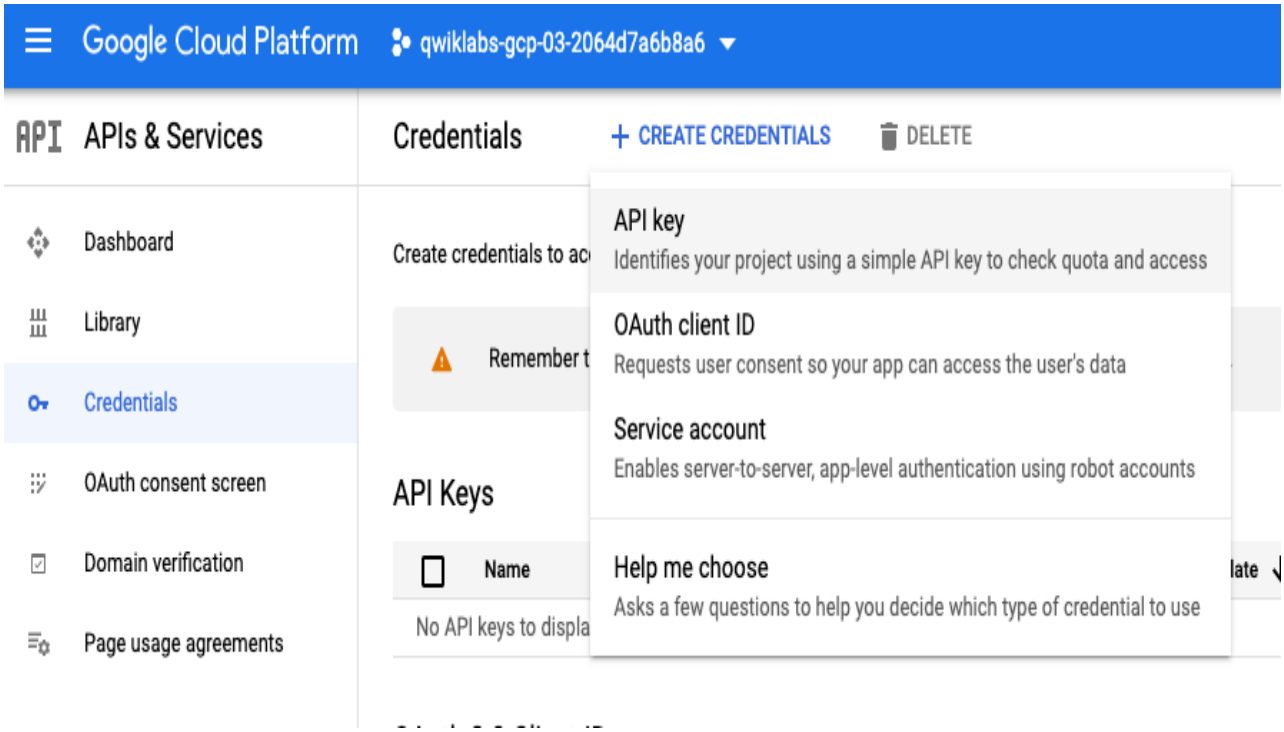
Run the following curl command and validate that the response returned is Hello World!.

```
curl -s -w "\n" https://$GATEWAY_URL/hello
```
Copied!

Securing Access by Using an API Key

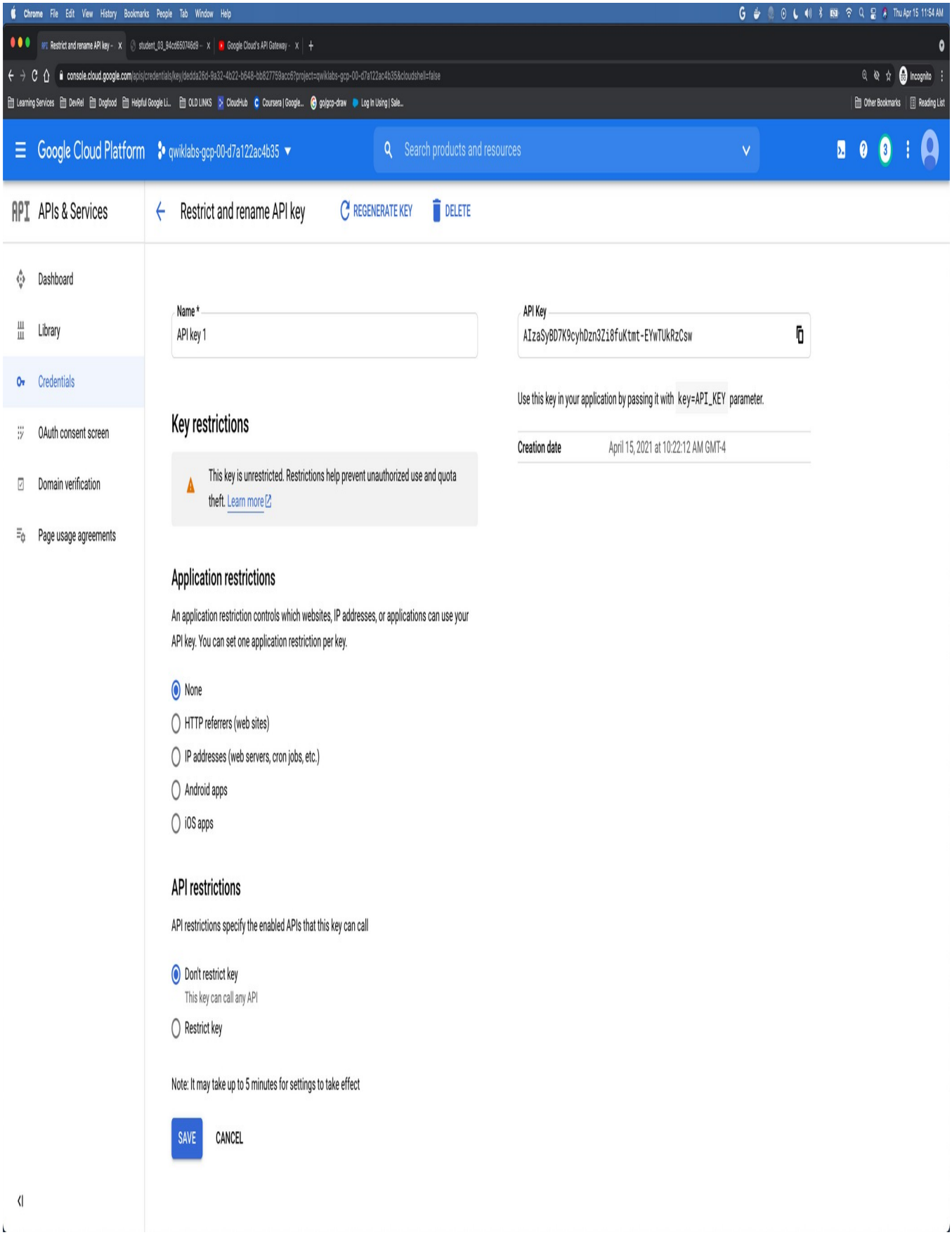
To secure access to your API backend, you can generate an API key associated with your project and grant that key access to call your API. To create an API Key you must do the following:

•In the Cloud Console, navigate to **APIs & Services** > **Credentials**.
•Select **Create credentials**, then select **API Key** from the dropdown menu.
•The **API key created** dialog box displays your newly created key.



Click Check my progress to verify the objective.

Copy the API Key by clicking on the copy icon next to the key in the **Credentials** page.



Store the API Key value in Cloud Shell by running the following command:

```
export API_KEY=REPLACE_WITH_COPIED_API_KEY
```
Copied!

Enable API Key support for your service.

1.In the Cloud Console, navigate to **APIs & Services** > **Library**.

2.In the search bar, enter the Managed Service name of the API you just created. You can find this value in the Managed Service column for your API on the APIs landing page.

3.In Cloud Shell, obtain the name of the managed service you just created using the following command:

```
gcloud api-gateway apis list --format json | jq -r .[0].managedService | cut -d'/' -f6
```
Copied!

4.In the search bar, enter the Managed Service name of the API you just created. Click the **Enable** button for the service once displayed.
This API is provided by a third-party and it will be visible then click **Ok**.

Modify the OpenAPI Spec to leverage API Key Security

In this section, modify the API config of the deployed API to enforce an API key validation security policy on all traffic.

1.Add the security type and securityDefinitions sections to a new file called openapi2-functions2.yaml file as shown below:

```
touch openapi2-functions2.yaml
```
Copied!

2.

```
# openapi2-functions.yaml
swagger: '2.0'
info:
   title: API_ID description
   description: Sample API on API Gateway with a Google Cloud Functions backend
   version: 1.0.0
schemes:
   - https
produces:
```

```
  - application/json
paths:
  /hello:
    get:
      summary: Greet a user
      operationId: hello
      x-google-backend:
        address: https://us-central1-PROJECT_ID.cloudfunctions.net/helloGET
      security:
        - api_key: []
      responses:
        '200':
          description: A successful response
          schema:
            type: string
securityDefinitions:
  api_key:
    type: "apiKey"
    name: "key"
    in: "query"
```

Copied!

3.Run the following commands to replace the variables set in the last step in the OpenAPI spec file:

```
sed -i "s/API_ID/${API_ID}/g" openapi2-functions2.yaml
sed -i "s/PROJECT_ID/$PROJECT_ID/g" openapi2-functions2.yaml
```

Copied!

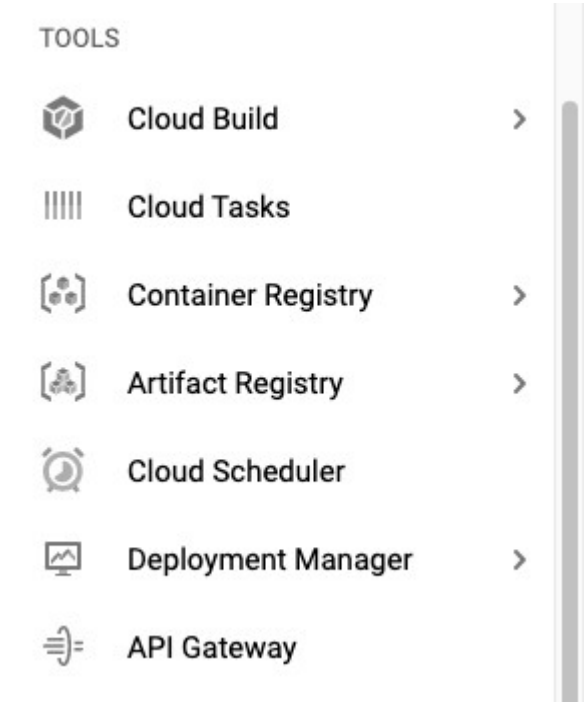
Download the updated API spec file, you will use it to update the Gateway config in the next step:

```
cloudshell download $HOME/openapi2-functions2.yaml
```

Copied!

Create and deploy a new API config to your existing gateway

1.Open the **API Gateway** page in Cloud Console. Under the Left Menu > API Gateway.

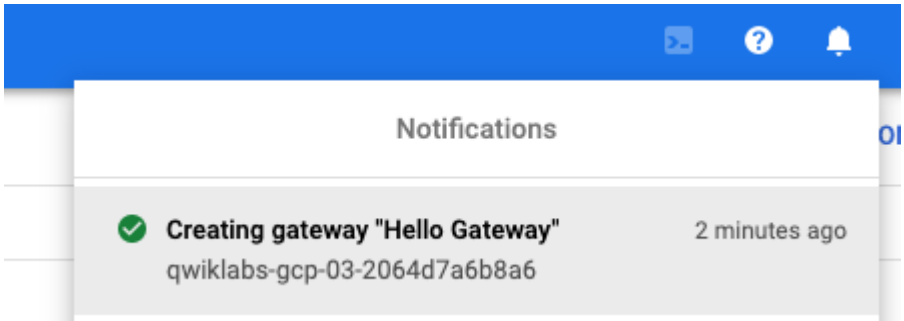


2.Select your API from the list to view details.

3.Select the **Gateways** tab.

4.Select Hello Gateway from the list of available **Gateways**.
5.Click on Edit at the top of the Gateway page.
6.Under **API Config** change the drop down to Create new API config.
7.Click **Browse** in the **Upload an API Spec** input box and select the openapi2-functions2.yaml file.
8.Enter Hello Config for **Display Name**.
9.Select Qwiklabs User Service Account for **Select a Service Account**.
10.Click **Update**.

It will take several minutes (~10 minutes) for the Update Gateway operation to complete. To check the status of the creation and deployment process, you can click the Notification icon in the main navigation bar to display a status notification, as shown in the image below. Please ensure that the icon status has a green check next to it before proceeding.



Click Check my progress to verify the objective.

Testing Calls Using Your API Key

To test using your API key run the following command:

```
export GATEWAY_URL=$(gcloud api-gateway gateways describe hello-gateway --location us-central1 --format json | jq -r .defaultHostname)
curl -sL $GATEWAY_URL/hello
```
Copied!

You should see a response similar to the following error as an API key was not supplied with the curl call: UNAUTHENTICATED:Method doesn't allow unregistered callers (callers without established identity). Please use API Key or other form of API consumer identity to call this API.

Run the following curl command with the key query parameter and use the API key previously created to call the API. If you do not have the API_KEY environment variable set you can get your API key from the left menu by navigating **APIs & Services** > **Credentials**. The key will be available under the **API Keys** section.

```
curl -sL -w "\n" $GATEWAY_URL/hello?key=$API_KEY
```
Copied!

The response returned from the API should now be Hello World!.

Click Check my progress to verify the objective.

Congratulations!

You have successfully protected an API backend with API Gateway. Now you can start onboarding new API clients by generating additional API keys.

Finish Your Quest

This self-paced lab is part of the Qwiklabs Secure and Rate Limit API calls with API Gateway 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.

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: July 19, 2021
Lab Last Tested: May 25, 2021