

**TY B.Tech Computer Engineering
2018 Course Cloud Computing**



SUBMITTED TO: PROF. VISHAL MESHRAM

**VISHWAKARMA INSTITUTE OF INFORMATION
TECHNOLOGY, PUNE**

COMPUTER ENGINEERING DEPARTMENT

SUBMITTED BY:

NAME: AMEY SUNIL BOBADE

G.R NO.: 21810866

ROLL NO.: 321007

CLASS: COMP A

BATCH: A1

Assignment No 1

Title: Study the Google Cloud Platform.

Theory : link

(<https://cloud.google.com/docs>)

(<https://www.youtube.com/playlist?list=PLIivdWyY5sqIUkH9XhgMkAOyIwCN4H0As>)

What Is Cloud Computing?

Cloud computing, in simple words, is accessing and storing data over the Internet instead of doing it on your personal hard drive.

It offers services like storage, database, networking, and more over the Internet to provide faster, innovative, and flexible resources to its customers. The customers get to pay only for the resources they use, hence helping them lower their operating costs and run their business infrastructure more efficiently.

Now, among various cloud providers like AWS, Microsoft, VMWare, IBM, etc., Google Cloud has been the talk of the town in recent years and there are enough reasons behind it. Let's begin with starters and understand what Google Cloud is.

What Is Google Cloud Platform?

Google Cloud is a suite of Cloud Computing services offered by Google. The platform provides various services like compute, storage, networking, Big Data, and many more that run on the same infrastructure that Google uses internally for its end users like Google Search and YouTube.

Google server hasn't gone down in years. So, if you are planning to run your application on the Google Cloud infrastructure, then you can be assured of your applications being safe and secure.

ABOUT GOOGLE CLOUD SERVICES

([HTTPS://CLOUD.GOOGLE.COM/DOCS/OVERVIEW/CLOUD-PLATFORM-SERVICES](https://cloud.google.com/docs/overview/cloud-platform-services))

This overview introduces some of the commonly used Google Cloud services. For the full list of services, see the [Products and services page](#).

This overview covers the following types of services:

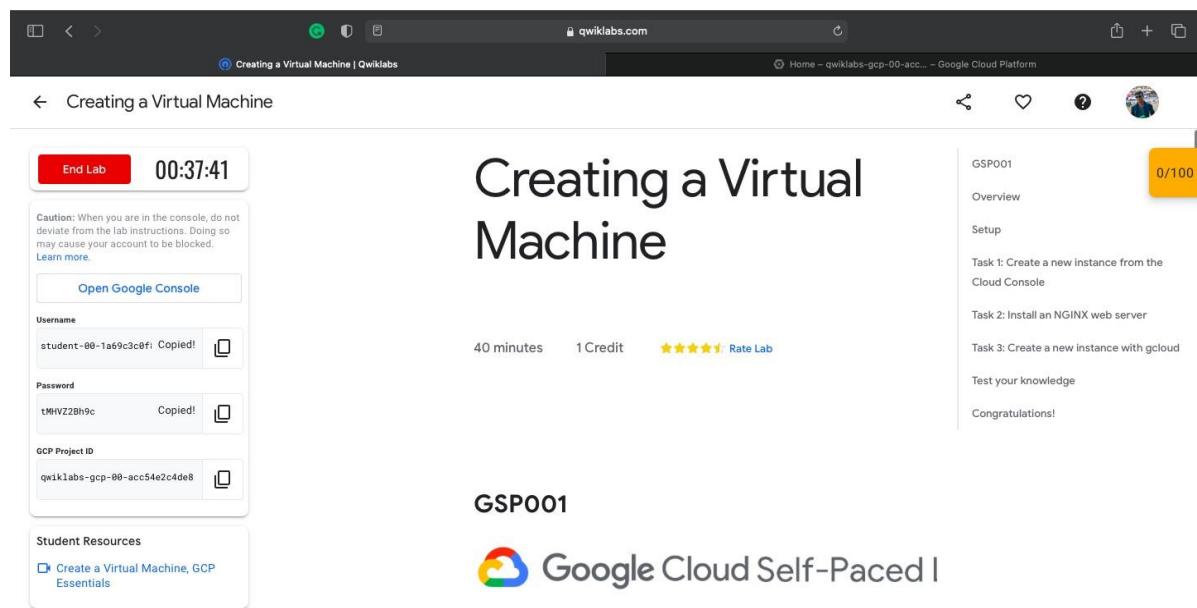
- [Computing and hosting](#)
- [Storage](#)
- [Databases](#)
- [Networking](#)
- [Big data](#)
- [Machine learning](#)

OUT of Google Product you have to study 3 services in detail:

1. Compute (<https://cloud.google.com/docs#section-7>)

a) In compute write detail steps of how to create a Linux virtual machine instance in Compute Engine using the Google Cloud Console. (Reference: <https://cloud.google.com/compute/docs/quickstart-linux>)

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.

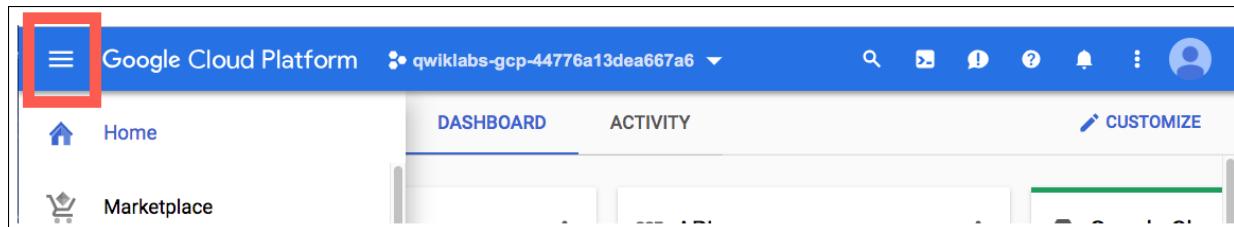


Overview

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.

1. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.
2. 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.

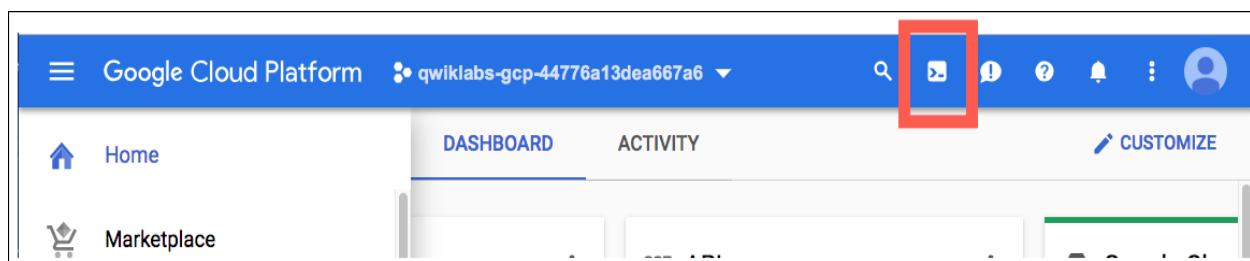
After a few moments, the Cloud Console opens in this tab.



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**.

A screenshot of a confirmation page for activating Cloud Shell. At the top left is a terminal icon followed by the text "Cloud Shell". Below this is a descriptive paragraph: "Google Cloud Shell provides you with command-line access to your cloud resources directly from your browser. You can easily manage your projects and resources without having to install the Google Cloud SDK or other tools on your system." A blue "Learn more" link is present in the text. At the bottom of the page is a large blue "Continue" button.

The screenshot shows the Google Cloud Platform dashboard for the project 'qwiklabs-gcp-00-acc54e2c4de8'. The left sidebar includes sections for Home, Marketplace, Billing, APIs & Services, Support, IAM & Admin, Getting started, Compliance, Security, and Anthos. Under COMPUTE, there are links for App Engine, Compute Engine, Kubernetes Engine, Cloud Functions, Cloud Run, and VMware Engine. The main content area displays 'Project info' (Project name: qwiklabs-gcp-00-acc54e2c4de8, Project ID: qwiklabs-gcp-00-acc54e2c4de8, Project number: 644198355469), 'API APIs' (Requests (requests/sec) chart from 8:30 to 9:15), 'Google Cloud Platform status' (All services normal), and 'Billing' (Estimated charges USD \$0.00 for Mar 1 – 29, 2021). A modal window titled 'Authorize Cloud Shell' is open, stating 'gcloud is requesting your credentials to make a GCP API call.' It has 'Authorize' and 'Reject' buttons.

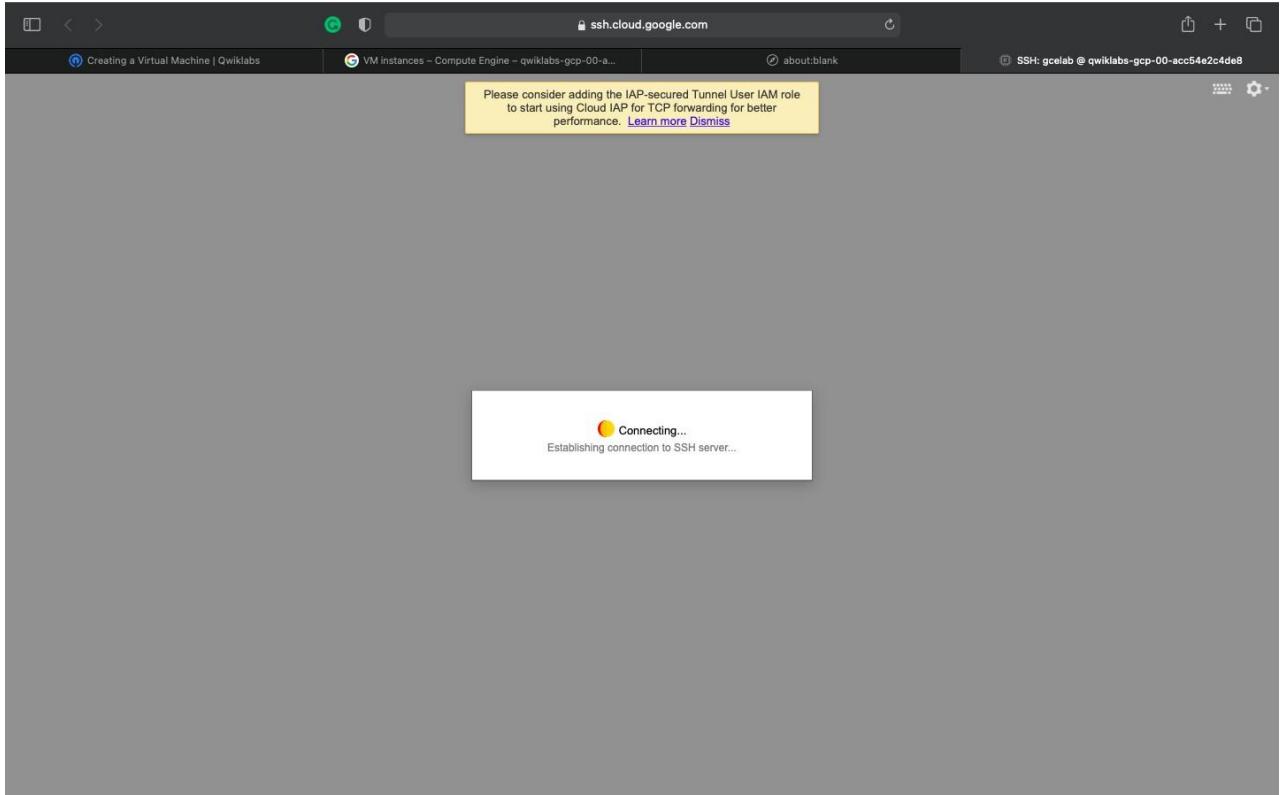
You can list the active account name with the written commands in console :

The screenshot shows the Google Cloud Platform Compute Engine VM instances page. The left sidebar lists Virtual machines, VM instances, Instance templates, Sole-tenant nodes, Machine images, TPUs, Migrate for Compute Engine, Committed use discounts, Marketplace, and Help & Support. The main content area shows a list of VM instances with columns for Name, Zone, Recommendations, In use by, Internal IP, and External IP. A modal window titled 'Select an instance' is open, prompting to 'Please select at least one resource.' Below the list is a 'PERMISSIONS' section. At the bottom, a terminal window shows the output of the command 'gcloud auth list', which lists the active account 'student-00-1a69c3c0f857@qwiklabs.net'.

a) Write detail steps for Deploy a simple Apache web server to learn the basics of running a server on a virtual machine instance. (Reference: <https://cloud.google.com/compute/docs/tutorials/basic-webserver-apache>)

Now you'll install an NGINX web server, one of the most popular web servers in the world, to connect your virtual machine to something.

1. In the SSH terminal, to get root access, run the following command:



In the SSH terminal, to get root access, run the following command:

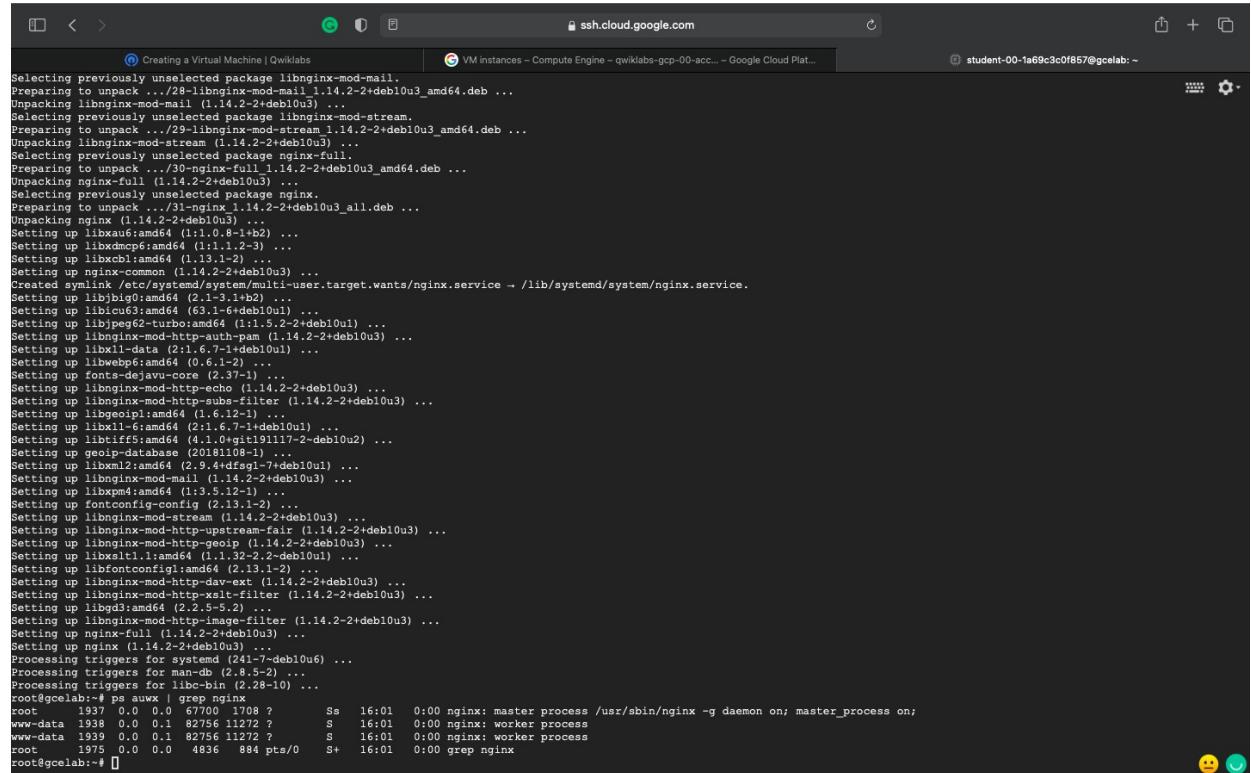
```
sudo su -  
● As the root user, update your OS:
```

```
apt-get update

Get:1 http://security.debian.org stretch/updates InRelease [94.3 kB]
Ign http://deb.debian.org stretch InRelease
Get:2 http://deb.debian.org stretch-updates InRelease [91.0 kB]
```

1. Install NGINX:

```
apt-get install nginx -y
content_copy
Reading package lists... Done
Building dependency tree
```



```
Creating a Virtual Machine | Qwiklabs
ssh.cloud.google.com
student-00-1a69c3c0f857@gcelab: ~
Selecting previously unselected package libnginx-mod-mail.
Preparing to unpack .../28-libnginx-mod-mail_1.14.2-2+deb10u3_amd64.deb ...
Unpacking libnginx-mod-mail (1.14.2-2+deb10u3) ...
Selecting previously unselected package libnginx-mod-stream.
Preparing to unpack .../29-libnginx-mod-stream_1.14.2-2+deb10u3_amd64.deb ...
Unpacking libnginx-mod-stream (1.14.2-2+deb10u3) ...
Selecting previously unselected package nginx-full.
Preparing to unpack .../30-nginx-full_1.14.2-2+deb10u3_amd64.deb ...
Unpacking nginx-full (1.14.2-2+deb10u3) ...
Selecting previously unselected package nginx.
Preparing to unpack .../31-nginx_1.14.2-2+deb10u3_all.deb ...
Unpacking nginx (1.14.2-2+deb10u3) ...
Setting up libxau6:amd64 (1:1.0.8-1+b2) ...
Setting up libxdmcp6:amd64 (1:1.1.2-3) ...
Setting up libxcb1:amd64 (1:1.1.2-3) ...
Setting up libxext6:amd64 (1:1.3.1-2+deb10u3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /lib/systemd/system/nginx.service.
Setting up libxbit0:amd64 (2.1-1.1+b2) ...
Setting up libxcu6:amd64 (63.1-6+deb10u1) ...
Setting up libjpeg62-turbo:amd64 (1:1.5.2-2+deb10u1) ...
Setting up libnginx-mod-http-auth-pam (1.14.2-2+deb10u3) ...
Setting up libx11-data (2:1.6.7-1+deb10u1) ...
Setting up libwebp6:amd64 (0.6.1-2) ...
Setting up fonts-dejavu-core (2.37-1) ...
Setting up libnginx-mod-http-echo (1.14.2-2+deb10u3) ...
Setting up libnginx-mod-http-subrequest (1.14.2-2+deb10u3) ...
Setting up libgeoip1:amd64 (1:1.6.12-1) ...
Setting up libx11-6:amd64 (2:1.6.7-1+deb10u1) ...
Setting up libtiff5:amd64 (4.1.0+git191117-2+deb10u2) ...
Setting up libxml2:amd64 (2.9.4+dfsg1-7+deb10u1) ...
Setting up libnginx-mod-mail (1.14.2-2+deb10u3) ...
Setting up libxmlpp4:amd64 (1:3.5.12-1) ...
Setting up fontconfig-config (2.13.1-2) ...
Setting up libnginx-mod-stream (1.14.2-2+deb10u3) ...
Setting up libnginx-mod-http-upstream-fair (1.14.2-2+deb10u3) ...
Setting up libnginx-mod-http-geopip (1.14.2-2+deb10u3) ...
Setting up libxslt1.1:amd64 (1:1.32-2.2+deb10u1) ...
Setting up libfontconfig1:amd64 (2.13.1-2) ...
Setting up libnginx-mod-http-dav-ext (1.14.2-2+deb10u3) ...
Setting up libnginx-mod-http-geoip (1.14.2-2+deb10u3) ...
Setting up libpdo1:amd64 (2.0.8-5.2) ...
Setting up libnginx-mod-http-image-filter (1.14.2-2+deb10u3) ...
Setting up nginx-full (1.14.2-2+deb10u3) ...
Setting up nginx (1.14.2-2+deb10u3) ...
Processing triggers for systemd (241-7+deb10u6) ...
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for libc-bin (2.28-10) ...
root@gcelab:~# ps auxw | grep nginx
root    1937  0.0  0.0 67700 1708 ?        Ss   16:01  0:00 nginx: master process /usr/sbin/nginx -g daemon on; master_process on;
www-data 1938  0.0  0.1 82756 11272 ?       S    16:01  0:00 nginx: worker process
www-data 1939  0.0  0.1 82756 11272 ?       S    16:01  0:00 nginx: worker process
root    1975  0.0  0.0  4836  884 pts/0   S+   16:01  0:00 grep nginx
root@gcelab:~#
```

2. GOOGLE APP ENGINE (REF: [HTTPS://CLOUD.GOOGLE.COM/APPENGINE/DOCS](https://cloud.google.com/appengine/docs))

- 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 browser window for 'App Engine: Qwik Start - Go | Qwiklabs'. The title bar says 'qwiklabs.com' and 'Google'. The main content area displays the heading 'App Engine: Qwik Start - Go' and a progress bar showing 'GSP070' at '0/5'. Below the heading, it says 'Overview' and lists several steps: 'Setup', 'Enable Google App Engine Admin API', 'Download the Hello World app', 'Deploy your app', 'View your application', 'Test your knowledge', and 'Congratulations!'. To the left, there's a sidebar with a timer '00:19:59', a red 'End Lab' button, and a 'Caution' message about not deviating from lab instructions. It also contains fields for 'Username' (student-00-e11a8c1863c6@qwik1), 'Password' (TBJ34sjg3), and 'GCP Project ID' (qwiklabs-gcp-00-bacff2f14066). A 'Student Resources' section links to 'Build Apps at Scale with Google App Engine'.

Overview

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.

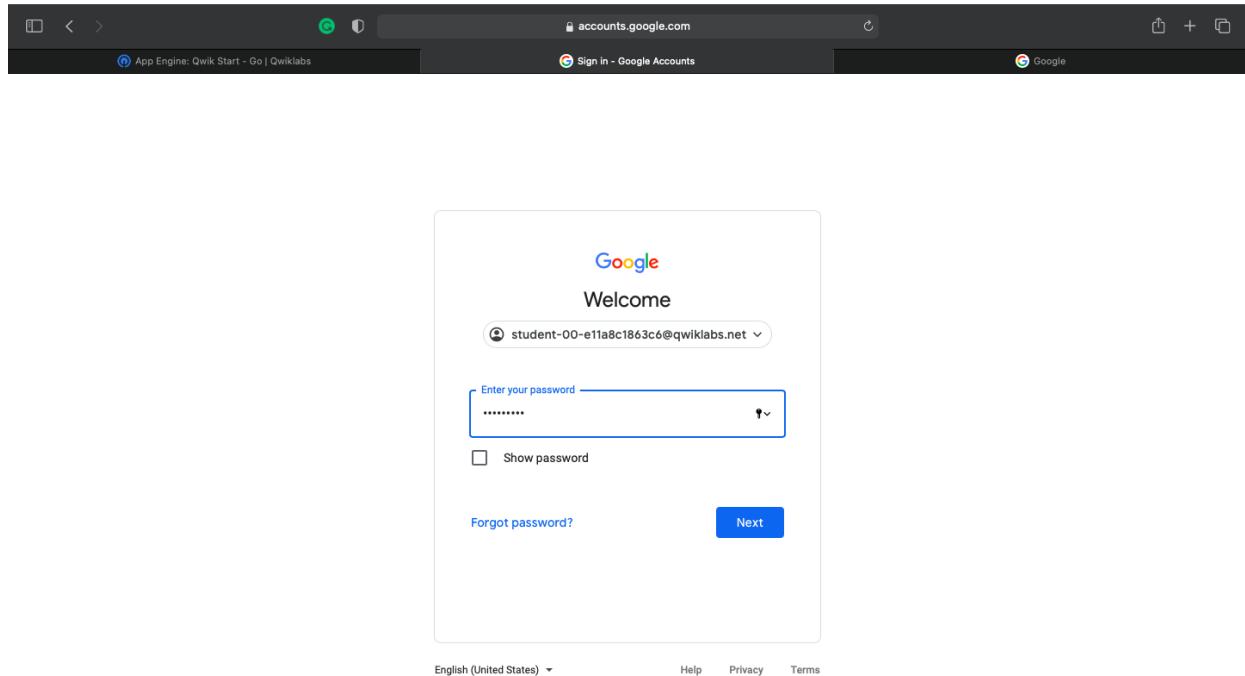
1. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

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).

2. 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.

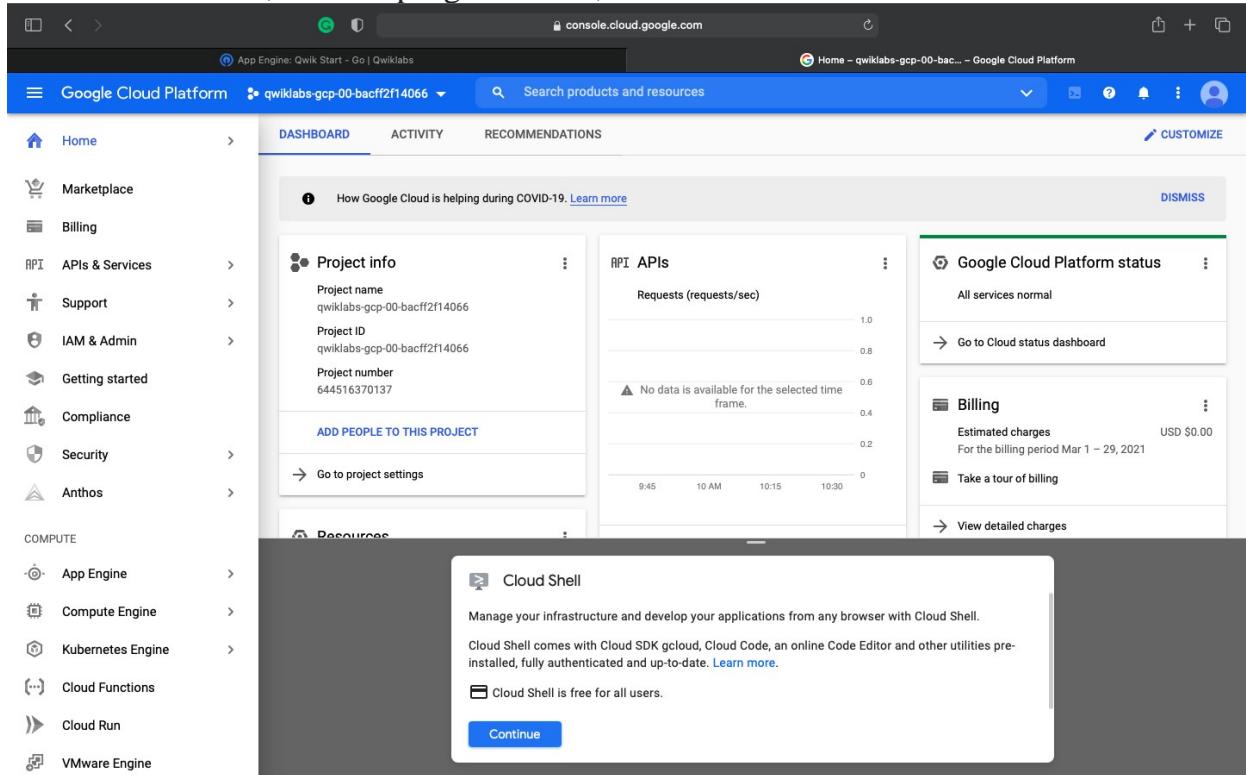
After a few moments, the Cloud Console opens in this tab.



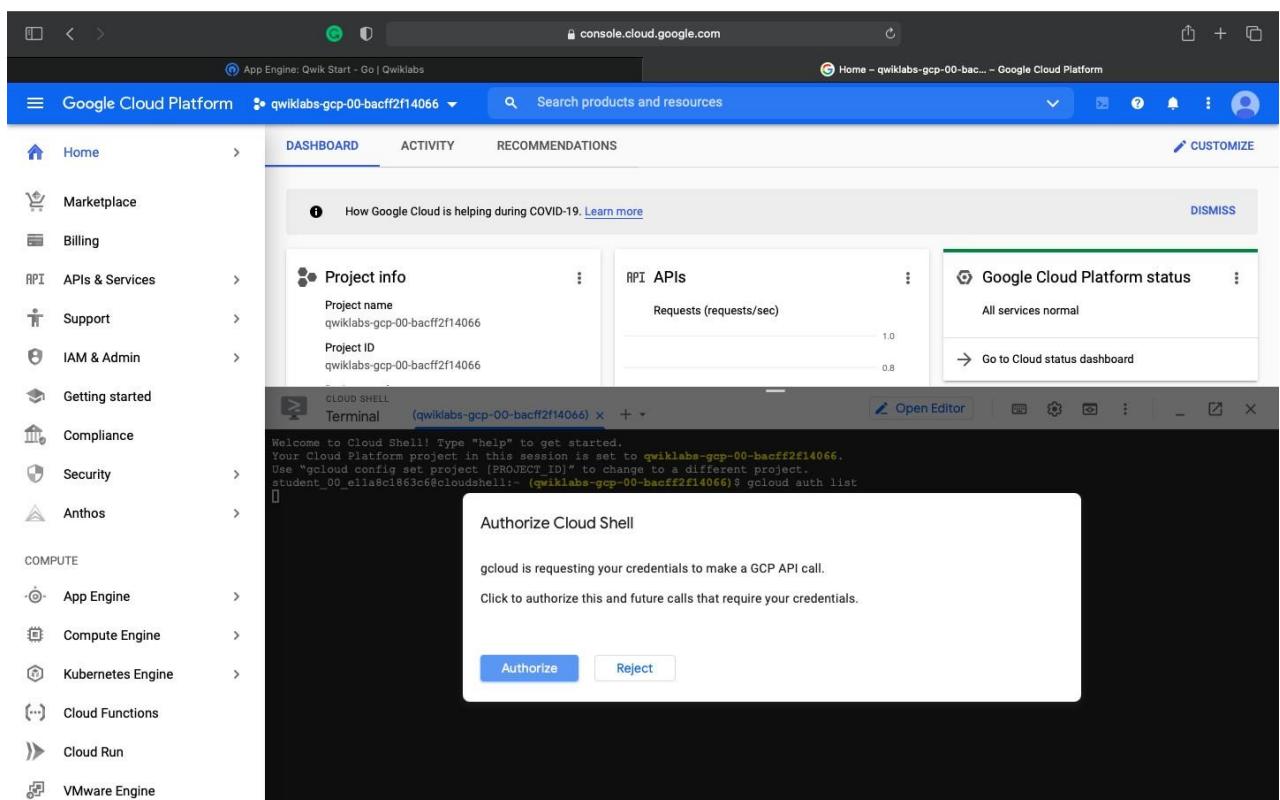
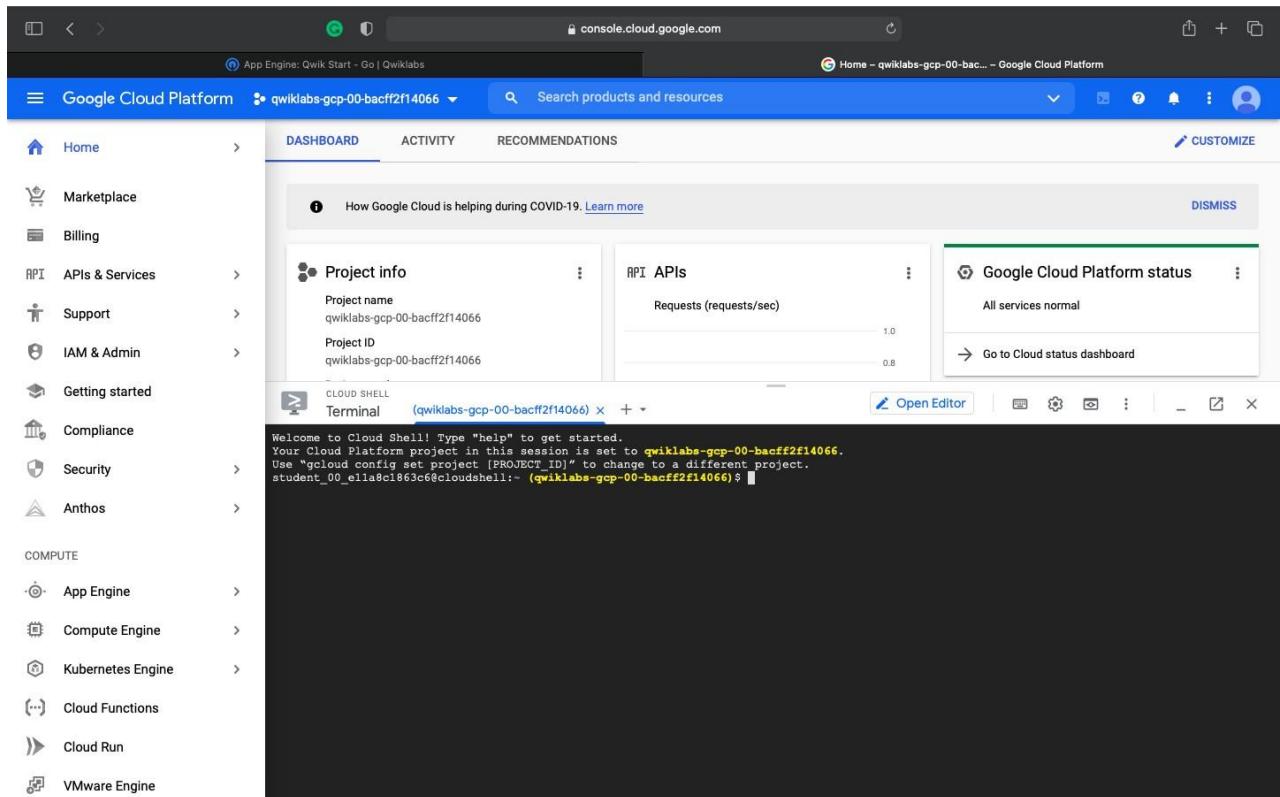
ACTIVATE CLOUD SHELL

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.



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:



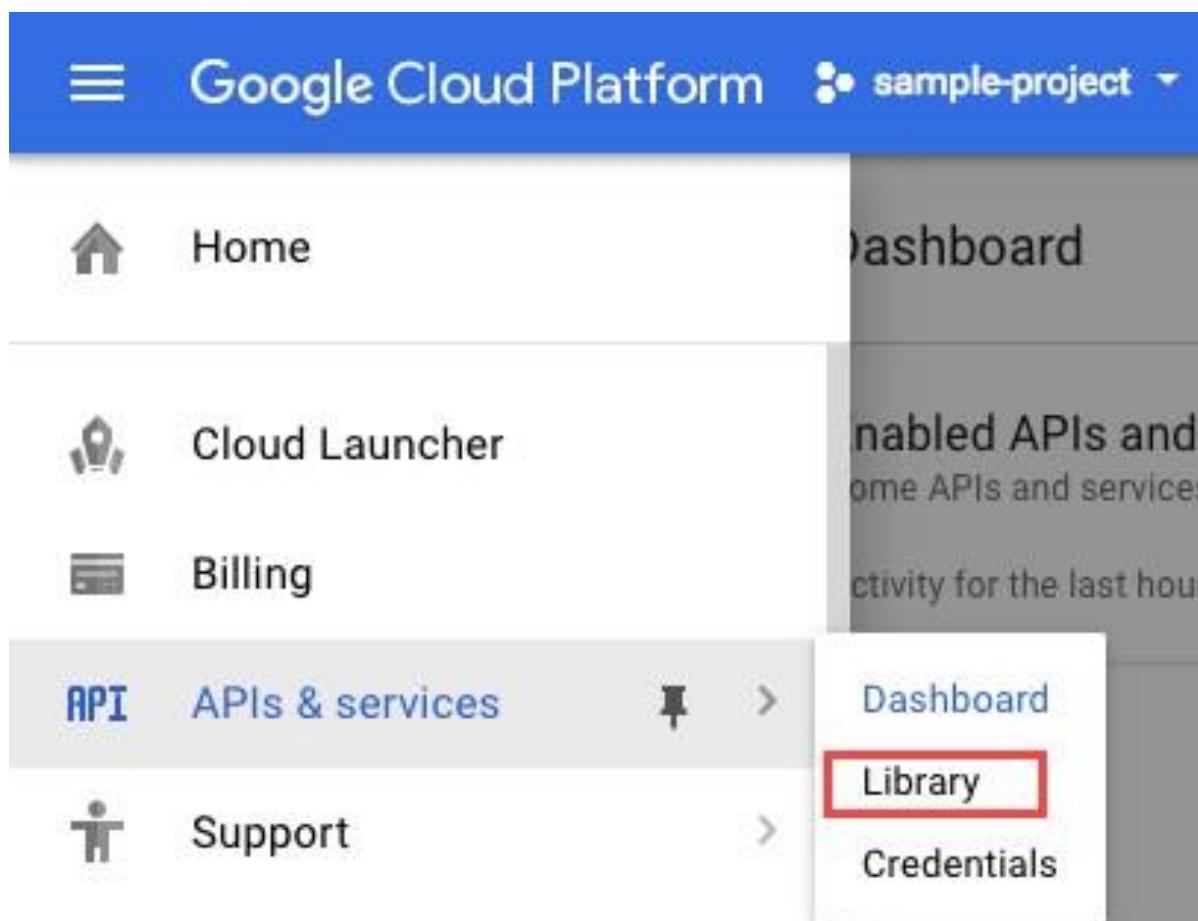
The screenshot shows the Google Cloud Platform dashboard for the project "qwiklabs-gcp-00-bacff2f14066". The left sidebar lists various services: Marketplace, Billing, APIs & Services, Support, IAM & Admin, Getting started, Compliance, Security, Anthos, COMPUTE (App Engine, Compute Engine, Kubernetes Engine), Cloud Functions, Cloud Run, and VMware Engine. The main area displays the "DASHBOARD" tab, which includes sections for "Project info", "API APIs", and "Google Cloud Platform status". A "Cloud Shell" terminal window is open, showing a gcloud session for the specified project. The terminal output includes:

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to qwiklabs-gcp-00-bacff2f14066.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
student_00_ella8c1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ gcloud auth list  
          Credentialed Accounts  
*   student-00-ella8c1863c6@qwiklabs.net  
  
To set the active account, run:  
  $ gcloud config set account 'ACCOUNT'  
student_00_ella8c1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ gcloud config list project  
[core]  
project = qwiklabs-gcp-00-bacff2f14066  
  
Your active configuration is: [cloudshell-10278]  
student_00_ella8c1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$
```

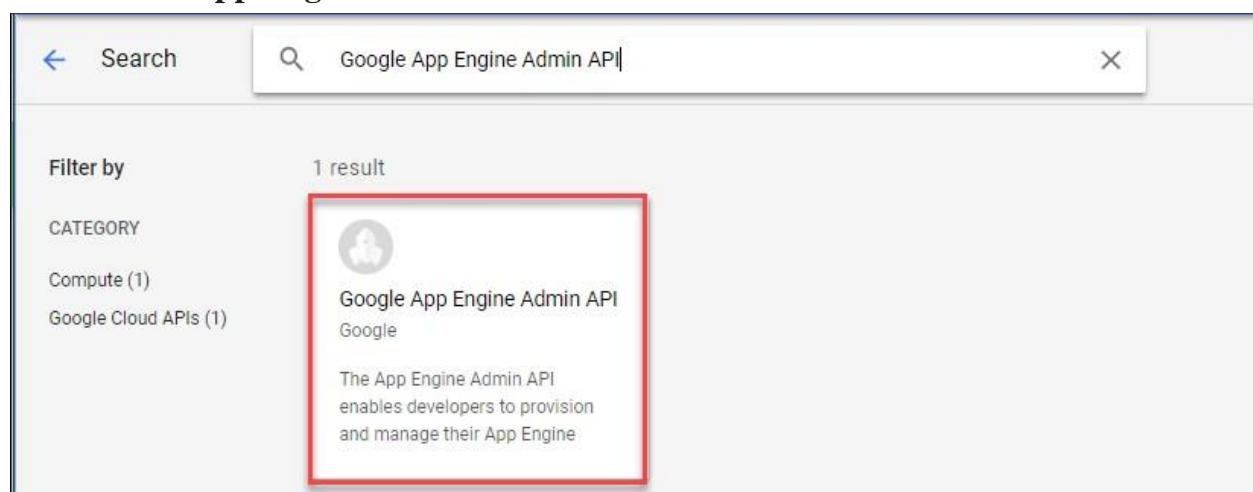
Enable Google App Engine Admin API

The App Engine Admin API enables developers to provision and manage their App Engine Applications.

1. In the left menu, click **APIs & Services > Library**.

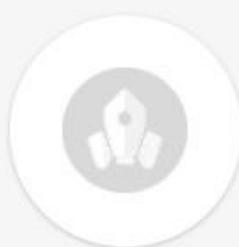


2. Type "App Engine Admin API" in search box.
3. Click **App Engine Admin API**.



4. Click Enable.

[← API Library](#)



Google App Engine Admin API

Google

The App Engine Admin API enables developers to provision and manage their App Engine applications.

[ENABLE](#) [TRY THIS API](#)

console.cloud.google.com

App Engine: Qwik Start - Go | Qwiklabs

Google Cloud Platform • qwiklabs-gcp-00-bacff2f14066

[MANAGE](#) [TRY THIS API](#) API Enabled

App Engine Admin API

Provisions and manages developers' App Engine applications.

[OVERVIEW](#) [DOCUMENTATION](#)

Overview

Provisions and manages developers' App Engine applications.

About Google

Google's mission is to organize the world's information and make it

Additional details

Type: SaaS
Last updated: 3/19/21

CLOUD SHELL Terminal (qwiklabs-gcp-00-bacff2f14066) + ×

ACTIVE ACCOUNT * student-00-ella8c1863c6@qwiklabs.net

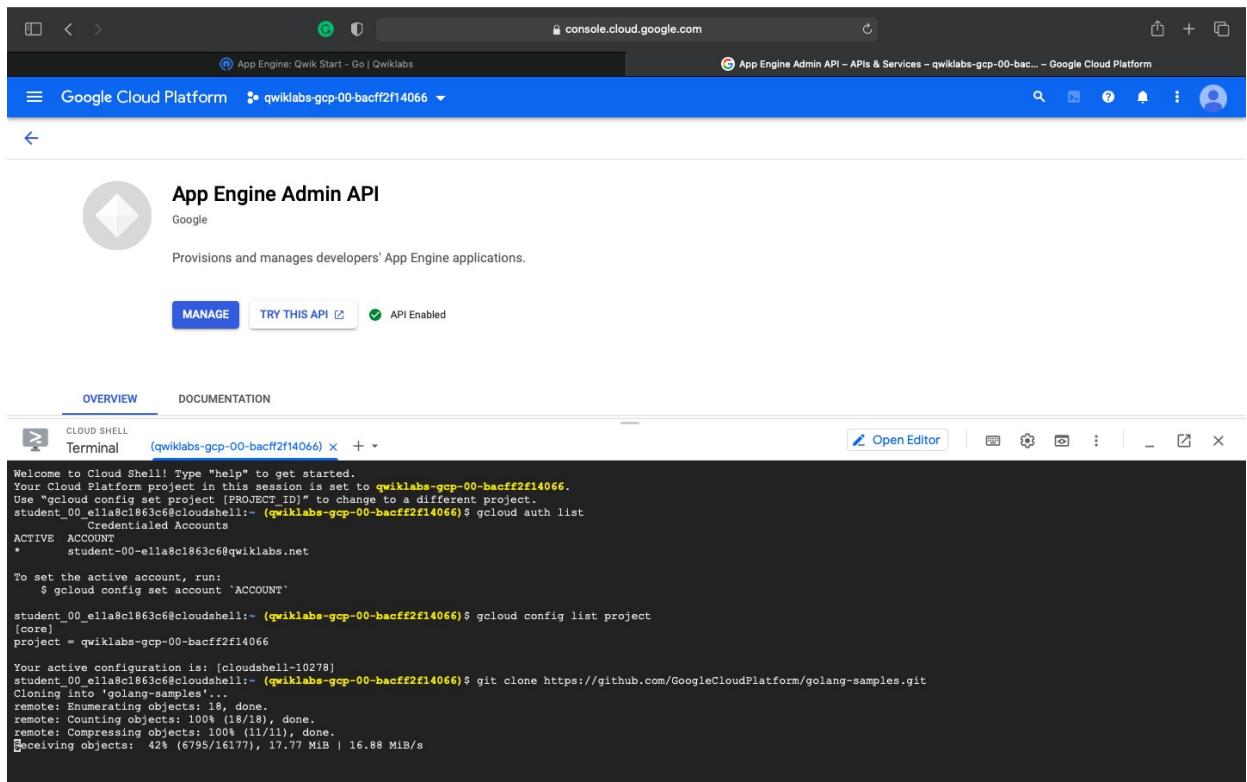
To set the active account, run:
\$ gcloud config set account 'ACCOUNT'

```
student_00_ella8c1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ gcloud config list project
[core]
project = qwiklabs-gcp-00-bacff2f14066

Your active configuration is: [cloudshell-10278]
student_00_ella8c1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ []
```

Deploy your app

1. To deploy your app to App Engine, run the following command from within the root directory of your application where the app.yaml file is located:

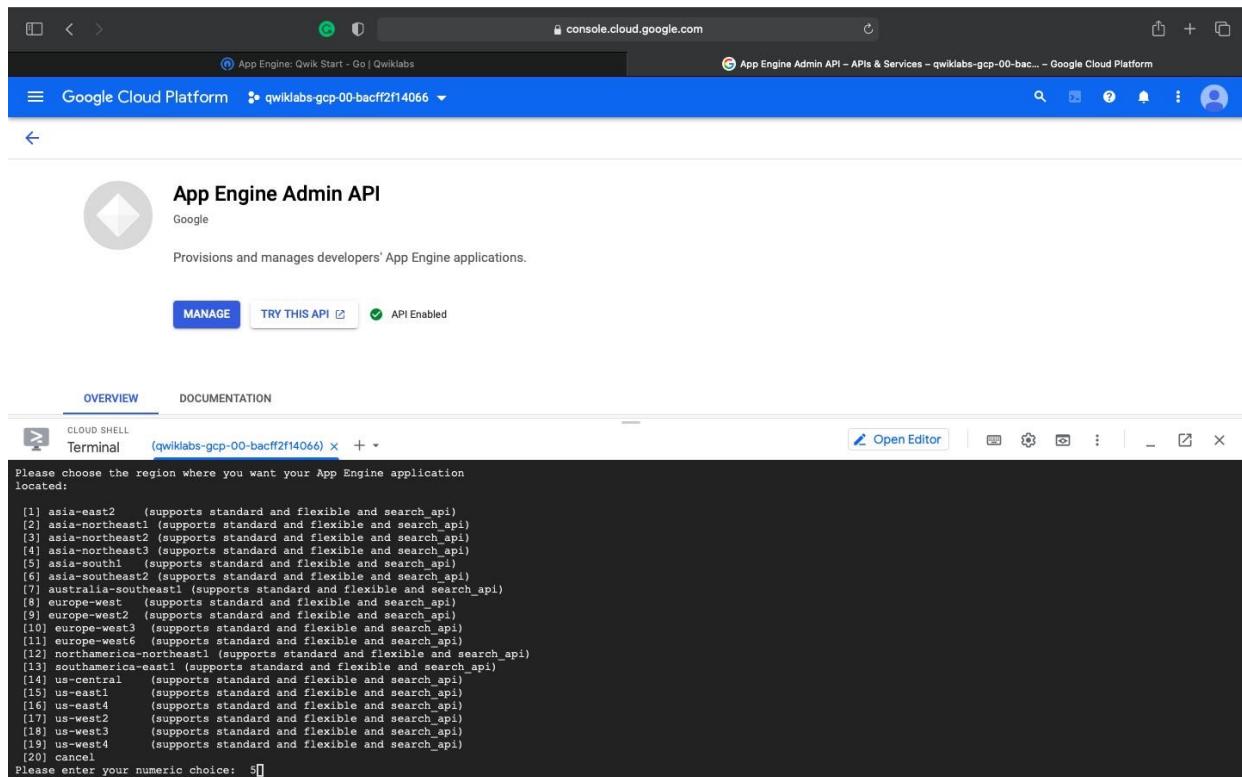


The screenshot shows the Google Cloud Platform interface for the App Engine Admin API. The title bar indicates the project is "qwiklabs-gcp-00-bacff2f14066". The main area displays the "App Engine Admin API" documentation, which includes a "TRY THIS API" button and a status message "API Enabled". Below this, there are tabs for "OVERVIEW" and "DOCUMENTATION". A terminal window titled "CLOUD SHELL Terminal" is open, showing a session for the project "qwiklabs-gcp-00-bacff2f14066". The terminal output shows the user navigating through Cloud Shell, setting the active account, listing credentials, and cloning a GitHub repository for Golang samples.

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to qwiklabs-gcp-00-bacff2f14066.  
Use "gcloud config set project [PROJECT_ID]" to change to a different project.  
student_00_ella@cl1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ gcloud auth list  
  Credentialled Accounts  
ACTIVE ACCOUNT  
* student-00-ella@cl1863c6@google.com  
To set the active account, run:  
$ gcloud config set account 'ACCOUNT'  
student_00_ella@cl1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ gcloud config list project  
[core]  
project = qwiklabs-gcp-00-bacff2f14066  
  
Your active configuration is: [cloudshell-10278]  
student_00_ella@cl1863c6@cloudshell:~ (qwiklabs-gcp-00-bacff2f14066)$ git clone https://github.com/GoogleCloudPlatform/golang-samples.git  

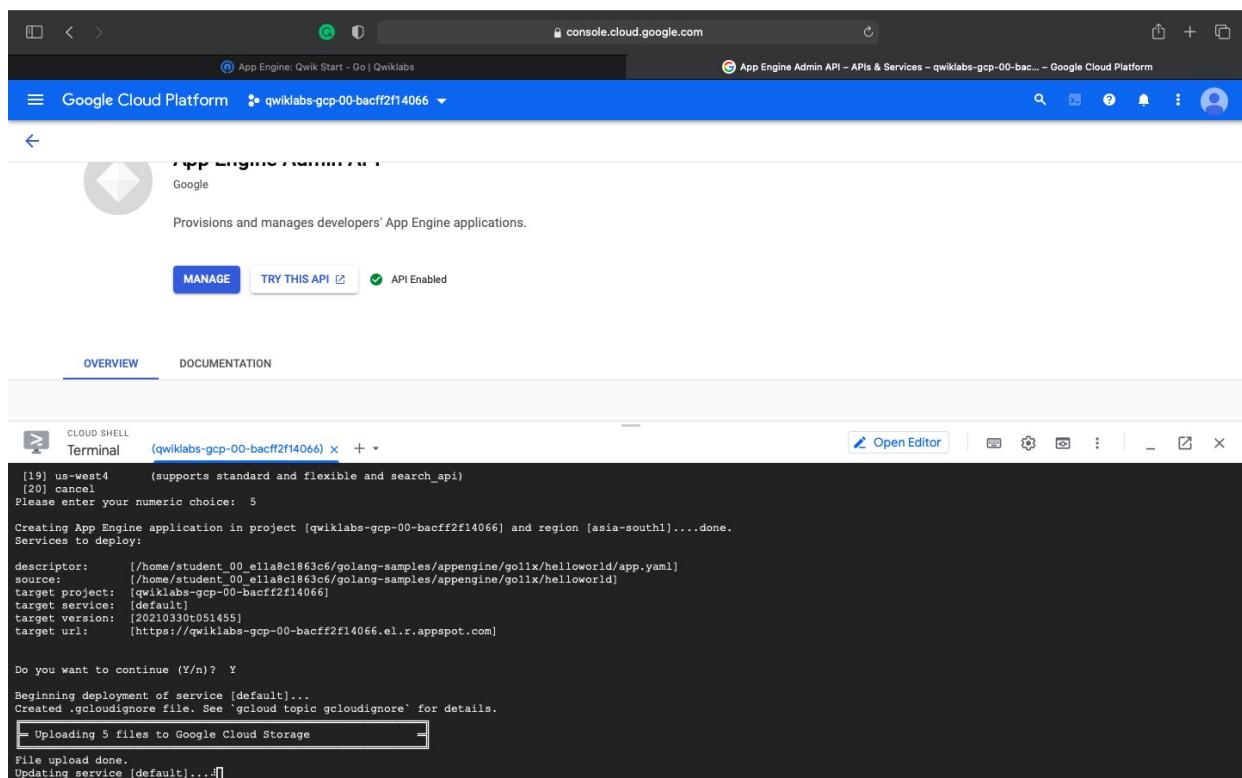
```

2. Type the number for your region.

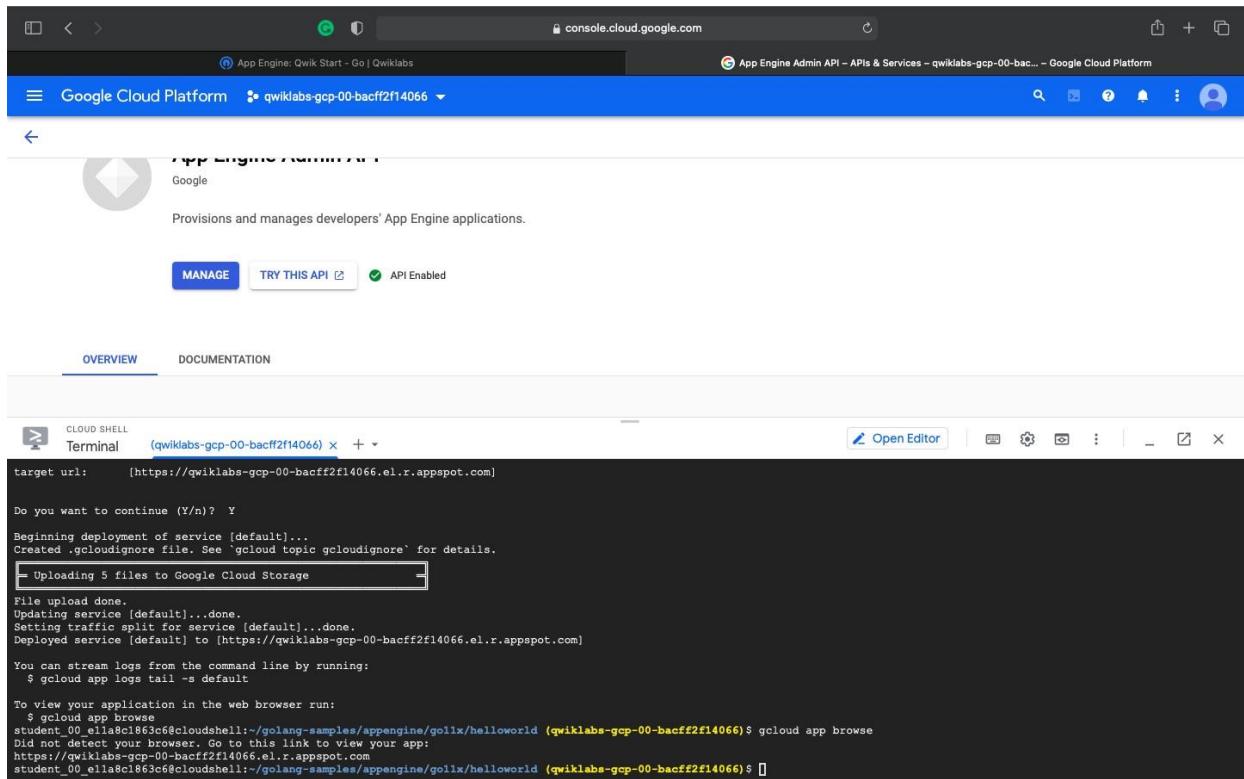


The screenshot shows the Google Cloud Platform interface for the App Engine Admin API. At the top, there are two tabs: "App Engine: Qwik Start - Go | Qwiklabs" and "App Engine Admin API - APIs & Services - qwiklabs-gcp-00-bacff2f14066 - Google Cloud Platform". Below the tabs, the URL "console.cloud.google.com" is visible. The main content area is titled "App Engine Admin API" with a "Google" logo. It says "Provisions and manages developers' App Engine applications." There are three buttons: "MANAGE", "TRY THIS API", and "API Enabled". Below these buttons, there are two tabs: "OVERVIEW" (which is selected) and "DOCUMENTATION". A "CLOUD SHELL" tab is open, showing a terminal window with the title "(qwiklabs-gcp-00-bacff2f14066)". The terminal window displays a list of regions with their descriptions and a prompt: "Please choose the region where you want your App Engine application located:" followed by a numbered list from 1 to 20. The list includes regions like "asia-east2", "asia-northeast1", "asia-northeast2", "asia-northeast3", "asia-south1", "asia-southeast1", "asia-southeast2", "australia-southeast1", "europe-west", "europe-west2", "europe-west3", "europe-west6", "europe-west7", "northamerica-northeast1", "southamerica-east1", "us-central", "us-east1", "us-east2", "us-west2", "us-west3", "us-west4", and "cancel". At the bottom of the terminal window, there is a prompt: "Please enter your numeric choice: 5".

3. Enter Y to confirm the deployment of service when prompted.



The screenshot shows the Google Cloud Platform interface for the App Engine Admin API, similar to the previous one. The terminal window now shows the deployment process. It starts with a list of regions and a choice prompt. Then it moves to a "Creating App Engine application in project [qwiklabs-gcp-00-bacff2f14066] and region [asia-south1]...done." message. It then lists the descriptor, source, target project, target service, target version, and target URL. Finally, it asks "Do you want to continue (Y/n)? Y" and begins the deployment process, showing progress bars for "Uploading 5 files to Google Cloud Storage" and "Updating service [default]...".



The screenshot shows the Google Cloud Platform App Engine Admin API interface. At the top, there are tabs for "App Engine Admin API - APIs & Services - qwiklabs-gcp-00-bacff2f14066 - Google Cloud Platform". Below the tabs, the title is "App Engine Admin API - APIs & Services - qwiklabs-gcp-00-bacff2f14066". There is a "Google" logo and a message stating "Provisions and manages developers' App Engine applications." Below this, there are buttons for "MANAGE" and "TRY THIS API". A status indicator shows "API Enabled".

Below the main header, there are two tabs: "OVERVIEW" (which is selected) and "DOCUMENTATION".

The main content area is a "CLOUD SHELL" terminal window titled "Terminal (qwiklabs-gcp-00-bacff2f14066)". It displays the following deployment logs:

```
target url: [https://qwiklabs-gcp-00-bacff2f14066.el.r.appspot.com]

Do you want to continue (Y/n)? Y
Beginning deployment of service [default]...
Created .cloudignore file. See 'gcloud topic gcloudignore' for details.
Uploading 5 files to Google Cloud Storage
File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://qwiklabs-gcp-00-bacff2f14066.el.r.appspot.com]

You can stream logs from the command line by running:
$ gcloud app logs tail -s default
To view your application in the web browser run:
$ gcloud app browse
student_00_ellaa8c1863c6@cloudshell:~/golang-samples/appengine/golix/helloworld (qwiklabs-gcp-00-bacff2f14066)$ gcloud app browse
Did not detect your browser. Go to this link to view your app:
https://qwiklabs-gcp-00-bacff2f14066.el.r.appspot.com
student_00_ellaa8c1863c6@cloudshell:~/golang-samples/appengine/golix/helloworld (qwiklabs-gcp-00-bacff2f14066)$ []
```

VIEW YOUR APPLICATION

Your application is deployed and you can read the short message in your browser.

Click **Check my progress** to verify the objective.



3) Firebase database service (<https://firebase.google.com/docs/database/>)

CHOOSE A DATABASE: CLOUD FIRESTORE OR REALTIME DATABASE

Firebase offers two cloud-based, client-accessible database solutions that support realtime data syncing:

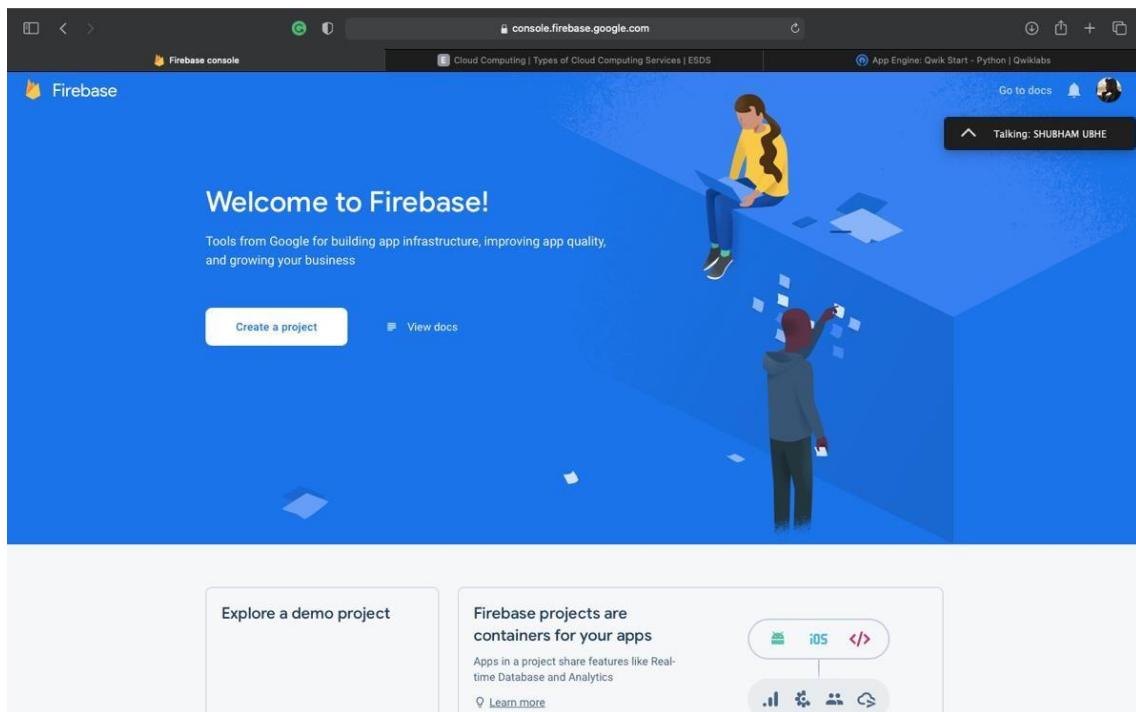
- Cloud Firestore** is Firebase's newest database for mobile app development. It builds on the successes of the Realtime Database with a new, more intuitive data model. Cloud Firestore also features richer, faster queries and scales further than the Realtime Database.
- Realtime Database** is Firebase's original database. It's an efficient, low-latency solution for mobile apps that require synced states across clients in realtime.

Firebase Realtime Database :

Store and sync data with our NoSQL cloud database. Data is synced across all clients in Realtime, and remains available when your app goes offline.

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in Realtime to every connected client. When you build cross-platform apps with our iOS, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

You can create a Firebase Project:



1) You can add firebase to an app.

A screenshot of the Firebase documentation page for 'Add Firebase to your iOS project'. The top navigation bar includes links for 'Products', 'Use Cases', 'Pricing', 'Docs', 'Community', 'Support', 'Search', 'English', 'Go to console', and a user profile. The left sidebar has sections for 'Fundamentals', 'Manage your Firebase projects', 'Platforms and frameworks', and 'Prototype and test with Emulator Suite'. The main content area shows a banner about racial equity, followed by the title 'Add Firebase to your iOS project'. It includes a 'Prerequisites' section with requirements like Xcode 12.2 or later and CocoaPods 1.10.0 or later, and a 'Getting started with Firebase on iOS' video thumbnail. A sidebar on the right lists steps for adding Firebase to an app, available pods, frameworks, and next steps.

2) You can add Firebase to an game.

The screenshot shows the Firebase documentation website for game development. The left sidebar is titled 'Fundamentals' and includes sections for 'Get started with Firebase', 'Add Firebase to an app', 'Add Firebase to a game' (which is currently selected), 'Add Firebase to a server', 'Use Firebase with a framework', 'Manage your Firebase projects', 'Platforms and frameworks', and 'Prototype and test with Emulator Suite'. The main content area has a heading 'Get started with game development using Firebase'. It explains that with Firebase, it's easy to add backend services and analytics to mobile games on iOS and Android. It features a 'Supercharge your games with Fire...' section with a yellow play button icon and the word 'Firebase'. Below this are two blue buttons: 'Get started with C++' and 'Get started with Unity'. A sidebar on the right contains a 'Table of contents' with links to 'Example use cases for Firebase in your games' and 'Supported Firebase products'.

3) You can add Firebase to a server.

The screenshot shows the Firebase documentation website for adding the Admin SDK to a server. The left sidebar is titled 'Fundamentals' and includes sections for 'Get started with Firebase', 'Add Firebase to an app', 'Add Firebase to a game', 'Add Firebase to a server' (which is currently selected), 'Use Firebase with a framework', 'Manage your Firebase projects', 'Platforms and frameworks', and 'Prototype and test with Emulator Suite'. The main content area has a heading 'Add the Firebase Admin SDK to your server'. It explains that the Admin SDK is a set of server libraries for interacting with Firebase from privileged environments. It lists several actions the Admin SDK can perform, such as reading and writing Realtime Database data, sending Firebase Cloud Messaging messages, generating and verifying auth tokens, and accessing Google Cloud resources. A sidebar on the right contains a 'Table of contents' with links to 'Prerequisites', 'Set up a Firebase project and service account', 'Add the SDK', 'Initialize the SDK', 'Using an OAuth 2.0 refresh token', 'Initialize without parameters', 'Initialize multiple apps', 'Set scopes for Realtime Database and Authentication', and 'Next steps'.

Assignment No 2

Title: Develop an application and use Google Firebase service to store your data.

Theory: Cloud Firestore is a cloud-hosted, NoSQL database that your iOS, Android, and web apps can access directly via native SDKs. Cloud Firestore is also available in native Node.js, Java, Python, Unity, C++ and Go SDKs, in addition to REST and RPC APIs.

Following Cloud Firestore's NoSQL data model, you store data in documents that contain fields mapping to values. These documents are stored in collections, which are containers for your documents that you can use to organize your data and build queries. Documents support many different [data types](#), from simple strings and numbers, to complex, nested objects. You can also create subcollections within documents and build hierarchical data structures that scale as your database grows. The Cloud Firestore [data model](#) supports whatever data structure works best for your app.

Additionally, querying in Cloud Firestore is expressive, efficient, and flexible. Create shallow queries to retrieve data at the document level without needing to retrieve the entire collection, or any nested subcollections. Add sorting, filtering, and limits to your queries or cursors to paginate your results. To keep data in your apps current, without retrieving your entire database each time an update happens, add realtime listeners. Adding realtime listeners to your app notifies you with a data snapshot whenever the data your client apps are listening to changes, retrieving only the new changes.

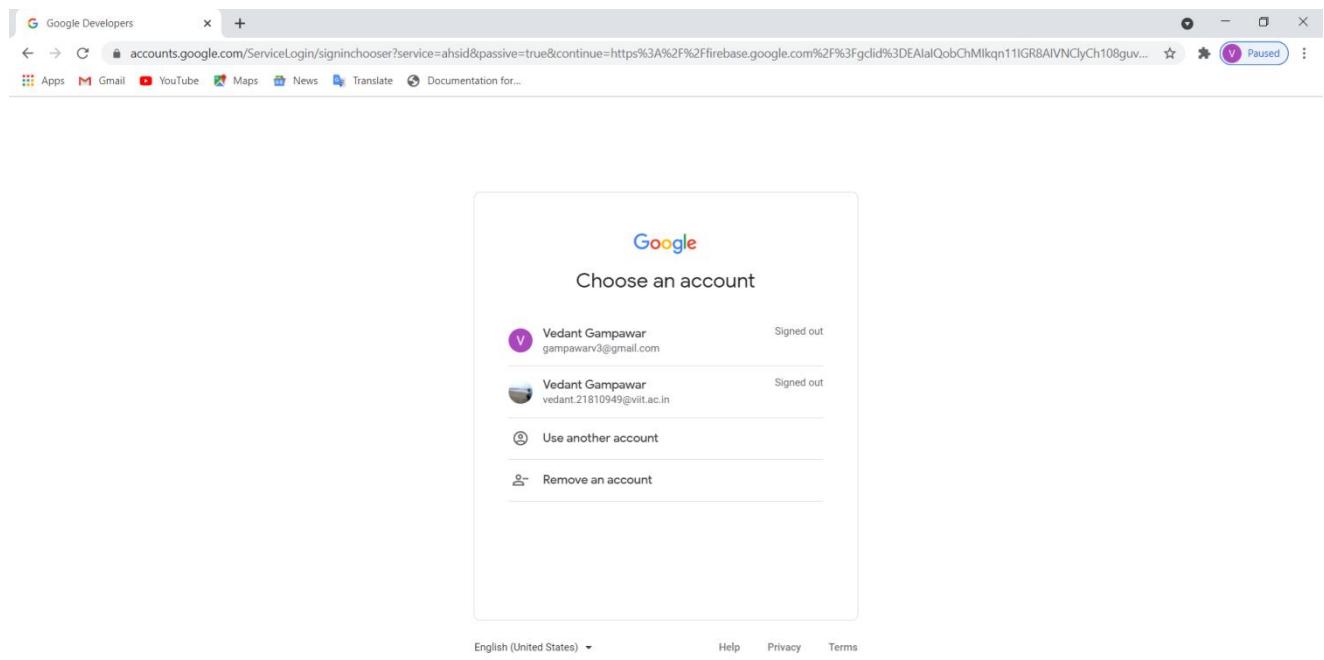
Protect access to your data in Cloud Firestore with Firebase Authentication and Cloud Firestore Security Rules for Android, iOS, and JavaScript, or Identity and Access Management (IAM) for server-side languages.

What is Firebase?

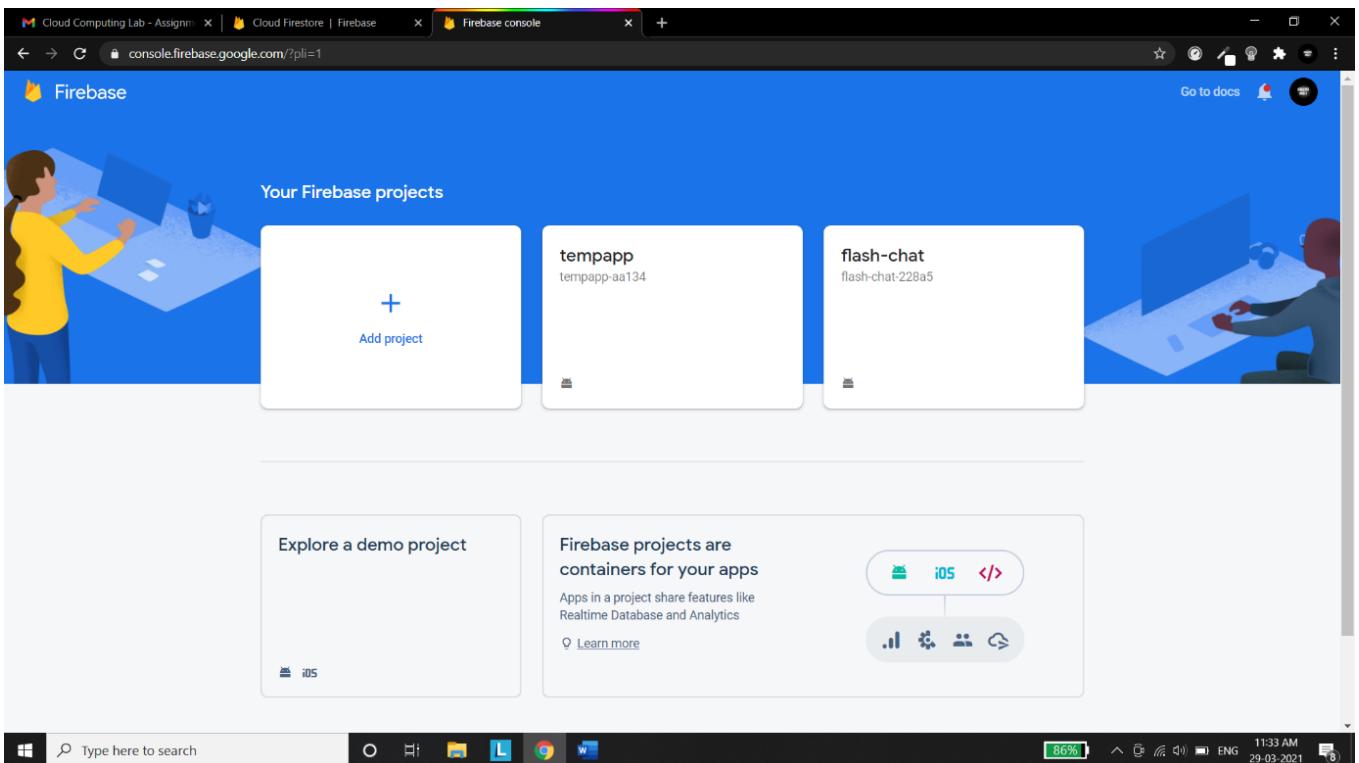
Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud. Like Firebase Realtime Database, it keeps your data in sync across client apps through realtime listeners and offers offline support for mobile and web so you can build responsive apps that work regardless of network latency or Internet connectivity. Cloud Firestore also offers seamless integration with other Firebase and Google Cloud products, including Cloud Functions.

How to use Firebase:

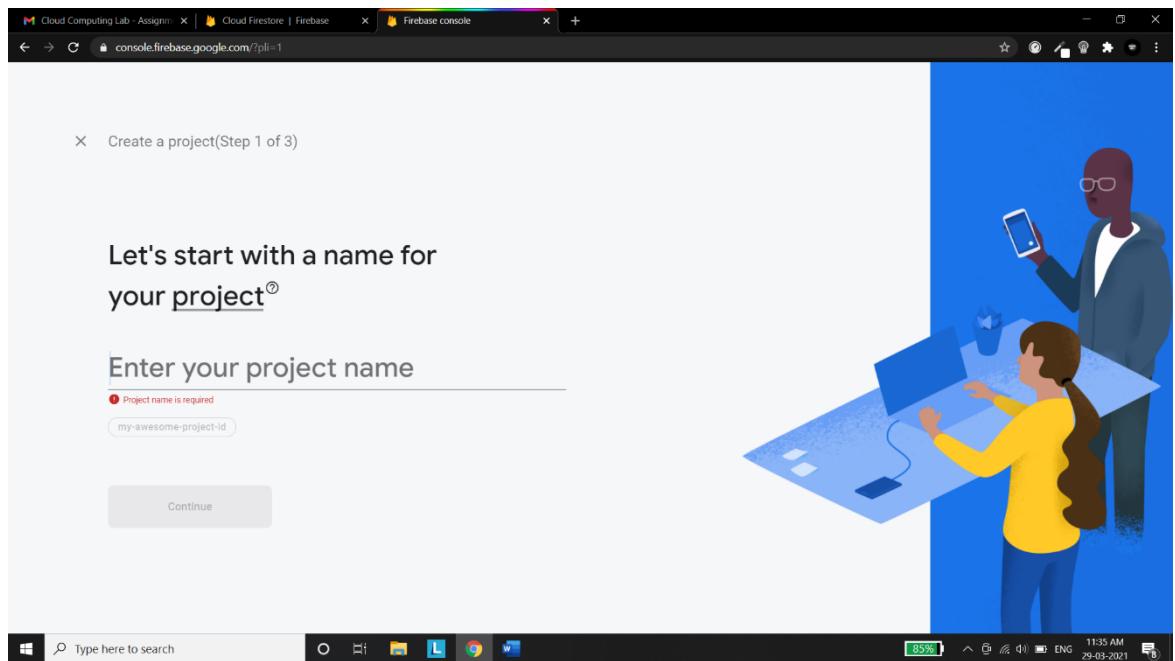
STEP 1): Sign In with Your Google account



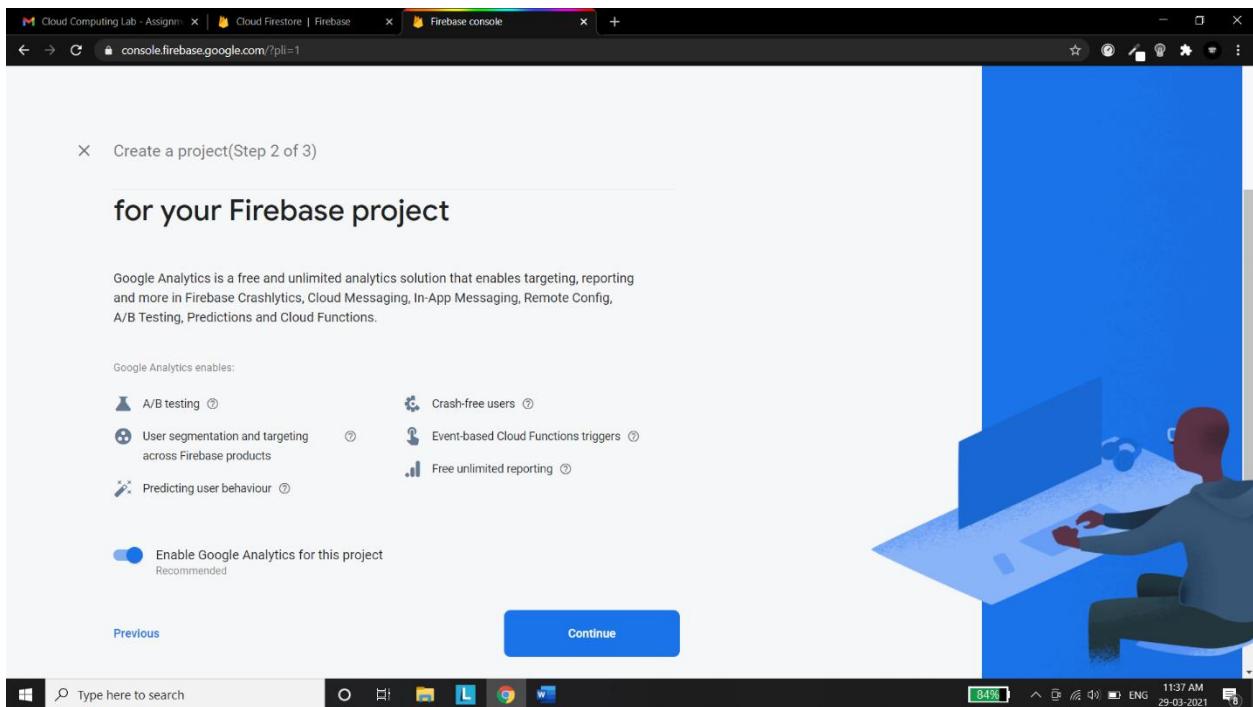
Step 2): Navigate to firebase console and add your project



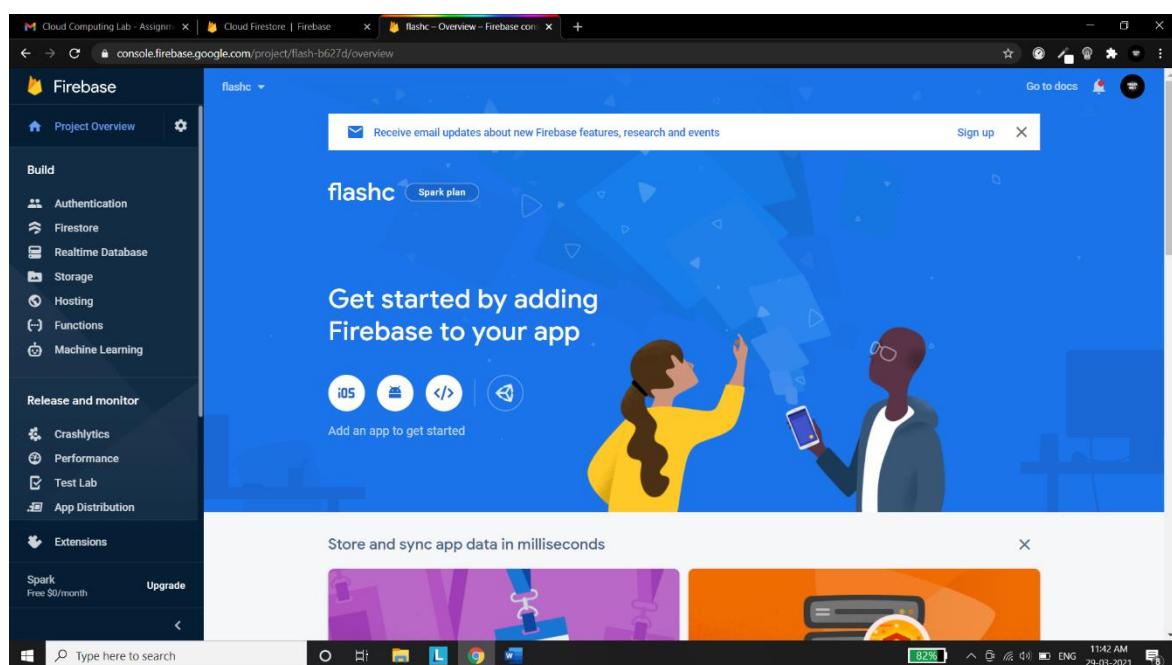
Step 3): Enter The project name



Step 4): If you want google analytics for the project then select the option or else press continue



Step 5): After Pressing on continue your project will be created now add an android or an ios or a web based app to the project . In my case I will add an android app.

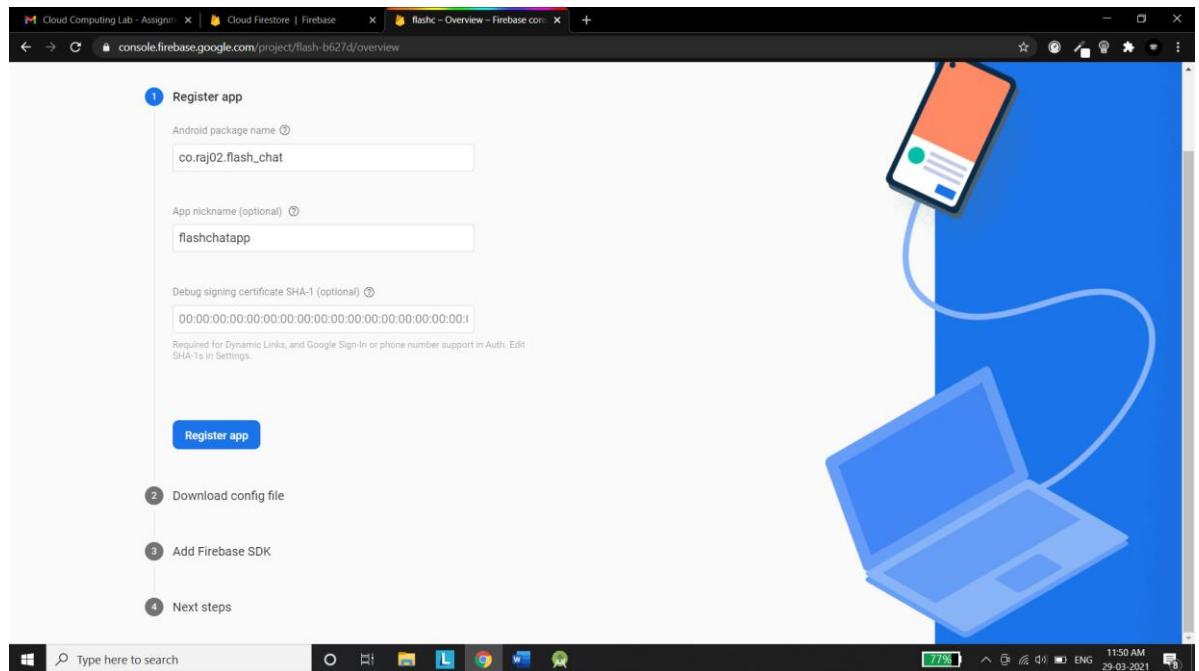


Step 6): Now you need to add an android package name , your package name is generally the application id in your app-level build.gradle file then you need to type the app nickname which is an optional field after performing the above mentioned procedure you need to press on register app button

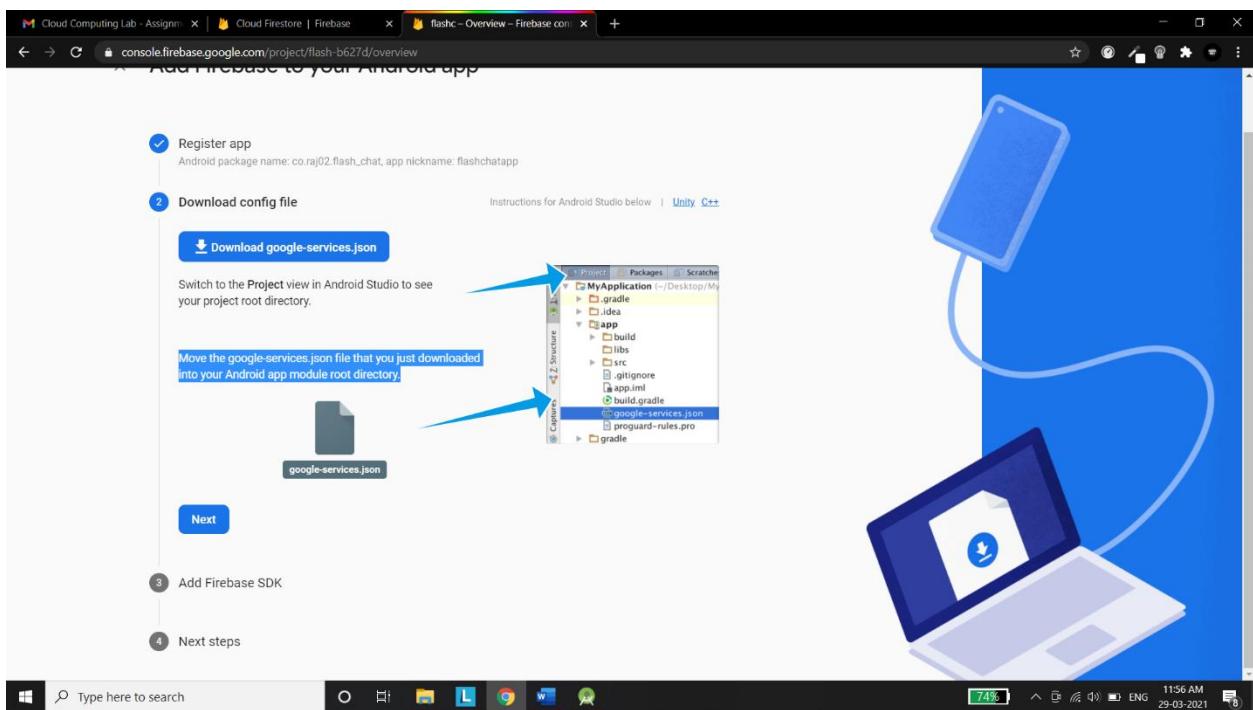
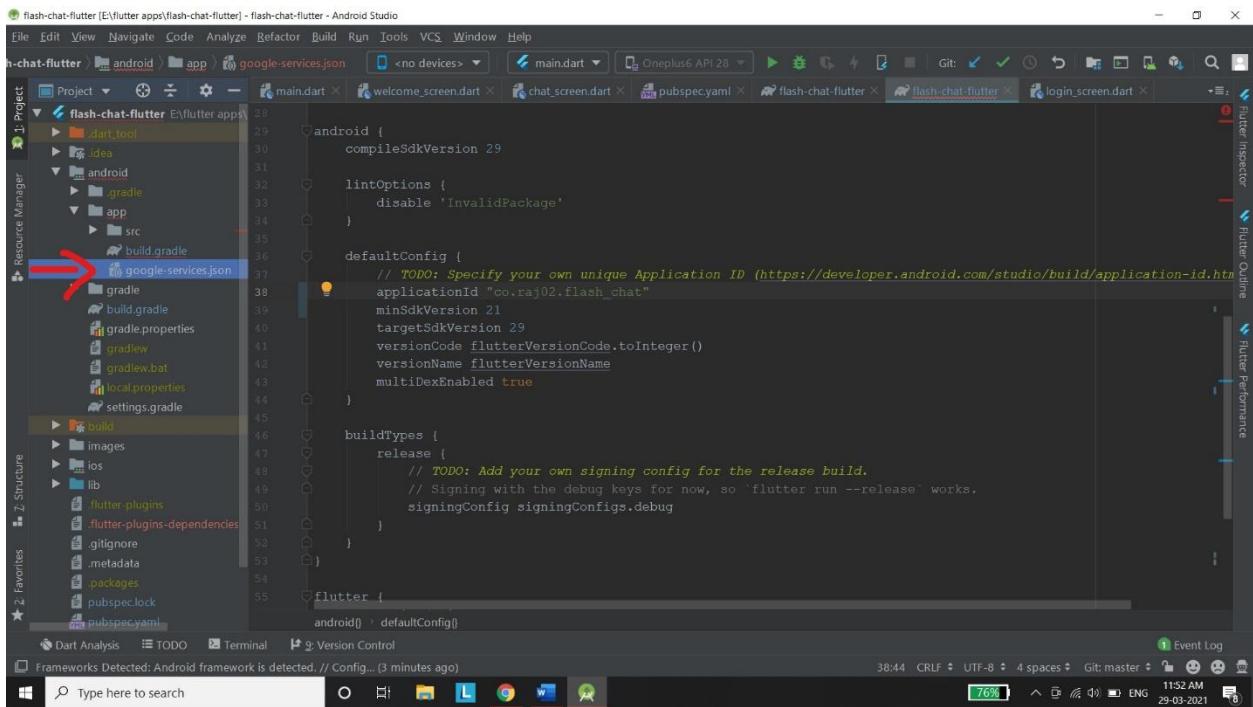
```

flash-chat-flutter [E:\flutter apps\flash-chat-flutter] - flash-chat-flutter - Android Studio
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
flash-chat-flutter > android > app > build.gradle < no devices > main.dart OnePlus API 28
Project > flash-chat-flutter E:\flutter apps\flash-chat-flutter
  +-- dart_tool
  +-- .idea
  +-- android
    +-- .gradle
    +-- app
      +-- build.gradle
      +-- google-services.json
    +-- gradle
    +-- build.gradle
    +-- gradle.properties
    +-- gradlew
    +-- gradlew.bat
    +-- local.properties
    +-- settings.gradle
  +-- build
  +-- images
  +-- ios
  +-- lib
    +-- flutter-plugins
    +-- flutter-plugins-dependencies
    +-- .gitignore
    +-- .metadata
    +-- packages
    +-- pubspec.lock
    +-- pubspec.yaml
  +-- .gitignore
  +-- .metadata
  +-- packages
  +-- pubspec.lock
  +-- pubspec.yaml
Dart Analysis TODO Terminal Version Control
Frameworks Detected: Android framework is detected. // Config... (moments ago)
Type here to search 35 chars 38:9 CRLF 4 spaces Git: master 11:49 AM 77% ENG 29-03-2021 Event Log

```



Step 7): Download the config file and move the google-services.json file that you just downloaded into your Android app module root directory and then click on next.



Step 8): Add the following lines of code in your app which are mentioned on the website and then press the next button.

Cloud Computing Lab - Assignment | Cloud Firestore | Firebase | flash - Overview - Firebase console | +

console.firebaseio.google.com/project/flash-b62fd/overview

```
Project-level build.gradle(<project>/build.gradle):
buildscript {
    repositories {
        // Check that you have the following line (if not, add it):
        google() // Google's Maven repository
    }
    dependencies {
        ...
        // Add this line
        classpath 'com.google.gms:google-services:4.3.5'
    }
}

allprojects {
    ...
    repositories {
        // Check that you have the following line (if not, add it):
        google() // Google's Maven repository
    }
}

Java (Kotlin)
```

App-level build.gradle(<project>/<app-module>/build.gradle):
apply plugin: 'com.android.application'
// Add this line
apply plugin: 'com.google.gms.google-services'

dependencies {
 // Import the Firebase BoM
 implementation platform('com.google.firebase:firebase-bom:26.8.0')

 // Add the dependency for the Firebase SDK for Google Analytics
 // When using the BoM, don't specify versions in Firebase dependencies
 implementation 'com.google.firebase:firebase-analytics'

 // Add the dependencies for any other desired Firebase products
 // https://firebase.google.com/docs/android/setup#available-libraries
}

By using the Firebase Android BoM, your app will always use compatible Firebase library versions. [Learn more](#)

flash-chat-flutter [E:\flutter\apps\flash-chat-flutter] - flash-chat-flutter - Android Studio

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

flash-chat-flutter > android > build.gradle

Project

```
buildscript {
    repositories {
        google()
        jcenter()
    }

    dependencies {
        classpath 'com.android.tools.build:gradle:3.5.3'
        classpath 'com.google.gms:google-services:4.3.5'
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

rootProject.buildDir = '../build'
subprojects {
    project.buildDir = "${rootProject.buildDir}/${project.name}"
}
subprojects {
    project.evaluationDependsOn(':app')
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Resource Manager

main.dart welcome_screen.dart chat_screen.dart pubspec.yaml flash-chat.flutter login_screen.dart

gradle build gradle.properties gradlew gradlew.bat local.properties settings.gradle

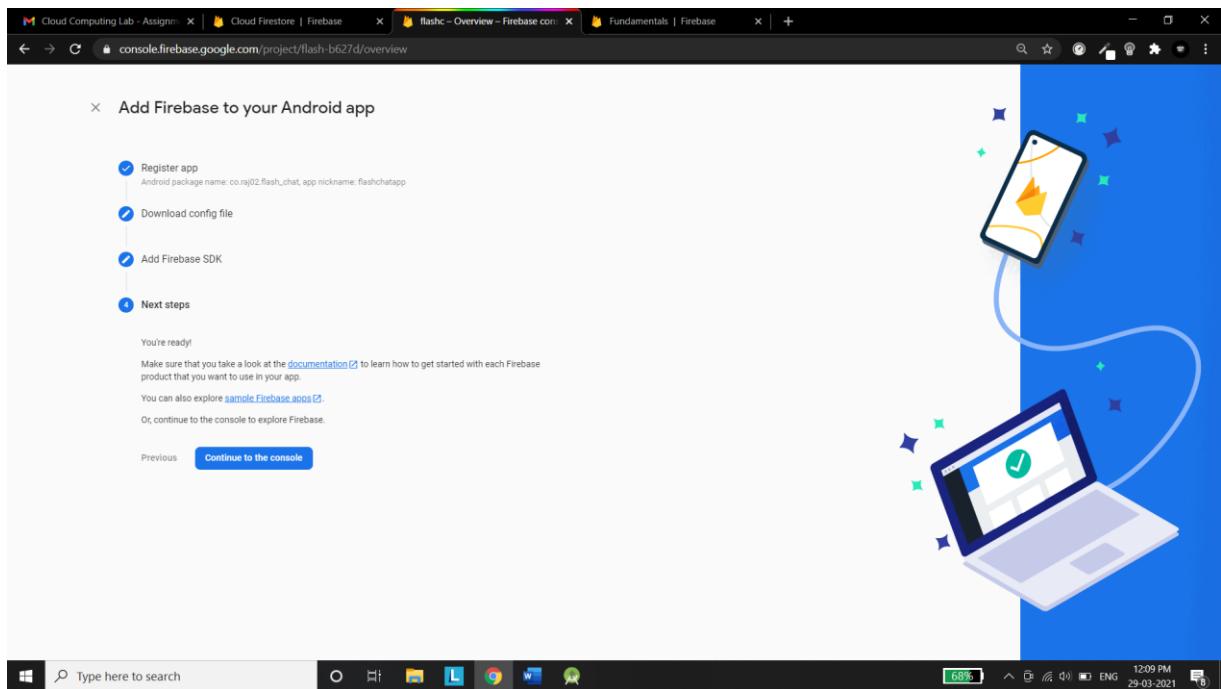
build images lib flutter_plugins .flutter-plugins-dependencies .gitignore .metadata packages pubspec.lock pubspec.yaml

Dart Analysis TODO Terminal Version Control

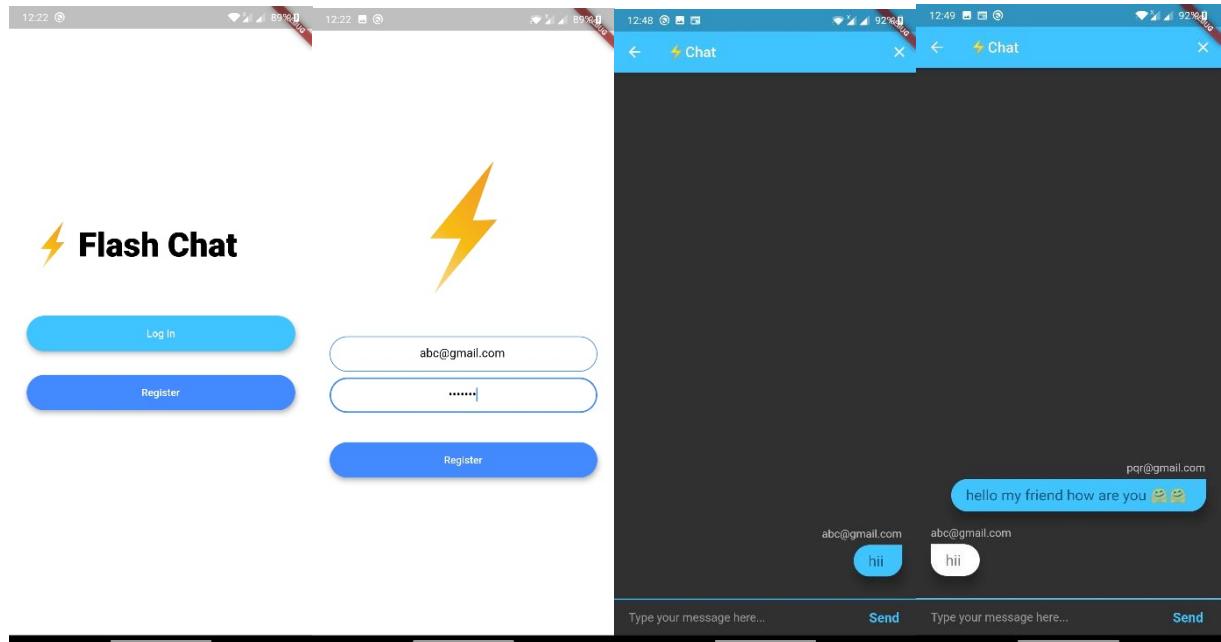
Frameworks Detected: Android framework is detected. // Config... (13 minutes ago)

```
10  def flutterRoot = localProperties.getProperty('flutter.root')
11  if (flutterRoot == null) {
12      throw new FileNotFoundException("Flutter SDK not found. Define location with flutter.sdk in the local.properties")
13  }
14
15  def flutterVersionCode = localProperties.getProperty('flutter.versionCode')
16  if (flutterVersionCode == null) {
17      flutterVersionCode = '1'
18  }
19
20  def flutterVersionName = localProperties.getProperty('flutter.versionName')
21  if (flutterVersionName == null) {
22      flutterVersionName = '1.0'
23  }
24
25  apply plugin: 'com.android.application'
26  apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"
27  //apply plugin: 'com.google.gms.google-services'
28
29  android {
30      compileSdkVersion 29
31
32      lintOptions {
33          disable 'InvalidPackage'
34      }
35
36      defaultConfig {
37          // TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html)
38          applicationId "com.vivek.flash.chat"
39      }
40  }
```

Step 9): You have now integrated firebase in your android app now click on continue to navigate to the console



You can now add code in your android app and can access the firebase services. In my app I have used the firestore and the authentication services of the firebase. Here are some screen shots from my android app.



Data Which is stored on the firestore

The screenshot shows the Firebase Cloud Firestore interface. On the left, the navigation sidebar includes sections for Authentication, Firestore, Realtime Database, Storage, Hosting, Functions, Machine Learning, Release and monitor, Analytics, Engage, and Extensions. The main area displays the Cloud Firestore interface with tabs for Data, Rules, Indexes, and Usage. A modal window is open, showing a hierarchical view of collections: messages > o9eUD0NXI23NcogwxVCu > messages. One document under 'messages' is expanded, showing fields: sender: "pqr@gmail.com" and text: "hello my friend how are you 😊😊".

The screenshot shows the Firebase Authentication interface. The left sidebar is identical to the previous screenshot. The main area displays the Authentication interface with tabs for Users, Sign-in method, Templates, and Usage. A table lists two users: abc@gmail.com and pqr@gmail.com. The table columns include Identifier, Providers, Created, Signed in, and User UID.

Identifier	Providers	Created	Signed in	User UID
abc@gmail.com	✉	29 Mar 2021	29 Mar 2021	JUJ1Hf0f09QtzTJLinqwpIAkSwG8
pqr@gmail.com	✉	29 Mar 2021	29 Mar 2021	uxYo4H8jdgZwF1GcF2zMKOoywC...

Assignment No 3

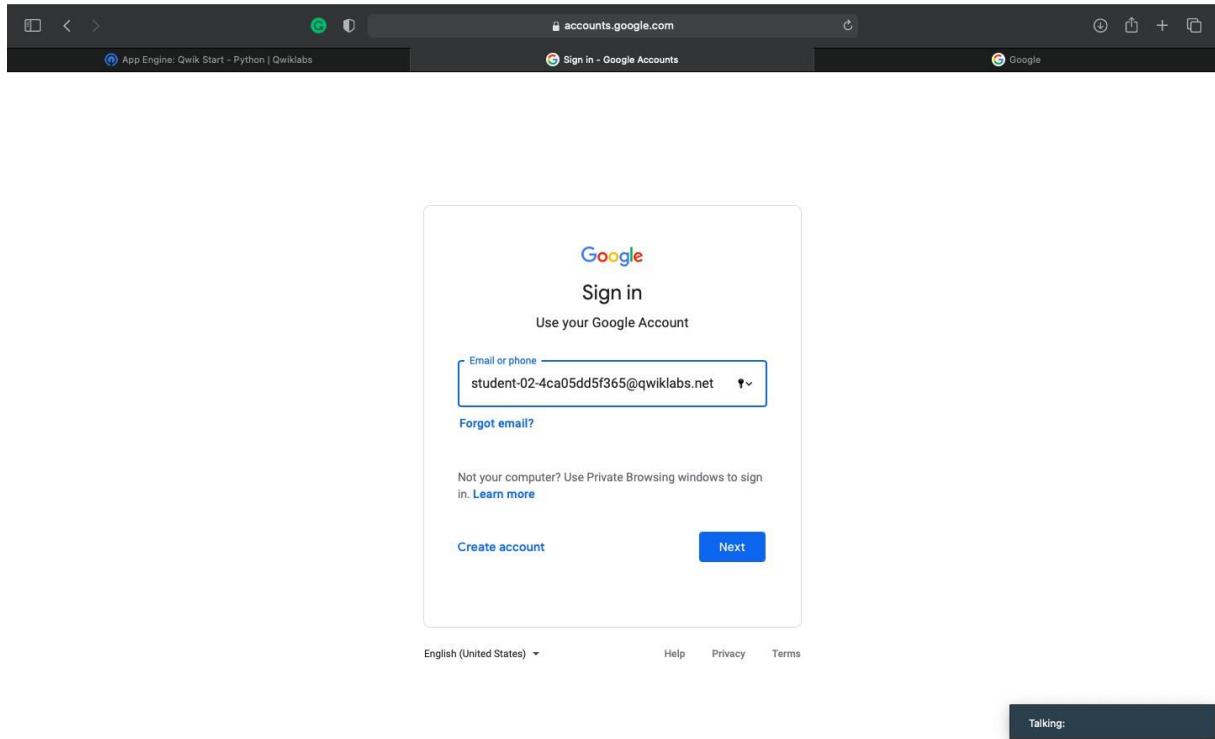
Create a sample web-based application using Python and deploy it on AWS.

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 web browser window with the following details:

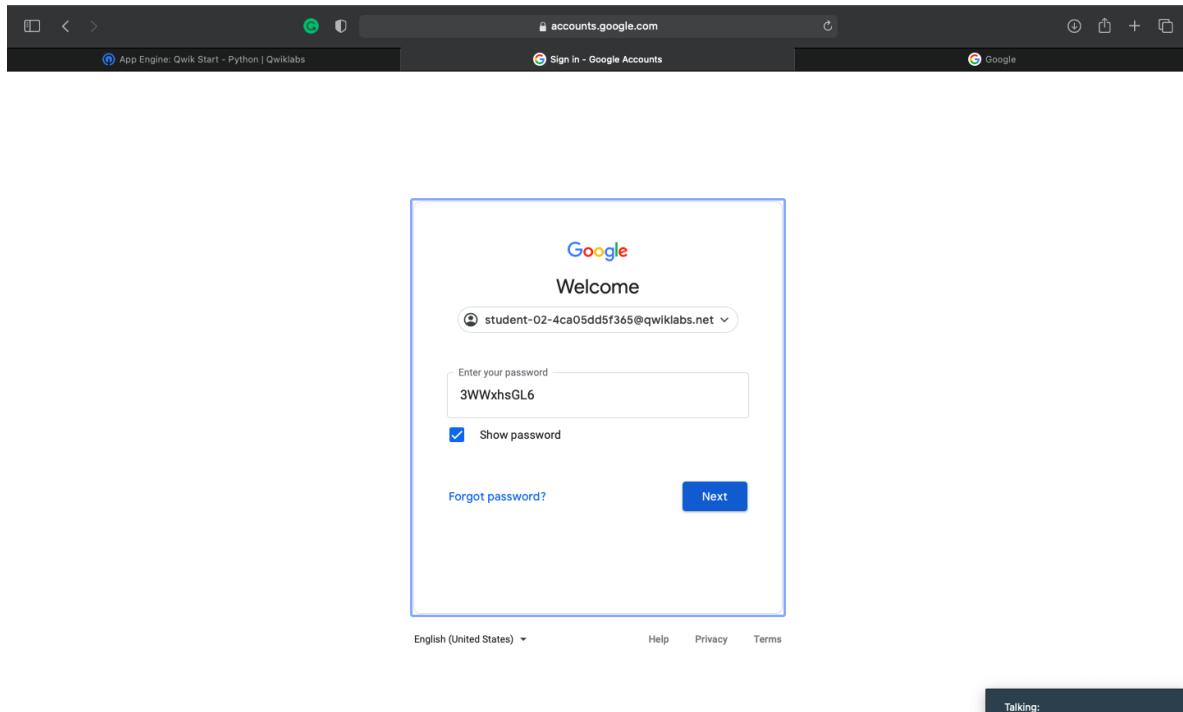
- Title Bar:** App Engine: Qwik Start - Python | Qwiklabs
- Page Content:**
 - Course Title:** App Engine: Qwik Start - Python
 - Duration:** 00:20:00
 - Student Resources:** Build Apps at Scale with Google, App Engine
 - Metrics:** 20 minutes, 1 Credit, ★★★★½
 - Navigation:** GSP067, Overview, Setup, Enable Google App Engine Admin API, Download the Hello World app, Test the application, Make a change, Deploy your app, View your application, Test your knowledge, Congratulations!
 - Logos:** GSP067, Google Cloud Self-Paced Labs
- Section:** Overview
- Description:** App Engine allows developers to focus on doing what they do best, writing code. The App Engine standard environment is based on container instances running on Google's infrastructure. Containers are preconfigured with one of several available runtimes (Java 7, Java 8, Python 2.7, Go and PHP). Each runtime also includes libraries that support [App Engine Standard APIs](#). For many applications, the standard environment runtimes and libraries might be all you need.
- Callout:** Talking: [redacted]

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.



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

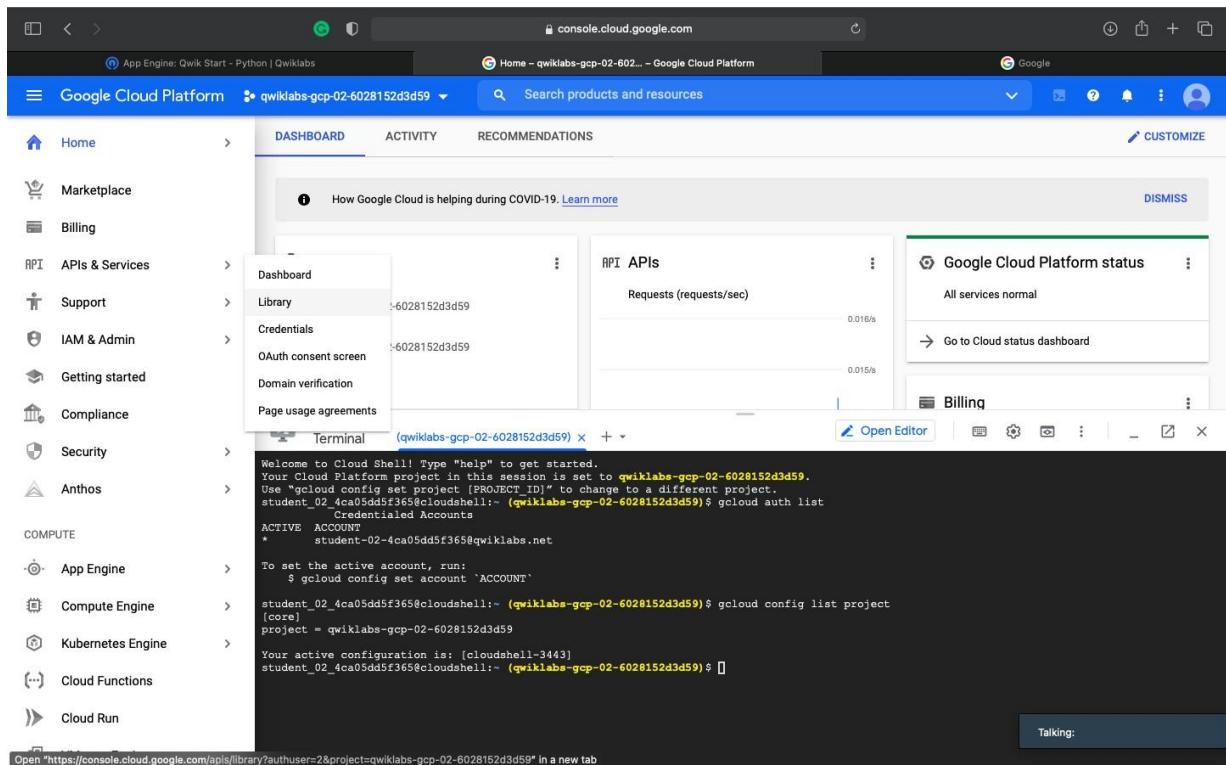
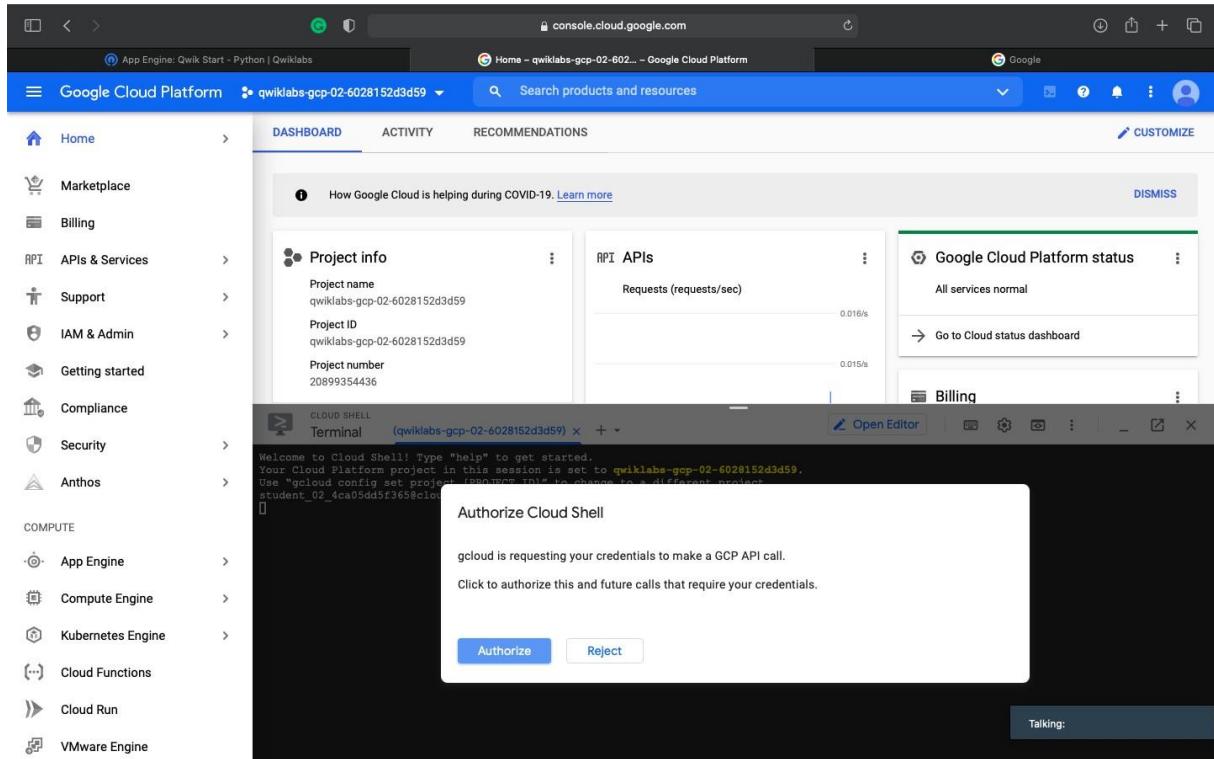
Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).



The screenshot shows the Google Cloud Platform dashboard with a modal window titled "Google Cloud Platform" open. The modal contains the message "Welcome student!" and instructions to "Create and manage your Google Cloud Platform instances, disks, networks, and other resources in one place." It includes a dropdown for "Country" set to "India", a section for "Terms of Service" with a checked checkbox, and a "Resources" section. The background dashboard shows project info (Project name: quickstart-gcp, Project ID: quickstart-gcp-02-6028152d3d59, Project number: 20899354436) and various service status cards for Status, Billing, and Monitoring.

The screenshot shows the Google Cloud Platform dashboard with the Cloud Shell terminal open. The terminal window displays the command "Terminal" and a help message: "No data is available for the selected time frame." The dashboard also features the "Project info" card and service status cards for Status, Billing, and Monitoring. A "Talking:" indicator is visible at the bottom right of the dashboard area.

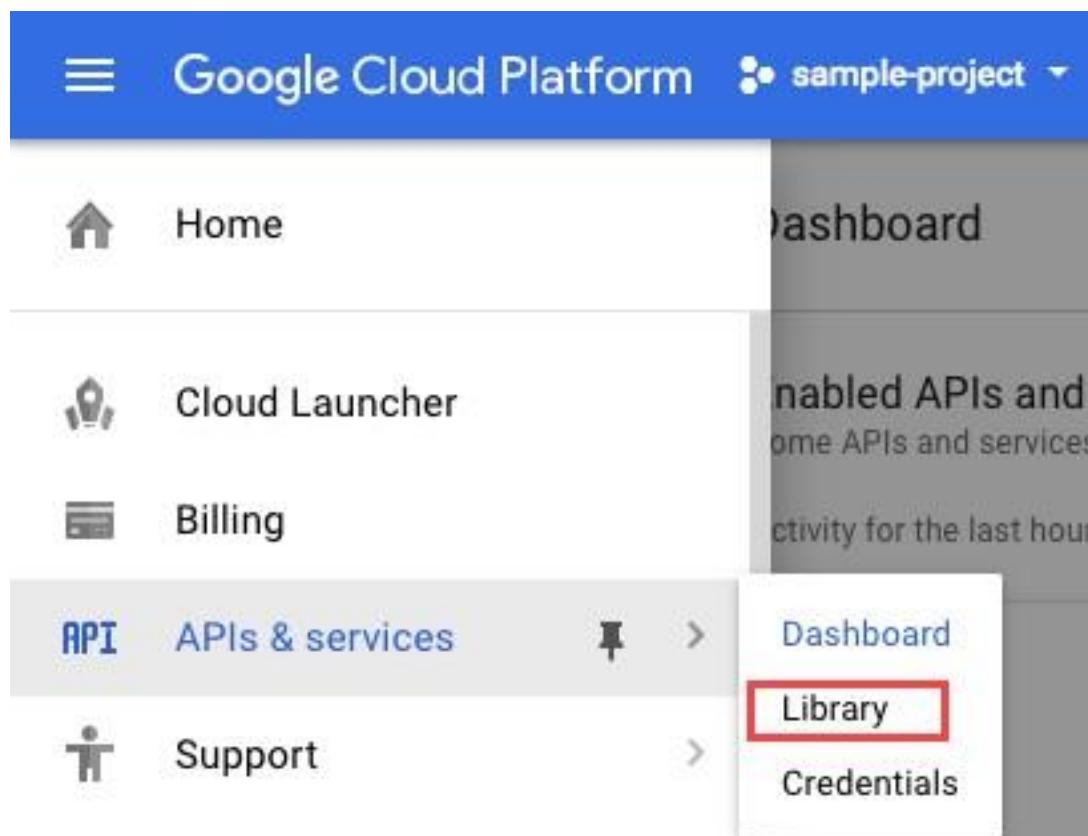
Now Activate Cloud Shell,



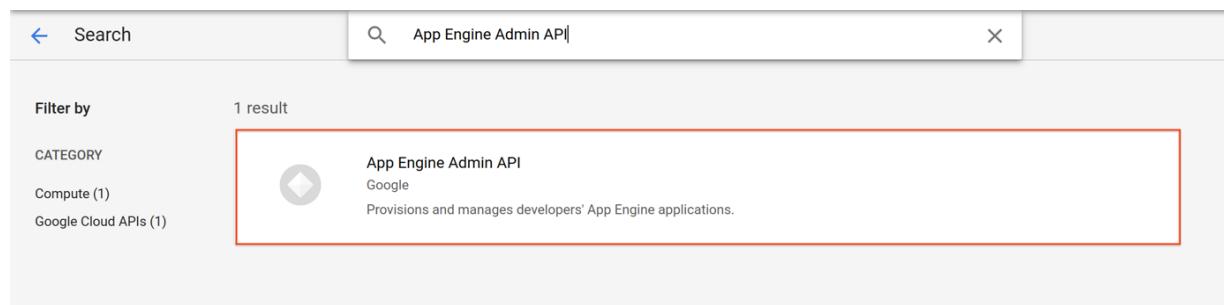
Enable Google App Engine Admin API

The App Engine Admin API enables developers to provision and manage their App Engine Applications.

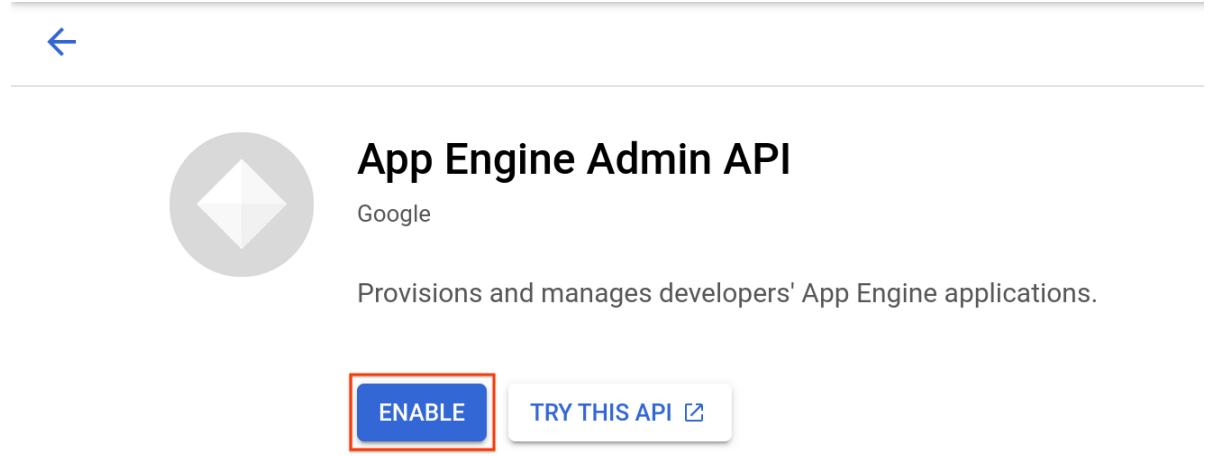
1. In the left menu click **APIs & Services > Library**.



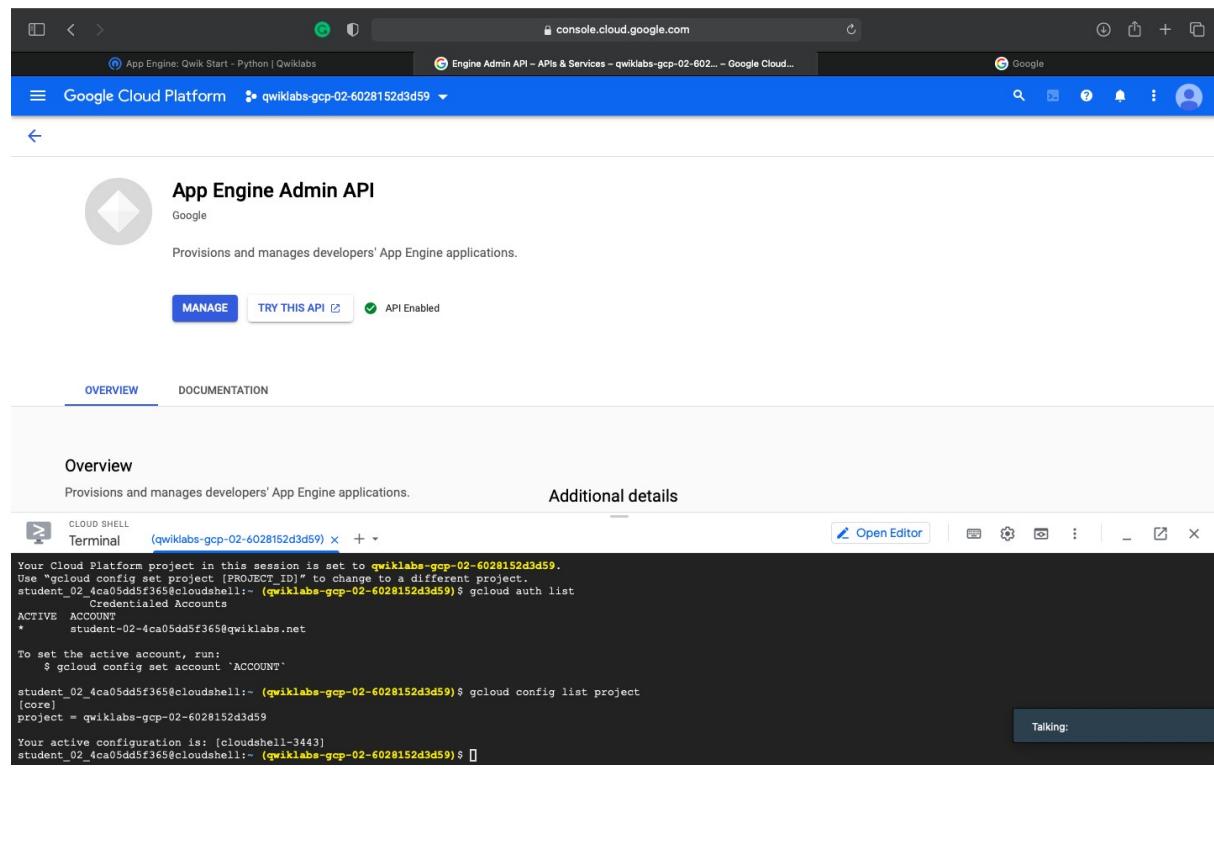
2. Type "App Engine Admin API" in search box.
3. Click **App Engine Admin API**.



- Click **Enable**. If there is no prompt to enable the API, then it is already enabled and no action is needed.



The screenshot shows the "App Engine Admin API" page from Google Cloud Platform. At the top, there's a large circular icon with a diamond shape inside, followed by the text "App Engine Admin API" and "Google". Below this, a description reads "Provisions and manages developers' App Engine applications.". Two buttons are present: a blue "ENABLE" button with a red border, and a white "TRY THIS API" button with blue text. The "ENABLE" button is highlighted with a red box.

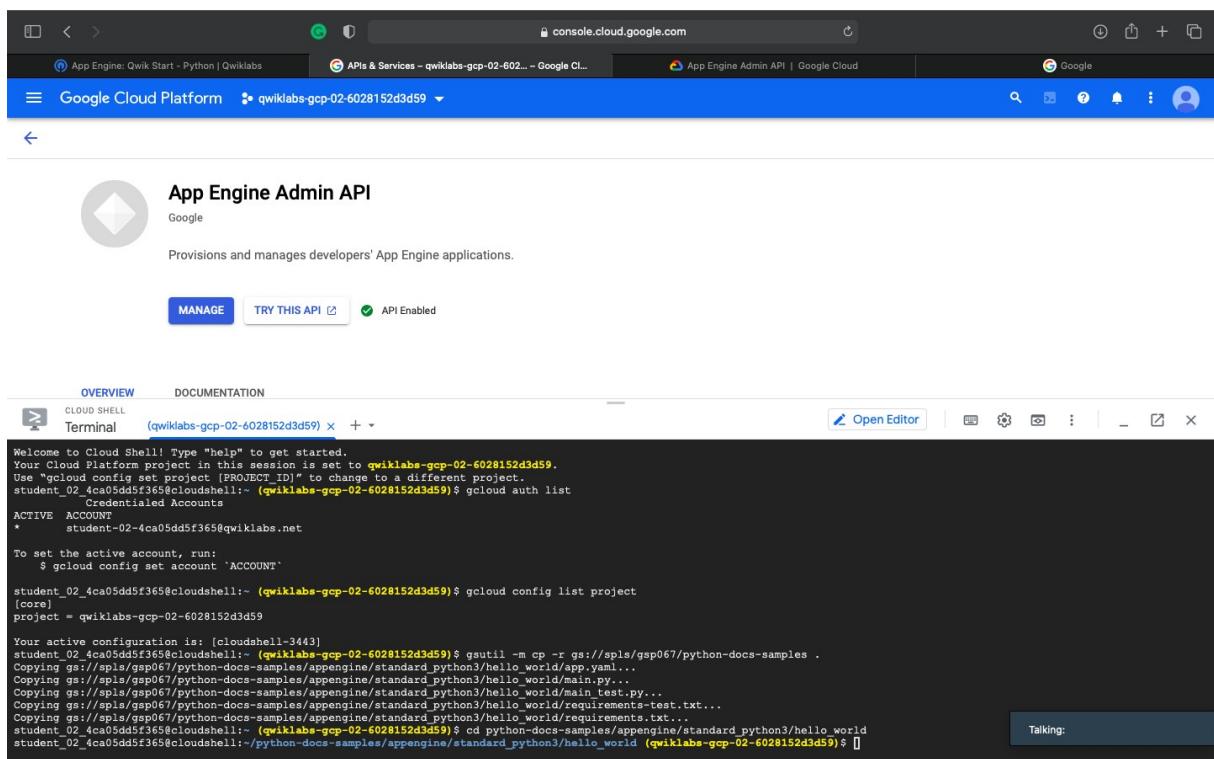


The screenshot shows the "OVERVIEW" tab of the "App Engine Admin API" page. It includes a "Cloud Shell" terminal window displaying a session for project "qwiklabs-gcp-02-6028152d3d59". The terminal output shows the user has run "gcloud auth list" and "gcloud config list project", confirming the active configuration. The "DOCUMENTATION" tab is also visible at the top.

Download the Hello World app

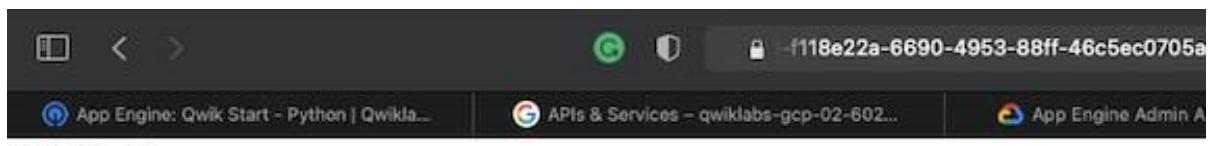
There is a simple Hello World app for Python you can use to quickly get a feel for deploying an app to Google Cloud. Follow these steps to download Hello World to your Google Cloud instance.

1. Enter the following command to copy the Hello World sample app repository to your Google Cloud instance:
2. Go to the directory that contains the sample code:



The screenshot shows the Google Cloud Platform interface with the URL `console.cloud.google.com`. The main title bar says "App Engine Admin API". Below it, there's a navigation bar with "Google Cloud Platform" and "qwiklabs-gcp-02-6028152d3d59". The main content area is titled "App Engine Admin API" and "Google". It shows a "MANAGE" button, a "TRY THIS API" button, and an "API Enabled" status. Below this, there's a "OVERVIEW" tab and a "DOCUMENTATION" tab. A "Terminal" tab is active, showing a terminal session with the following text:

```
Welcome to Cloud Shell! Type "help" to get started.  
Your Cloud Platform project in this session is set to qwiklabs-gcp-02-6028152d3d59.  
Use "gcloud config set project [PROJECT ID]" to change to a different project.  
student_02_4ca05dd5f365@cloudshell:~ (qwiklabs-gcp-02-6028152d3d59)$ gcloud auth list  
  Credentialed Accounts  
* student-02-4ca05dd5f365@qwiklabs.net  
  
To set the active account, run:  
$ gcloud config set account 'ACCOUNT'  
  
student_02_4ca05dd5f365@cloudshell:~ (qwiklabs-gcp-02-6028152d3d59)$ gcloud config list project  
[core]  
project = qwiklabs-gcp-02-6028152d3d59  
  
Your active configuration is: [cloudshell-3443]  
student_02_4ca05dd5f365@cloudshell:~ (qwiklabs-gcp-02-6028152d3d59)$ gsutil -m cp -r gs://splas/gsp067/python-docs-samples .  
Copying gs://splas/gsp067/python-docs-samples/appengine/standard_python3/hello_world/app.yaml...  
Copying gs://splas/gsp067/python-docs-samples/appengine/standard_python3/hello_world/main.py...  
Copying gs://splas/gsp067/python-docs-samples/appengine/standard_python3/hello_world/main-test.py...  
Copying gs://splas/gsp067/python-docs-samples/appengine/standard_python3/hello_world/requirements-test.txt...  
Copying gs://splas/gsp067/python-docs-samples/appengine/standard_python3/hello_world/requirements.txt...  
student_02_4ca05dd5f365@cloudshell:~ (qwiklabs-gcp-02-6028152d3d59)$ cd python-docs-samples/appengine/standard_python3/hello_world  
student_02_4ca05dd5f365@cloudshell:~/python-docs-samples/appengine/standard_python3/hello_world (qwiklabs-gcp-02-6028152d3d59)$
```



Make a change

You can leave the development server running while you develop your application. The development server watches for changes in your source files and reloads them if necessary.

Let's try it. Leave the development server running. We'll open another command line window, then edit main.py to change "Hello World!" to "Hello, Cruel World!".

1. Click the (+) next to your Cloud Shell tab to open a new command line session.



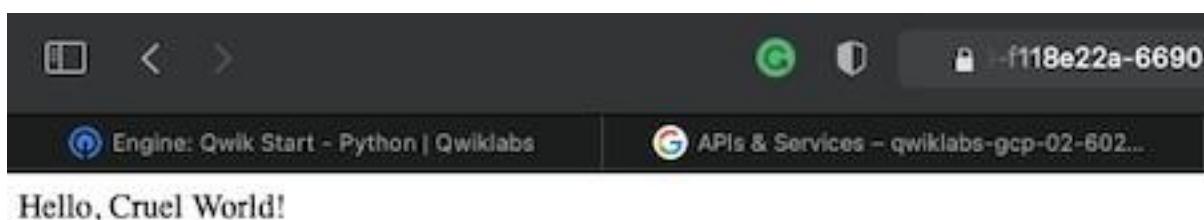
2. Enter this command to go to the directory that contains the sample code.

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

3. Enter the following to open main.py in nano to edit the content.

```
nano main.py
```

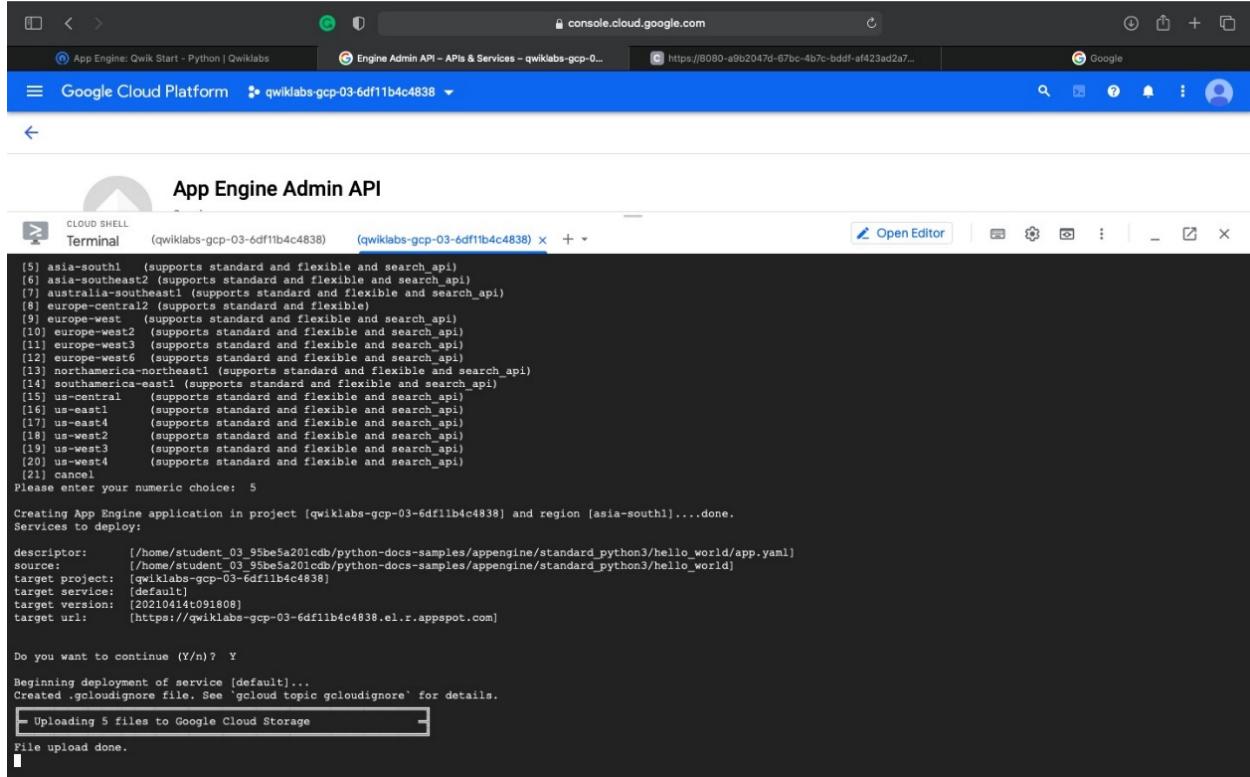
4. Change "Hello World!" to "Hello, Cruel World!". Exit and save the file.
5. Reload the Hello World! Browser or click the **Web Preview > Preview on port 8080** to see the results.



Deploy your app

To deploy your app to App Engine, run the following command from within the root directory of your application where the app.yaml file is located:

```
gcloud app deploy
```



The screenshot shows a Google Cloud Platform terminal window titled "App Engine Admin API". It displays the output of the "gcloud app deploy" command. The terminal shows a list of regions and their support for standard and flexible environments, followed by the creation of an App Engine application in the "asia-south1" region. It then asks for confirmation to continue deployment. The deployment process is shown as "Uploading 5 files to Google Cloud Storage", which has been completed.

```
(5) asia-south1 (supports standard and flexible and search_api)
(6) asia-southeast2 (supports standard and flexible and search_api)
(7) australia-southeast1 (supports standard and flexible and search_api)
(8) europe-central2 (supports standard and flexible)
(9) europe-west (supports standard and flexible and search_api)
(10) europe-west2 (supports standard and flexible and search_api)
(11) europe-west3 (supports standard and flexible and search_api)
(12) europe-west6 (supports standard and flexible and search_api)
(13) northamerica-northeast1 (supports standard and flexible and search_api)
(14) southamerica-east1 (supports standard and flexible and search_api)
(15) us-central (supports standard and flexible and search_api)
(16) us-east1 (supports standard and flexible and search_api)
(17) us-east4 (supports standard and flexible and search_api)
(18) us-west2 (supports standard and flexible and search_api)
(19) us-west3 (supports standard and flexible and search_api)
(20) us-west4 (supports standard and flexible and search_api)
[21] cancel
Please enter your numeric choice: 5

Creating App Engine application in project [qwiklabs-gcp-03-6df11b4c4838] and region [asia-south1]....done.
Services to deploy:
descriptor: [/home/student_03_55be5a201cd/python-docs-samples/appengine/standard_python3/hello_world/app.yaml]
source: [/home/student_03_55be5a201cd/python-docs-samples/appengine/standard_python3/hello_world]
target project: [qwiklabs-gcp-03-6df11b4c4838]
target service: [default]
target version: [20210414t091808]
target url: [https://qwiklabs-gcp-03-6df11b4c4838.el.r.appspot.com]

Do you want to continue (Y/n)? Y
Beginning deployment of service [default]...
Created .gcloudignore file. See 'gcloud topic gcloudignore' for details.
Uploading 5 files to Google Cloud Storage
File upload done.
```

View your application

To launch your browser enter the following command, then click on the link it provides.

```

[17] us-east4   (supports standard and flexible and search_api)
[18] us-west2   (supports standard and flexible and search_api)
[19] us-west3   (supports standard and flexible and search_api)
[20] us-west4   (supports standard and flexible and search_api)
[21] cancel

Please enter your numeric choice: 5

Creating App Engine application in project [qwiklabs-gcp-03-6df11b4c4838] and region [asia-south1]...done.
Services to deploy:
descriptor:  [/home/student_03_95be5a201cdb/python-docs-samples/appengine/standard_python3/hello_world/app.yaml]
source:      [/home/student_03_95be5a201cdb/python-docs-samples/appengine/standard_python3/hello_world]
target project: [qwiklabs-gcp-03-6df11b4c4838]
target service: [default]
target version: [20210414t091808]
target url:   [https://qwiklabs-gcp-03-6df11b4c4838.el.r.appspot.com]

Do you want to continue (Y/n)? Y
Beginning deployment of service [default]...
Created .gcloudignore file. See 'gcloud topic gcloudignore' for details.
Uploading 5 files to Google Cloud Storage
File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://qwiklabs-gcp-03-6df11b4c4838.el.r.appspot.com]
You can stream logs from the command line by running:
$ gcloud app logs tail -s default
To view your application in the web browser run:
$ gcloud app browse
student_03_95be5a201cdb@cloudshell:/python-docs-samples/appengine/standard_python3/hello_world (qwiklabs-gcp-03-6df11b4c4838)$ gcloud app browse
Did not detect your browser. Go to this link to view your app:
http://118e22a-6690.world (qwiklabs-gcp-03-6df11b4c4838) S

```

Hello, Cruel World!

Assignment 4

Salesforce

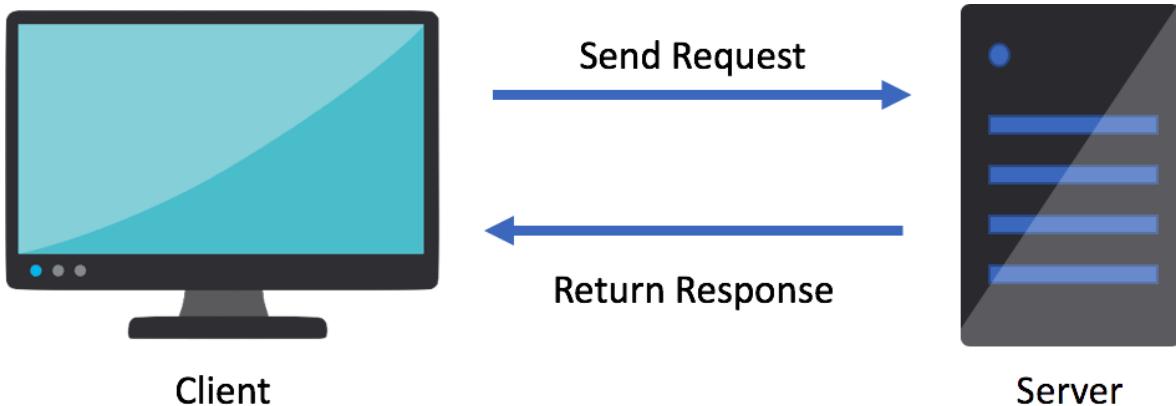
MODULE : API Basics

The screenshot shows the Salesforce Trailhead API Basics module. The page title is "API Basics | Salesforce Trailhead". The main content area features a blue header with the Trailhead logo and navigation links for Home, Learn, Credentials, Community, For Companies, and COVID-19. Below the header, there's a section titled "Module API Basics" with a sub-section "API Basics". It says "Learn the fundamentals and benefits of developing with APIs." There are three course cards listed:

- Make APIs for You and Me** - 10 mins
- Learn the Benefits of APIs** - 10 mins
- Put the Web in Web API** - 10 mins

A progress bar at the bottom indicates "-30 mins • 0%".

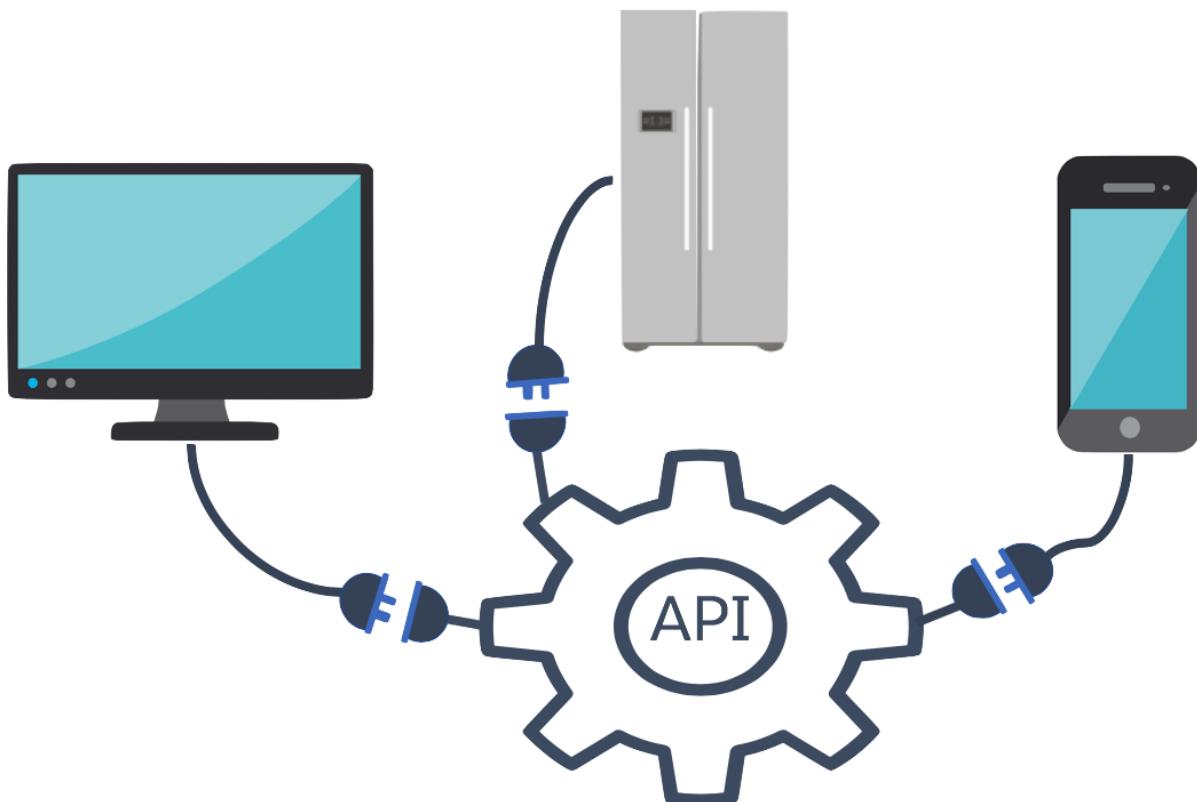
An API is equivalent to a user interface, except it's designed for software instead of humans. This is why APIs are often described in the media as technology that allows applications to talk to one another.



The client sends a request for specific information or functionality to another system. That system returns the data or functionality in a response. To send or receive data, there is an expectation that it will be in a specific format that both sides can understand. That format is often very sensitive to the context(s) it serves. Let's take a closer look.

APIs: The Electric Current Between Software

So what, if anything, do electric wall sockets have to do with APIs?



- The electricity coming from the wall is a service. It can stop and start at any time.
- The treadmill plugged into the wall uses the electricity to run.
- Since the treadmill does not have its own source of power, the treadmill is outsourcing the power it needs to a service provider, for example, to one that uses wind turbines or solar power.

While electrical sockets differ depending on where you are in the world, they have

predictable, standard patterns of openings; and the electrical plugs attached to devices like treadmills are designed to fit those standard patterns.

All these specifications essentially set expectations on behalf of any device that wants to use the service. The plugs and power supplies conform to the standard patterns and specifications (like 120 volts) for the service. The same goes for APIs.

The Benefits of Using APIs :

APIs help open up a plethora of opportunities. Here are ways that software, customers, citizen integrators, developers, and their teams can benefit from using APIs.

Outsourcing

In the name of repeatability, any compatible device (in this case, the gym equipment) can easily outsource its electrical requirements to a service, and those devices can expect to get the same results.

Similarly, APIs allow you to outsource key data and functionality through a predictable standard interface. Focus on making great applications, services, and customer experiences, not on figuring out how you're going to get common, yet nuanced information.

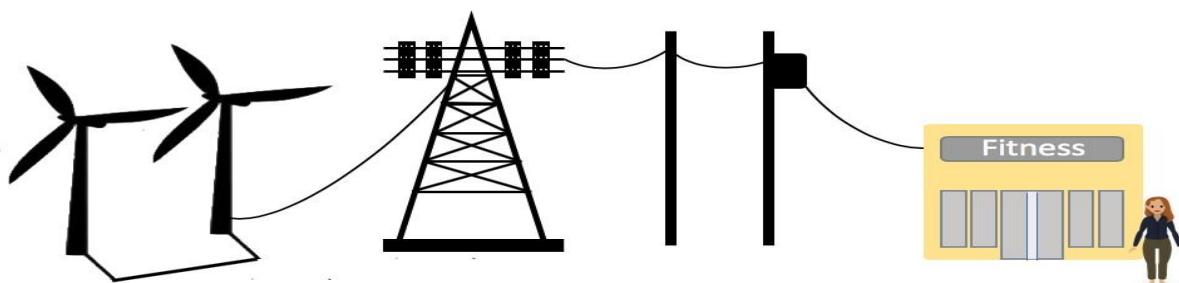
Increased Mobility

Consuming-devices are easily moved from one socket to another. For example, with no plug, matching socket or specifications, the gym owner may have to hardwire equipment into the walls of the building

Abstraction

When taking a look at the fitness club, the electrical socket is a layer of abstraction to the underlying service, or electricity. What is abstraction you ask? It is a way of hiding the working details of another system.

As long as the service delivers 120 volts of AC power to the wall socket in the standard way, the service provider is free to change anything and everything from just behind the socket all the way to the source of power. Any changes are opaque to consuming devices.



Increased Developer Productivity

When programmers write code, they rarely start from scratch. APIs are designed to take an existing code base and use it whenever, wherever instead of attempting to re-create those features.

While reusing existing code limits differentiation between applications, a reference to the API (more commonly referred to as "calling" an API) can supply the program with the expected data or functionality.

Using HTTP Protocols to Access Data

While there are no rules or laws deciding exactly how developers must connect their applications to an API, several standards have emerged.

For example, when applications connect to APIs from across the Internet, the majority of API providers make these connections available over HTTP, or hypertext transfer protocol, otherwise known as the World Wide Web.

HTTP Verbs	Descriptions
POST	Submit requested data to a server for processing
GET	Retrieve requested data from a server
PUT	Update and replace existing data with new data being sent in the request
DELETE	Remove the requested data from the server

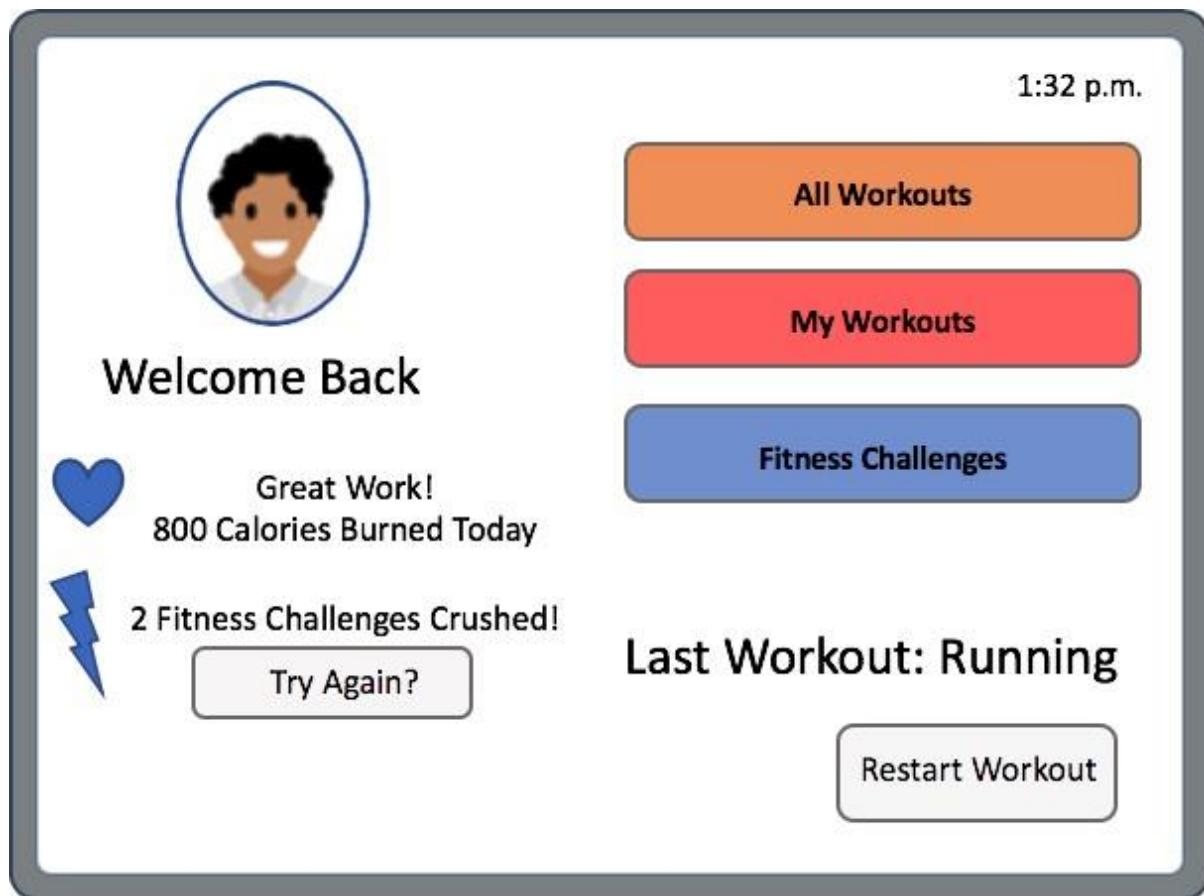
API

```
{
  "activities": [
    {
      "activityId": 51007,
      "activityParentId": 90019,
      "calories": 800,
      "description": "7mph",
      "distance": 3.04,
      "name": "Treadmill, 0% Incline",
      "startTime": "00:25",
      "steps": 6783
    }
  ],
  "fitchallenge": [
    {
      "caloriesOut": 2826,
      "distance": 8.05,
      "floors": 150,
      "steps": 10000
    },
    {
      "caloriesOut": 2021,
      "distance": 6.40,
      "floors": 80,
      "steps": 8000
    }
  ]
}
```

Of course, the above response isn't very intuitive and would never be presented to the treadmill user. It is formatted according to another standard called JSON (JavaScript Object Notation) that is often used with HTTP.

Treadmill Interface

Once the treadmill receives a response, the user can see total calories burned, how they're tracking against the most recent fitness challenge, and more.



Thanks to the thousands of APIs that software developers and non-programmers equipped with citizen integration tools (citizen integrators) can reach over the Internet (more than 23,000 by last count according to ProgrammableWeb.com), the Web has turned into a programmable platform that's equally, if not more, powerful than programmable platforms including Windows, Mac, and Linux.

2. Put the Web in Web API

Networkable APIs Are Game-Changers :

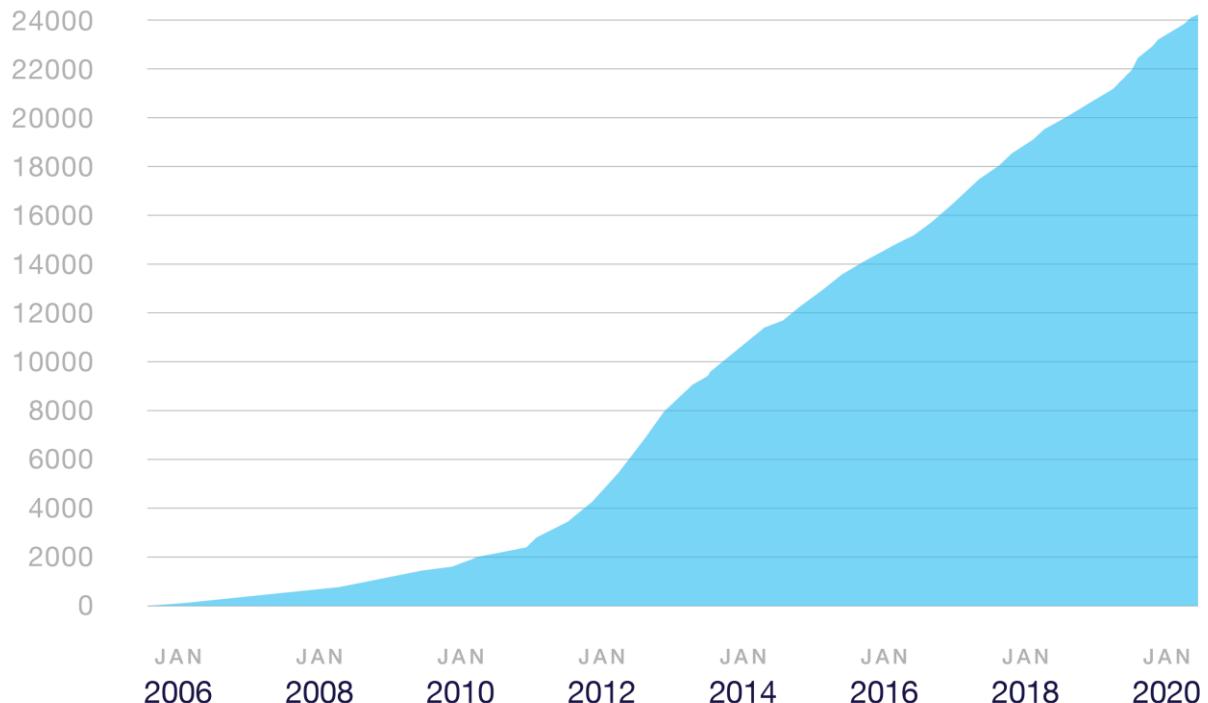
APIs aren't limited to what can be found in the same local network. Developers and citizen integrators can also consume APIs offered by remote systems and devices.

The number and types of devices that can be plugged into electrical sockets are limited only by the imaginations of inventors and the capacity of the utility.

API Economy

Depending on the volume of calls or some other way of breaking down different tiers of service, a provider like Google might charge the application developer a fee for using the API. This gives rise to the idea of an API economy.

Growth in Web APIs Since 2005



API Growth Then and Now

You may be thinking networkable APIs are the greatest thing since sliced bread. You also may be wondering, if they're so great, why didn't the tech industry come up with them earlier? As it turns out, it did.

Back in the days when Unix first came out, it was not uncommon for programmers to remotely invoke business logic from another machine across a network through a technology called RPC, or remote procedure call.

The (Possible) Future of Integration

So if history is any kind of indicator, the way we integrate between systems may be due for a change. There are now two relatively new API-like technologies that part ways with the currently favored web approach. One comes from Facebook, called GraphQL, and the other is from Google, called gRPC.

Assignment No 5

Title: create a VM using Docker and write your own docker file and publish it.

References : <https://docs.docker.com/get-started/overview/>

Note:- This assignemnt has 2 parts:

- 1) Install and use docker
- 2) Create your own image and shre it or publish it. (Image containing base : ubuntu and softwre isntalled: Apache web server)

- **What is Docker:**

Docker is a set of platform as a service (PaaS) products that use OS-level virtualization to deliver software in packages called containers. Containers are isolated from one another and bundle their own software, [libraries](#) and configuration files; they can communicate with each other through well-defined channels. All containers are run by a single operating system kernel and therefore use fewer resources than virtual machines.

- **What is the use of Docker:**

Fast, consistent delivery of your applications

Docker streamlines the development lifecycle by allowing developers to work in standardized environments using local containers which provide your applications and services. Containers are great for continuous integration and continuous delivery (CI/CD) workflows.

Consider the following example scenario:

- Your developers write code locally and share their work with their colleagues using Docker containers.
- They use Docker to push their applications into a test environment and execute automated and manual tests.
- When developers find bugs, they can fix them in the development environment and redeploy them to the test environment for testing and validation.
- When testing is complete, getting the fix to the customer is as simple as pushing the updated image to the production environment.

Responsive deployment and scaling

Docker's container-based platform allows for highly portable workloads. Docker containers can run on a developer's local laptop, on physical or virtual machines in a data center, on cloud providers, or in a mixture of environments.

Docker's portability and lightweight nature also make it easy to dynamically manage workloads, scaling up or tearing down applications and services as business needs dictate, in near real time.

Running more workloads on the same hardware

Docker is lightweight and fast. It provides a viable, cost-effective alternative to hypervisor-based virtual machines, so you can use more of your compute capacity to achieve your business goals. Docker is perfect for high density environments and for small and medium deployments where you need to do more with fewer resources.

- **Docker Architecture / Docker Engine**

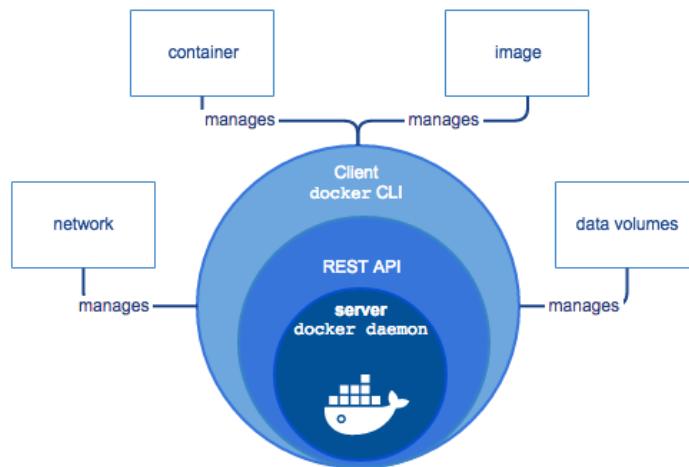


Fig: Docker Engine

Docker Engine is a client-server application with these major components:

- A server which is a type of long-running program called a daemon process (the `dockerd` command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the `docker` command).

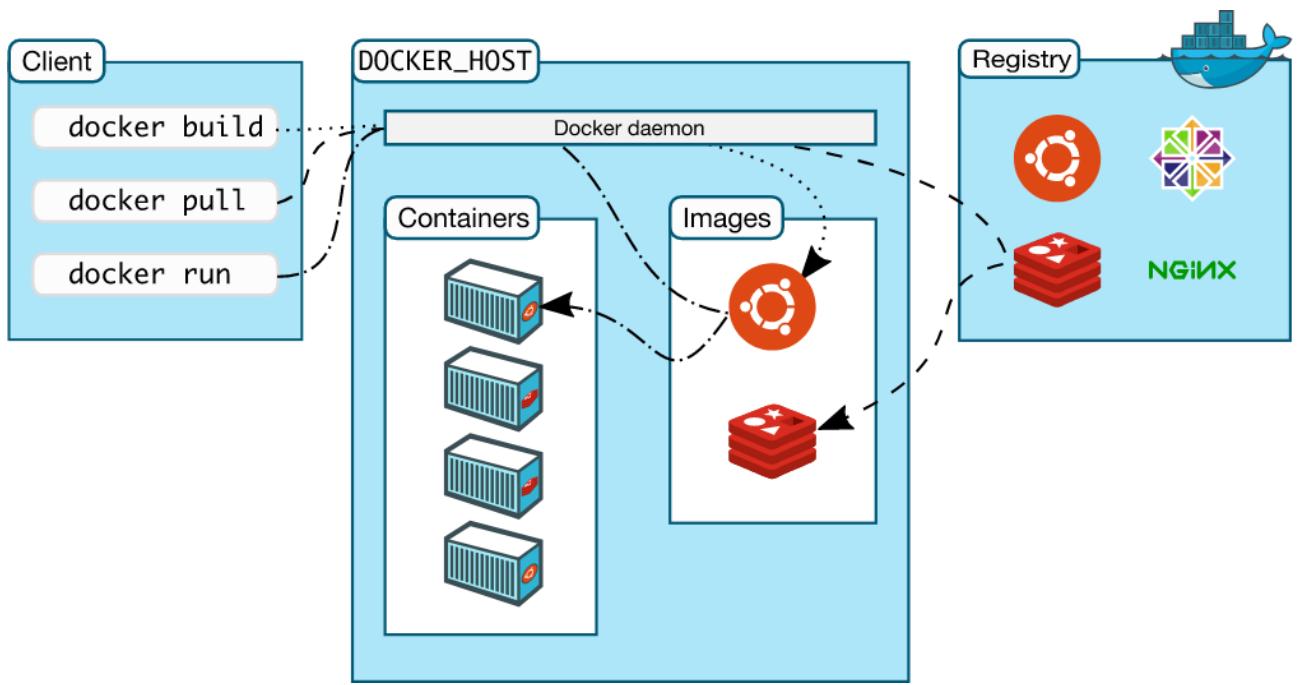


Fig: Docker Architecture

The Docker daemon

The Docker daemon (`dockerd`) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes. A daemon can also communicate with other daemons to manage Docker services.

The Docker client

The Docker client (`docker`) is the primary way that many Docker users interact with Docker. When you use commands such as `docker run`, the client sends these commands to `dockerd`, which carries them out. The `docker` command uses the Docker API. The Docker client can communicate with more than one daemon.

Docker registries

A Docker *registry* stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. You can even run your own private registry.

When you use the `docker pull` or `docker run` commands, the required images are pulled from your configured registry. When you use the `docker push` command, your image is pushed to your configured registry.

Docker objects

When you use Docker, you are creating and using images, containers, networks, volumes,

plugins, and other objects. This section is a brief overview of some of those objects.

Images

An *image* is a read-only template with instructions for creating a Docker container. Often, an image is *based on* another image, with some additional customization. For example, you may build an image which is based on the `ubuntu` image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.

You might create your own images or you might only use those created by others and published in a registry. To build your own image, you create a *Dockerfile* with a simple syntax for defining the steps needed to create the image and run it. Each instruction in a Dockerfile creates a layer in the image. When you change the Dockerfile and rebuild the image, only those layers which have changed are rebuilt. This is part of what makes images so lightweight, small, and fast, when compared to other virtualization technologies.

Containers

A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.

- **Steps to install Docker on Ubuntu** (<https://docs.docker.com/engine/install/ubuntu/>)

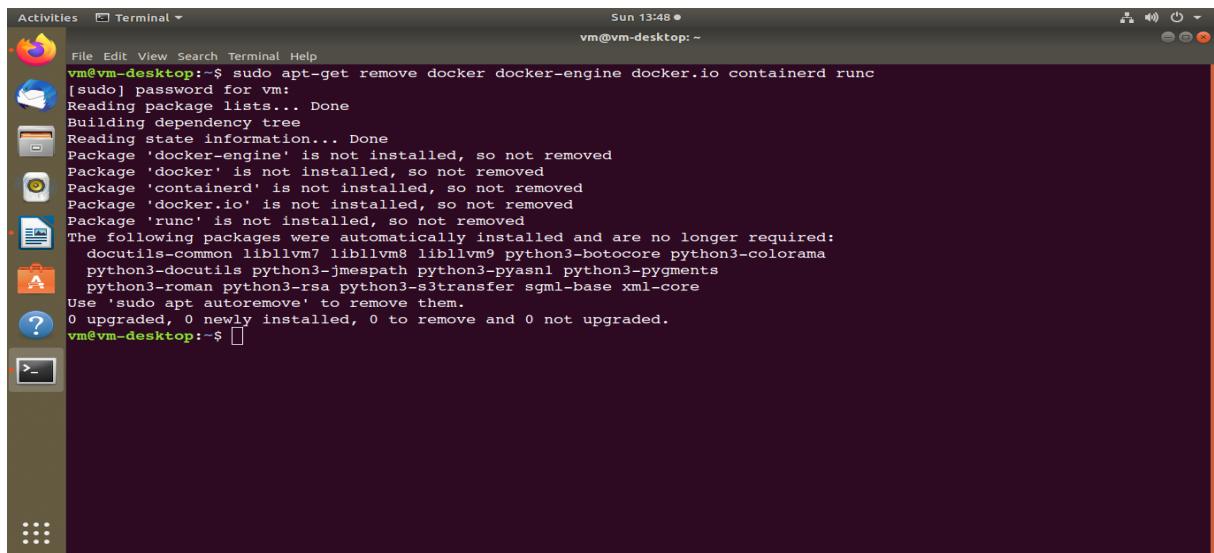
You can install docker in 3 ways:

- 1) Install using the repository (This is the recommended approach)
- 2) Install from a package
- 3) Install using the convenience script (most easiest way)

we will use first method i.e. Install using repository

Step1: Before starting installation remove the old installation if any, by using command:

```
$ sudo apt-get remove docker docker-engine docker.io containerd runc
```



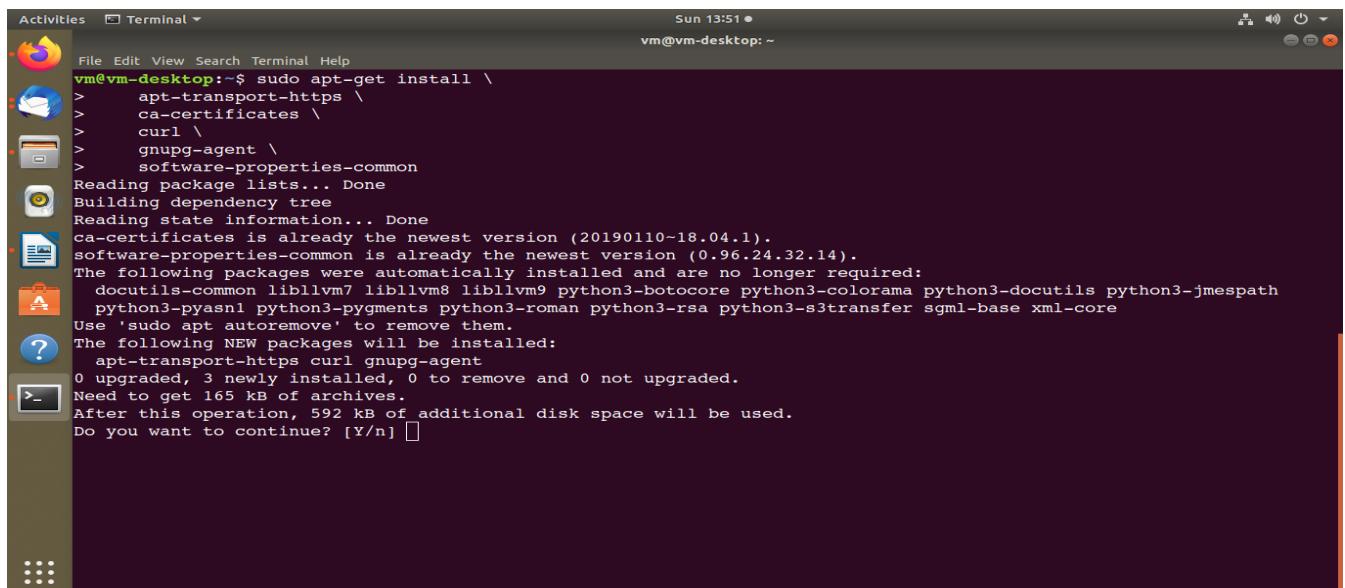
```
Activities Terminal Sun 13:48 ● vm@vm-desktop:~  
File Edit View Search Terminal Help  
vm@vm-desktop:~$ sudo apt-get remove docker docker-engine docker.io containerd runc  
[sudo] password for vm:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
Package 'docker-engine' is not installed, so not removed  
Package 'docker' is not installed, so not removed  
Package 'containerd' is not installed, so not removed  
Package 'docker.io' is not installed, so not removed  
Package 'runc' is not installed, so not removed  
The following packages were automatically installed and are no longer required:  
  docutils-common libl1vm7 libl1vm8 libl1vm9 python3-botocore python3-colorama  
  python3-docutils python3-jmespath python3-pyasn1 python3-pygments  
  python3-roman python3-rsa python3-s3transfer sgml-base xml-core  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
vm@vm-desktop:~$
```

Step 2: update the packages by using command:

```
sudo apt-get update
```

Step 3: install packages to allow apt to use a repository over HTTPS:

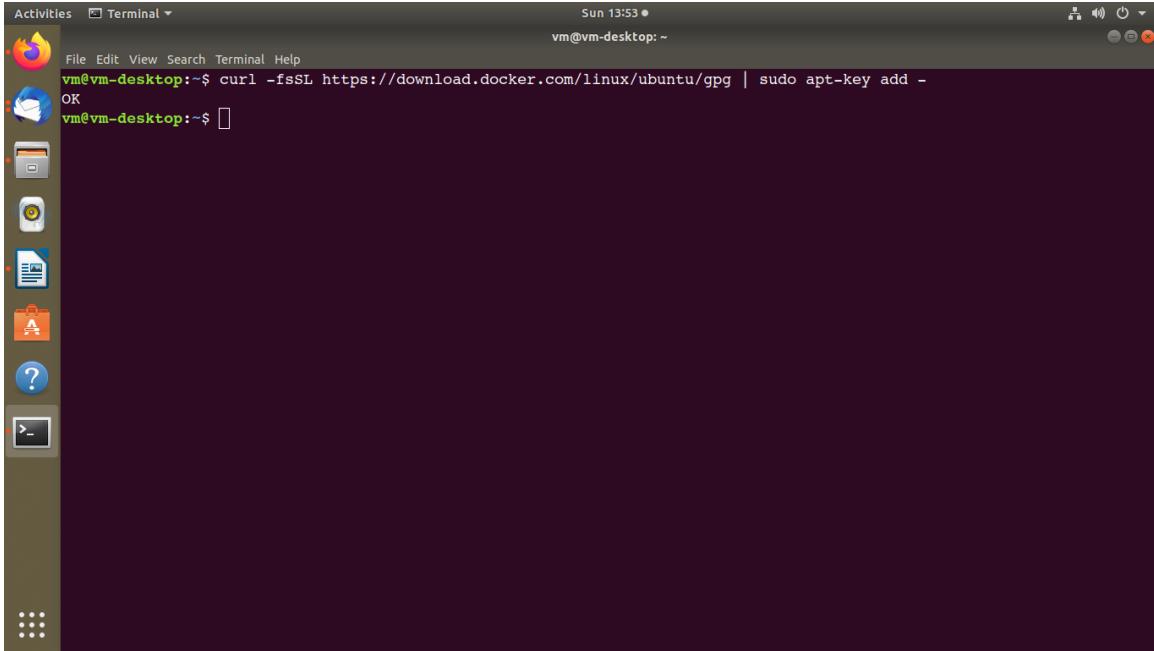
```
sudo apt-get install \  
    apt-transport-https \  
    ca-certificates \  
    curl \  
    gnupg-agent \  
    software-properties-common
```



```
Activities Terminal Sun 13:51 ● vm@vm-desktop:~  
File Edit View Search Terminal Help  
vm@vm-desktop:~$ sudo apt-get install \  
>     apt-transport-https \  
>     ca-certificates \  
>     curl \  
>     gnupg-agent \  
>     software-properties-common  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
ca-certificates is already the newest version (20190110-18.04.1).  
software-properties-common is already the newest version (0.96.24.32.14).  
The following packages were automatically installed and are no longer required:  
  docutils-common libl1vm7 libl1vm8 libl1vm9 python3-botocore python3-colorama python3-docutils python3-jmespath  
  python3-pyasn1 python3-pygments python3-roman python3-rsa python3-s3transfer sgml-base xml-core  
Use 'sudo apt autoremove' to remove them.  
The following NEW packages will be installed:  
  apt-transport-https curl gnupg-agent  
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.  
Need to get 165 kB of archives.  
After this operation, 592 kB of additional disk space will be used.  
Do you want to continue? [Y/n] 
```

Step 4: Add Docker's official GPG key:

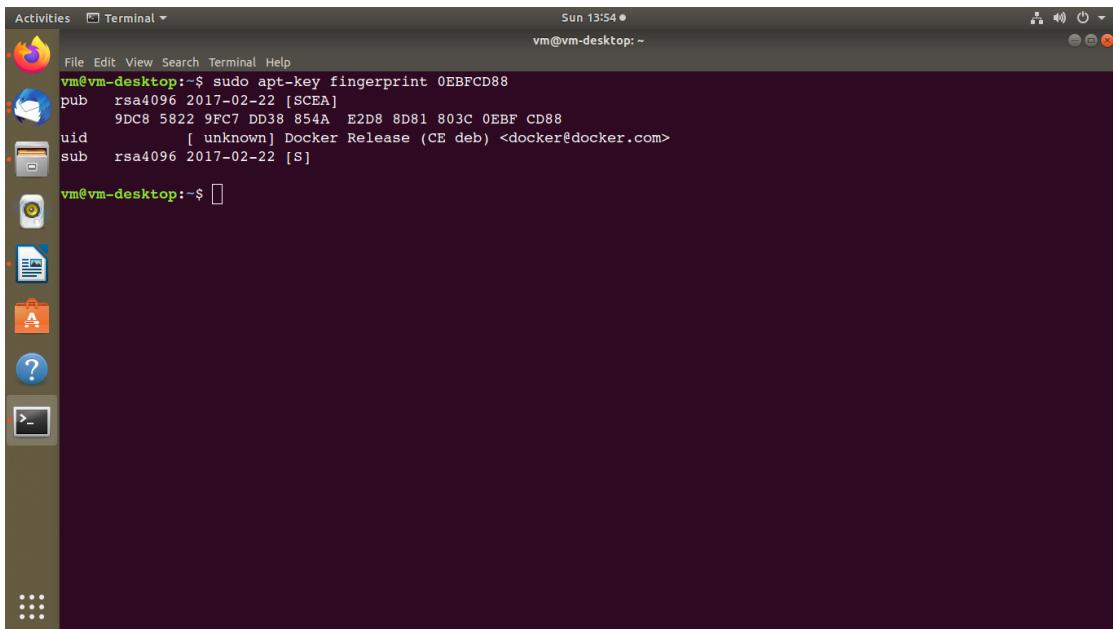
```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal shows the command `curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -` being run, followed by the output "OK". The desktop interface includes a dock with various icons like Dash, Home, and Applications.

Step 5: Verify that you now have the key with the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, by searching for the last 8 characters of the fingerprint.

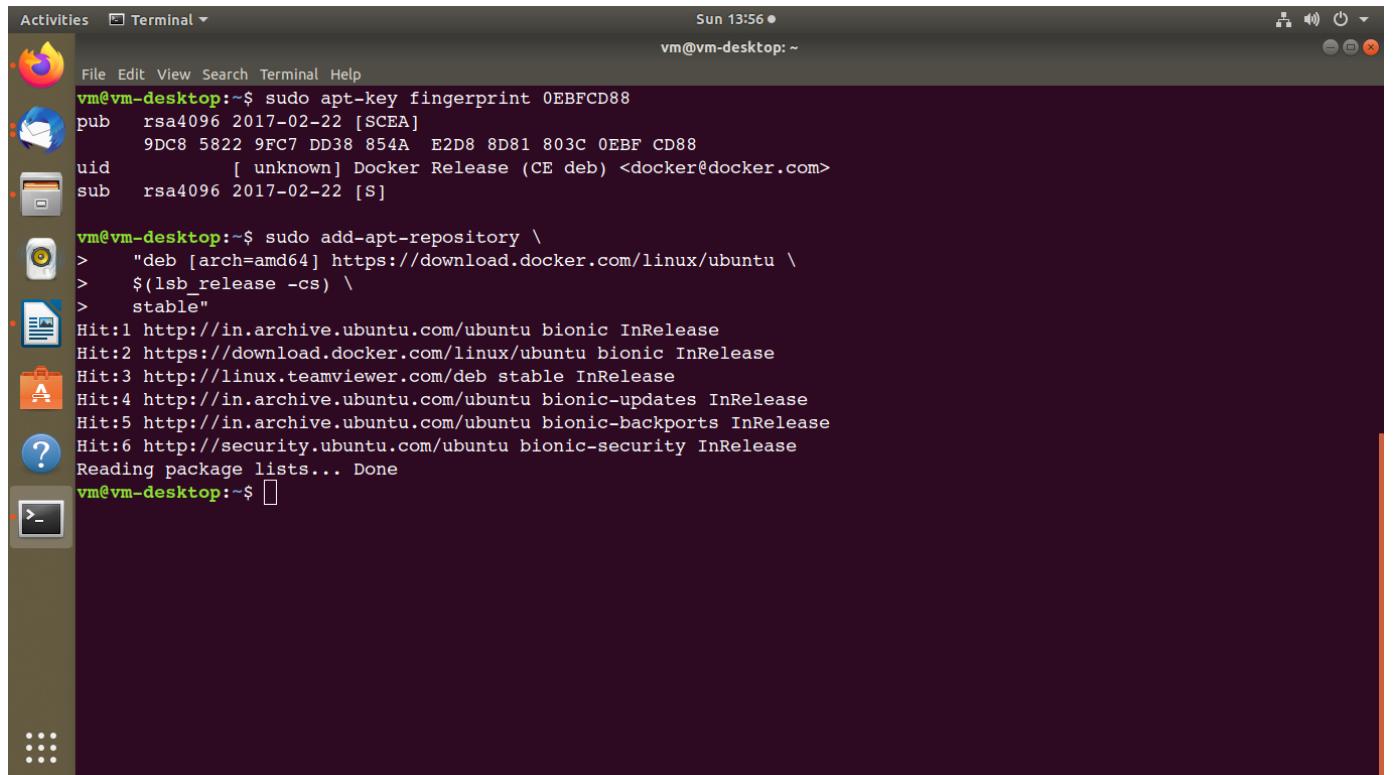
```
$ sudo apt-key fingerprint 0EBFCD88
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal shows the command `sudo apt-key fingerprint 0EBFCD88` being run, followed by the output of the key details, including the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88. The desktop interface includes a dock with various icons like Dash, Home, and Applications.

Step 6: Use the following command to set up the **stable** repository. To add the **nightly** or **test** repository, add the word nightly or test (or both) after the word stable in the commands below.

```
$ sudo add-apt-repository \
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) \
  stable"
```

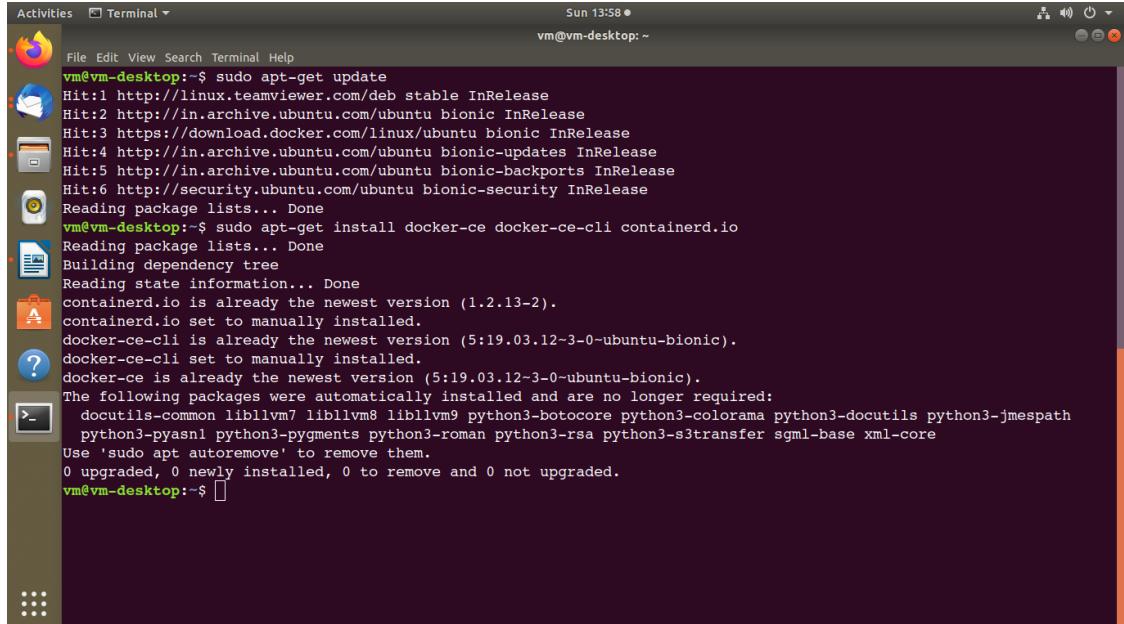


A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window is titled 'Terminal' and has the command 'sudo apt-key fingerprint 0EBFCD88' running. The output shows a public key fingerprint for Docker's release key. Below this, the command 'sudo add-apt-repository' is run again with the Docker repository URL and the current Ubuntu release code name ('bionic'). The terminal then lists several hits from package indexes, including 'in.archive.ubuntu.com/ubuntu', 'download.docker.com/linux/ubuntu', and 'linux.teamviewer.com/deb'. Finally, it shows 'Reading package lists... Done' and ends with the prompt 'vm@vm-desktop:~\$'. The desktop interface includes a dock with icons for various applications like a browser, file manager, and terminal, and a system tray at the top.

Install Docker Engine

Step 7: Update the apt package index, and install the *latest version* of Docker Engine and containerd, or go to the next step to install a specific version:

```
$ sudo apt-get update  
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

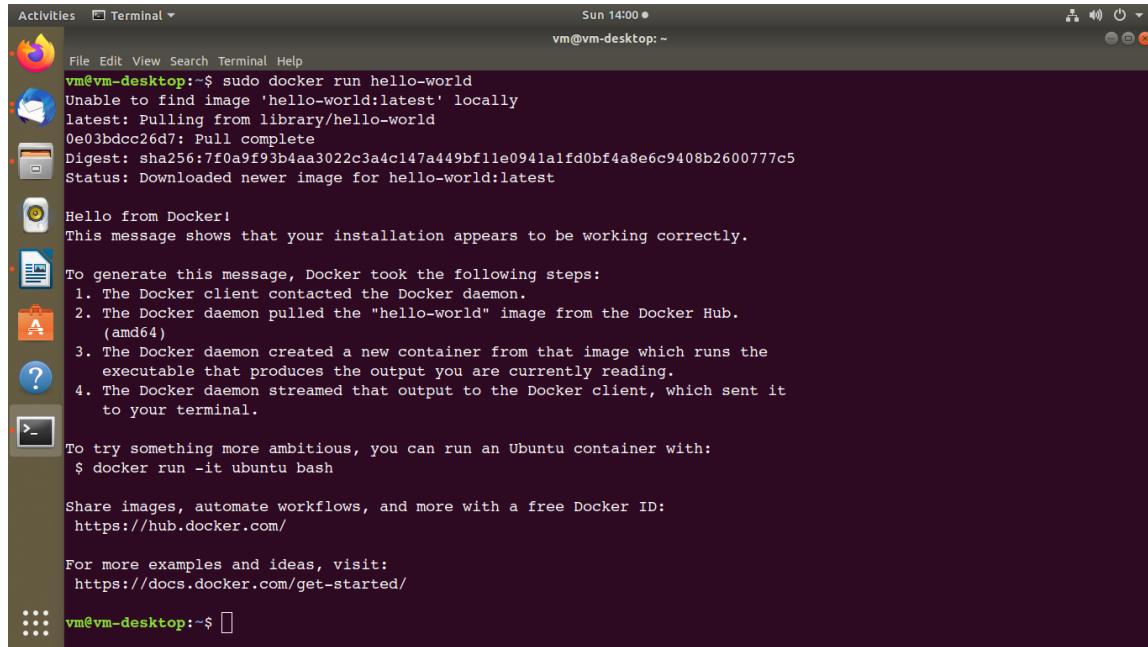


A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal output shows the execution of the following commands:

```
vm@vm-desktop:~$ sudo apt-get update  
Hit:1 http://linux.teamviewer.com/deb stable InRelease  
Hit:2 http://in.archive.ubuntu.com/ubuntu bionic InRelease  
Hit:3 https://download.docker.com/linux/ubuntu bionic InRelease  
Hit:4 http://in.archive.ubuntu.com/ubuntu bionic-updates InRelease  
Hit:5 http://in.archive.ubuntu.com/ubuntu bionic-backports InRelease  
Hit:6 http://security.ubuntu.com/ubuntu bionic-security InRelease  
Reading package lists... Done  
vm@vm-desktop:~$ sudo apt-get install docker-ce docker-ce-cli containerd.io  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
containerd.io is already the newest version (1.2.13-2).  
containerd.io set to manually installed.  
docker-ce-cli is already the newest version (5:19.03.12-3-0-ubuntu-bionic).  
docker-ce-cli set to manually installed.  
docker-ce is already the newest version (5:19.03.12-3-0-ubuntu-bionic).  
The following packages were automatically installed and are no longer required:  
  docutils-common libl1vm7 libl1vm8 libl1vm9 python3-botocore python3-colorama python3-docutils python3-jmespath  
  python3-pyasn1 python3-pymgments python3-roman python3-rsa python3-s3transfer sgml-base xml-core  
Use 'sudo apt autoremove' to remove them.  
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.  
vm@vm-desktop:~$
```

Step 8: Verify that Docker Engine is installed correctly by running the hello-world image.

```
$ sudo docker run hello-world
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal output shows the execution of the command:

```
vm@vm-desktop:~$ sudo docker run hello-world  
Unable to find image 'hello-world:latest' locally  
latest: Pulling from library/hello-world  
0e03bcc26d7: Pull complete  
Digest: sha256:7f0a9f93b4aa3022c3a4c147a449bf11e0941a1fd0bf4a8e6c9408b2600777c5  
Status: Downloaded newer image for hello-world:latest  
  
Hello from Docker!  
This message shows that your installation appears to be working correctly.  
  
To generate this message, Docker took the following steps:  
 1. The Docker client contacted the Docker daemon.  
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.  
     (amd64)  
 3. The Docker daemon created a new container from that image which runs the  
     executable that produces the output you are currently reading.  
 4. The Docker daemon streamed that output to the Docker client, which sent it  
     to your terminal.  
  
To try something more ambitious, you can run an Ubuntu container with:  
 $ docker run -it ubuntu bash  
  
Share images, automate workflows, and more with a free Docker ID:  
 https://hub.docker.com/  
  
For more examples and ideas, visit:  
 https://docs.docker.com/get-started/  
vm@vm-desktop:~$
```

Uninstall Docker Engine

Step 1: Uninstall the Docker Engine, CLI, and Containerd packages:

```
$ sudo apt-get purge docker-ce docker-ce-cli containerd.io
```

Step 2: Images, containers, volumes, or customized configuration files on your host are not automatically removed. To delete all images, containers, and volumes:

```
$ sudo rm -rf /var/lib/docker
```

Docker Compose

Compose is a tool for defining and running multi-container Docker applications. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration. To learn more about all the features of Compose, see the [list of features](#).

Compose works in all environments: production, staging, development, testing, as well as CI workflows. You can learn more about each case in [Common Use Cases](#).

Using Compose is basically a three-step process:

1. Define your app's environment with a Dockerfile so it can be reproduced anywhere.
 2. Define the services that make up your app in docker-compose.yml so they can be run together in an isolated environment.
 3. Run docker-compose up and Compose starts and runs your entire app
-

Docker Desktop

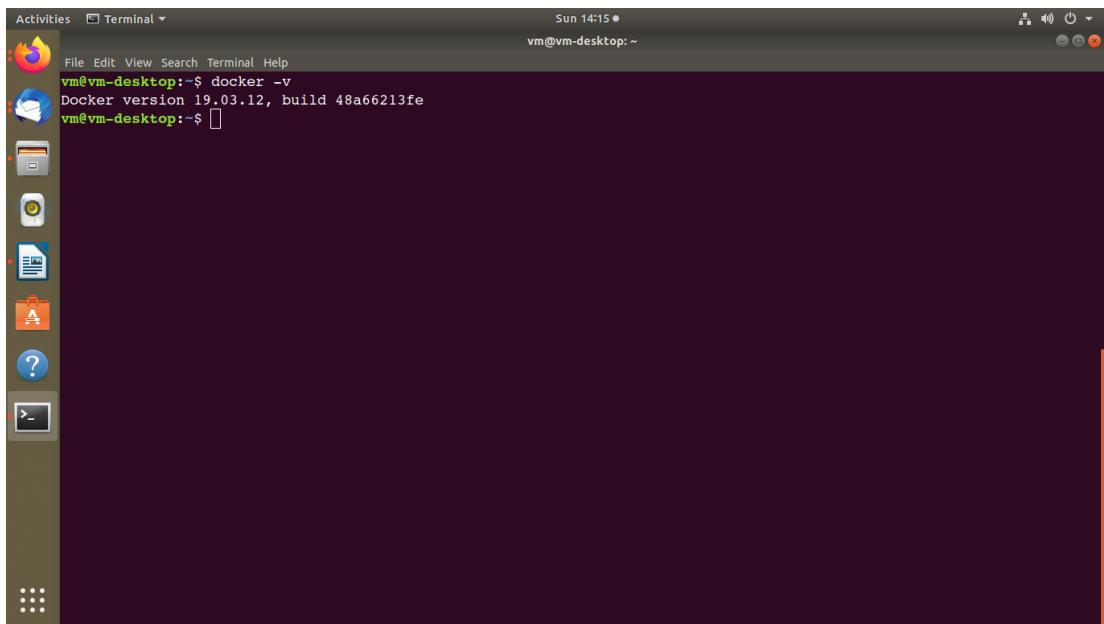
Docker Desktop is an easy-to-install application for your Mac or Windows environment that enables you to build and share containerized applications and microservices. Docker Desktop includes Docker Engine, Docker CLI client, Docker Compose, Notary, Kubernetes, and Credential Helper.

Docker Command Line (commands)

Ref: <https://docs.docker.com/engine/reference/commandline/docker/>

Commands

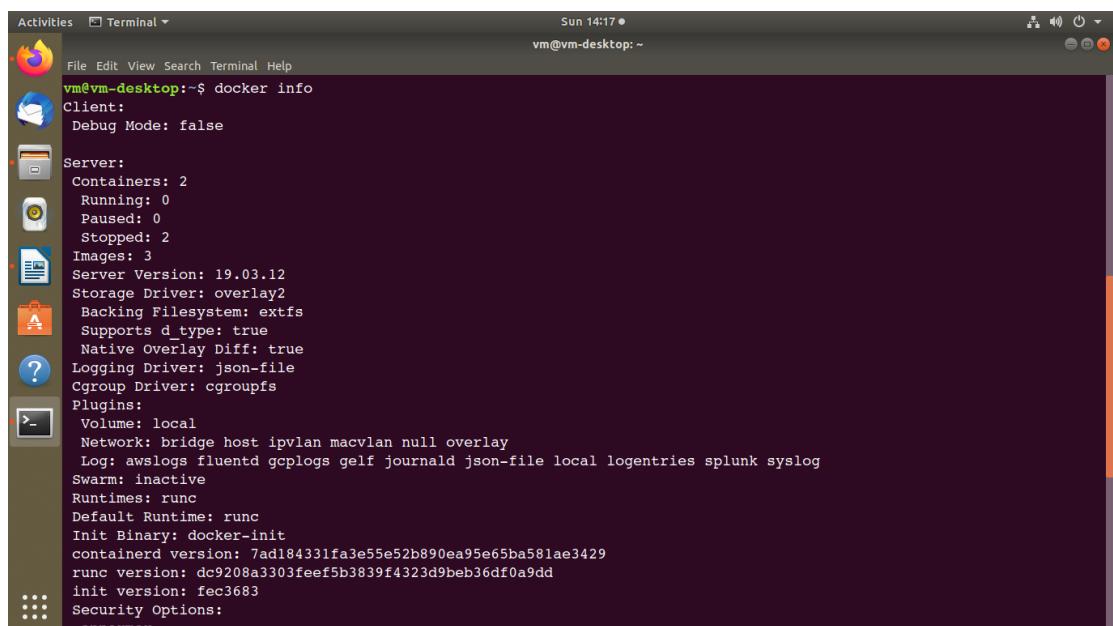
1) docker -v



A screenshot of an Ubuntu desktop environment. On the left is a vertical dock with icons for the Dash, Home, Applications, and Help. A terminal window is open in the center, showing the command 'docker -v' and its output: 'Docker version 19.03.12, build 48a66213fe'. The top bar shows the date and time as 'Sun 14:15' and the user as 'vm@vm-desktop'. The title bar of the terminal says 'Activities Terminal'.

```
File Edit View Search Terminal Help
vm@vm-desktop:~$ docker -v
Docker version 19.03.12, build 48a66213fe
vm@vm-desktop:~$
```

2) docker info - Display system-wide information
docker info [OPTIONS]



A screenshot of an Ubuntu desktop environment. The terminal window shows the command 'docker info' and its detailed output. The output includes information about the client (Debug Mode: false), server (Containers: 2, Images: 3, Server Version: 19.03.12), storage driver (overlay2), backing filesystem (extfs), and various log drivers (awslogs, fluentd, gcplogs, journald, json-file, local, logentries, splunk, syslog). It also lists swarm status (inactive), runtimes (runc), default runtime (runc), init binary (docker-init), containerd version (7ad184331fa3e55e52b890ea95e65ba581ae3429), runc version (dc9208a3303feef5b3839f4323d9beb36df0a9dd), and security options (apparmor).

```
File Edit View Search Terminal Help
vm@vm-desktop:~$ docker info
Client:
  Debug Mode: false

Server:
  Containers: 2
    Running: 0
    Paused: 0
    Stopped: 2
  Images: 3
    Server Version: 19.03.12
    Storage Driver: overlay2
      Backing Filesystem: extfs
      Supports d_type: true
      Native Overlay Diff: true
    Logging Driver: json-file
    Cgroup Driver: cgroups
  Plugins:
    Volume: local
    Network: bridge host ipvlan macvlan null overlay
      Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
  Swarm: inactive
  Runtimes: runc
  Default Runtime: runc
  Init Binary: docker-init
  containerd version: 7ad184331fa3e55e52b890ea95e65ba581ae3429
  runc version: dc9208a3303feef5b3839f4323d9beb36df0a9dd
  init version: fec3683
  Security Options:
    apparmor
```

3) docker history - Show the history of an image
docker history [OPTIONS] IMAGE

```

Activities Terminal Sun 14:19 ● vm@vm-desktop: ~
File Edit View Search Terminal Help
vm@vm-desktop:~$ docker history
"docker history" requires exactly 1 argument.
See 'docker history --help'.

Usage: docker history [OPTIONS] IMAGE

Show the history of an image
vm@vm-desktop:~$ docker history hello-world
IMAGE          CREATED      CREATED BY
bf756fb1ae65  7 months ago   /bin/sh -c #(nop) CMD ["/hello"]
<missing>      7 months ago   /bin/sh -c #(nop) COPY file:7bf12aab75c3867a...  13.3kB
vm@vm-desktop:~$ 

```

4) docker image - Manage images

a) docker ps - List containers

docker ps [OPTIONS]

b) docker images - List images

docker images [OPTIONS] [REPOSITORY[:TAG]]

c) docker rmi - Remove one or more images

docker rmi [OPTIONS] IMAGE [IMAGE...]

```

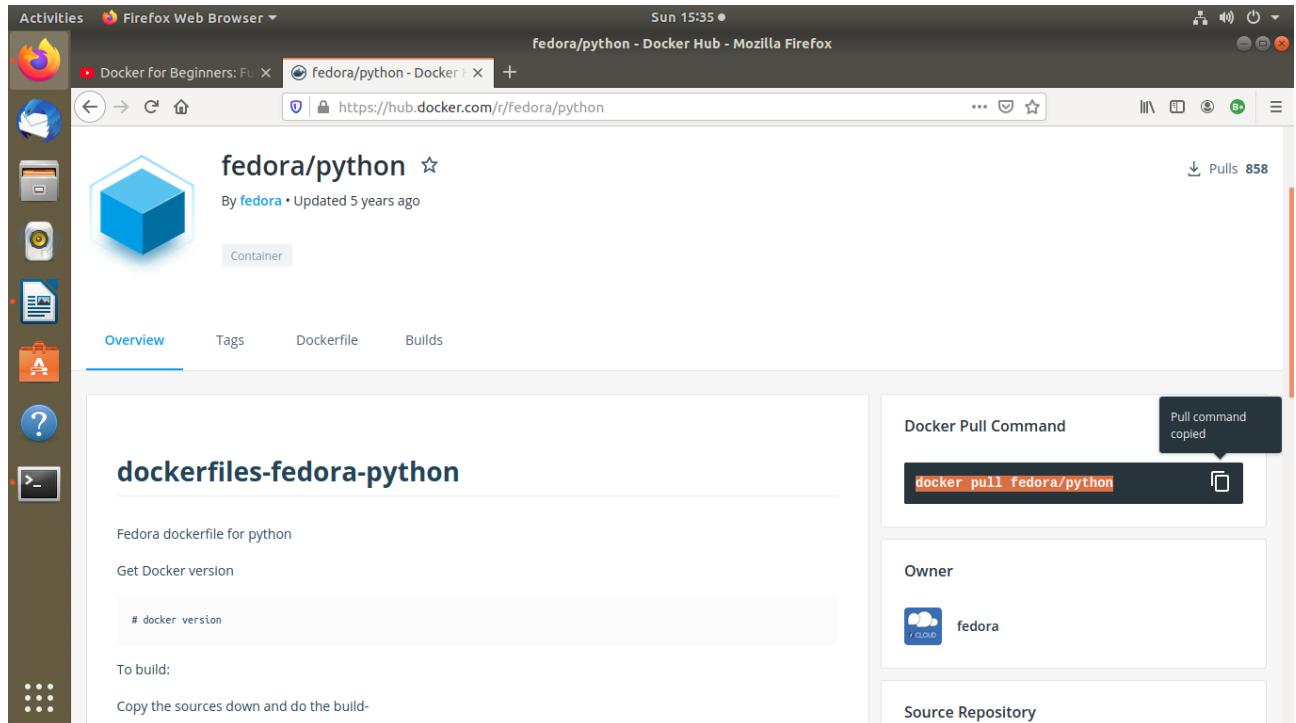
Activities Terminal Sun 14:43 ● vm@vm-desktop: ~
File Edit View Search Terminal Help
vm@vm-desktop:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS
PORTS              NAMES
aa3108fa1863      centos             "/bin/bash"   16 minutes ago   Exited (0) 16 m
inutes ago
6988ef964745      hello-world        "/hello"     40 minutes ago   Exited (0) 40 m
inutes ago
cb0f057c0651      h2oai/dai-centos7-x86_64:1.6.5-cuda9.0   "./run.sh"    9 months ago    Exited (137) 9
months ago
relaxed_satoshi
vm@vm-desktop:~$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED      SIZE
centos              latest   0d120b6ccaa8  12 days ago   215MB
hello-world         latest   bf756fb1ae65  7 months ago   13.3kB
h2oai/dai-centos7-x86_64  1.6.5-cuda9.0   e5bfadbb0e03b  10 months ago  10.8GB
h2oai/dai-centos7-x86_64  1.8.0-cuda10.0   4c71535dd36b  10 months ago  13.6GB
vm@vm-desktop:~$ docker rmi -f centos
Untagged: centos:latest
Untagged: centos@sha256:76d24f3ba3317fa945743bb3746fbaf3a0b752f10b10376960de01da70685fbd
Deleted: sha256:0d120b6ccaa8c5e149176798b3501d4dd1885f961922497cd0abef155c869566
vm@vm-desktop:~$ 

```

Install Tensorflow image or cupy image or any other image you want from the Docker hub.
Here, I am pulling plain fedora image from docker.

Steps 1) login to docker hub account.

- 2) search “image name” in search box (I have searched for fedora/python image)
- 3) open the image link
- 4) copy the command



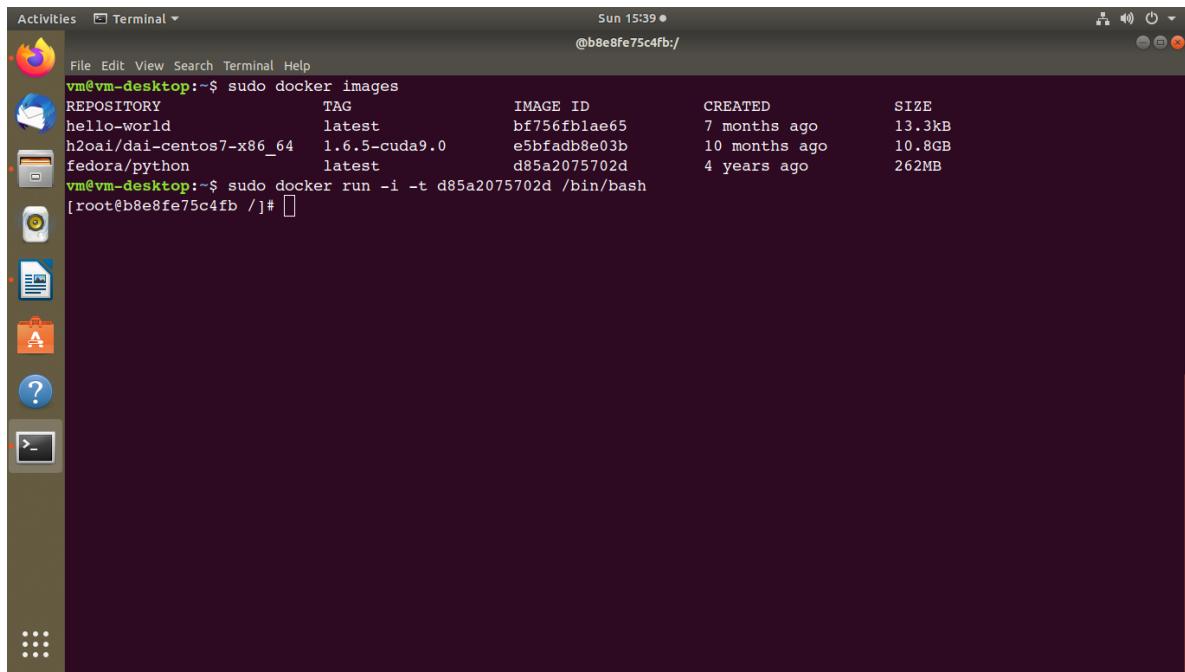
- 5) run the copied command in terminal
docker pull fedora/python

```
Activities Terminal Sun 15:31 ● vm@vm-desktop: ~
File Edit View Search Terminal Help
vm@vm-desktop:~$ sudo docker pull fedora/python
Using default tag: latest
latest: Pulling from fedora/python
Image docker.io/fedora/python:latest uses outdated schema1 manifest format. Please upgrade to a schema2 image for better future compatibility. More information at https://docs.docker.com/registry/spec/deprecated-schema-v1/
a3ed95caeb02: Pull complete
cadab8e68e03: Pull complete
700c4c2a43d3: Pull complete
f7205161d775: Pull complete
bd145e8f4e98: Pull complete
3ff8336e50c0: Pull complete
Digest: sha256:9cccd9c4e88fcf75c512ba079c37eb05b6297f6f77d52fa4ff2ef0817956725c9
Status: Downloaded newer image for fedora/python:latest
docker.io/fedora/python:latest
vm@vm-desktop:~$
```

- 6) list the images by using command: sudo docker images
- 7) run the currently downloaded image using command:
sudo docker run -i -t d85a2075702d /bin/bash

-i => for input or interactive

-t => terminal



```
Sun 15:39 ● @b8e8fe75c4fb:/  
File Edit View Search Terminal Help  
REPOSITORY TAG IMAGE ID CREATED SIZE  
hello-world latest bf756fb1ae65 7 months ago 13.3kB  
h2oai/dai-centos7-x86_64 1.6.5-cuda9.0 e5bfadb8e03b 10 months ago 10.8GB  
fedora/python latest d85a2075702d 4 years ago 262MB  
vm@vm-desktop:~$ sudo docker run -i -t d85a2075702d /bin/bash  
[root@b8e8fe75c4fb ~]#
```

8) docker rm - Remove one or more containers

docker rm [OPTIONS] CONTAINER [CONTAINER...]



```
Error: No such container: raj9367/assignment_5:myfirstimage  
root@Malilinux:~# docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES  
4653a848910f raj9367/assignment_5:myfirstimage "echo 'Hello ..! fro.'" 4 minutes ago Exited (0) 4 minutes ago gracious_shtern  
f64c2c6e9e20 raj9367/assignment_5:myfirstimage "echo 'Hello ..! fro.'" 5 minutes ago Exited (0) 5 minutes ago objective_fermi  
2059a2fe29ab hello-world "/hello" About an hour ago Exited (0) About an hour ago jolly_chaplygin  
490f9027882e d85a2075702d "/bin/bash" About an hour ago Exited (0) About an hour ago goofy_torvalds  
5bdd8898446f hello-world "/hello" About an hour ago Exited (0) About an hour ago charming_dirac  
root@Malilinux:~# docker rm 4653a848910f  
4653a848910f  
root@Malilinux:~#
```

9) Docker container logs

Usage: docker container logs [OPTIONS] CONTAINER

Fetch the logs of a container

The terminal window shows the output of the Docker hello-world container. It includes a welcome message, information about how Docker worked to run the container, instructions to run an Ubuntu container, details about sharing images, and links to Docker documentation. The terminal prompt is 'root@KaliLinux:~#'. A large number '10)' is visible in the top right corner of the window.

```
Hello From Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
root@KaliLinux:~#
```

Docker push

Usage: `docker push [OPTIONS] NAME[:TAG]`

Push an image or a repository to a registry

The terminal window shows the usage of the `docker push` command. It indicates that exactly one argument is required and provides the usage syntax. A large number '10)' is visible in the top right corner of the window.

```
"docker push" requires exactly 1 argument.
See 'docker push --help'.

Usage: docker push [OPTIONS] NAME[:TAG]

Push an image or a repository to a registry
root@KaliLinux:~/Downloads/Dockerfile#
```

11) Docker tag

Usage: `docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]`

Create a tag `TARGET_IMAGE` that refers to `SOURCE_IMAGE`

The GitHub repository page for 'assignment_5' shows basic details like the owner 'raj9367' and a public view link. Below the repository info is a terminal window demonstrating the `docker tag` command. It first runs a container from the source image, then attempts to tag it with the target image, which fails because the source image does not exist. A large number '11)' is visible in the top right corner of the window.

This repository does not have a description

Docker commands

To push a new tag to this repository,

```
root@KaliLinux:~/Downloads/Dockerfile
root@KaliLinux:~/Downloads/Dockerfile# docker run raj9367/assignment_5:myfirstimage
Hello ..! from my first docker image
root@KaliLinux:~/Downloads/Dockerfile# docker tag
"docker tag" requires exactly 2 arguments.
See 'docker tag --help'.

Usage: docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]

Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
root@KaliLinux:~/Downloads/Dockerfile# docker tag myimage:latest raj9367/assignment_5:myfirstimage
Error response from daemon: No such image: myimage:latest
root@KaliLinux:~/Downloads/Dockerfile#
```

10) Docker login

Usage: `docker login [OPTIONS] [SERVER]`

Log in to a Docker registry.

If no server is specified, the default is defined by the daemon.

```

root@KaliLinux:~/Downloads/Dockerfile# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: raj9367
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
lub or Bitbu root@KaliLinux:~/Downloads/Dockerfile# 

```

Docker Hub connects to GitHub and Bitbucket. You'll need to re-link a GitHub or create new automated builds. [Learn More](#)

Docker Hub

Ref: <https://docs.docker.com/docker-hub/>

Docker Hub is a service provided by Docker for finding and sharing container images with your team. It provides the following major features:

- [Repositories](#): Push and pull container images.
- [Teams & Organizations](#): Manage access to private repositories of container images.
- [Official Images](#): Pull and use high-quality container images provided by Docker.
- [Publisher Images](#): Pull and use high-quality container images provided by external vendors.
- [Builds](#): Automatically build container images from GitHub and Bitbucket and push them to Docker Hub.
- [Webhooks](#): Trigger actions after a successful push to a repository to integrate Docker Hub with other services.

Step 1: Sign up for Docker Hub

Start by [creating an account](#).

Step 2) write own docker file and push it.

```

Ste root@KaliLinux:~/Downloads
root@KaliLinux:~# cd Downloads/
root@KaliLinux:~/Downloads# mkdir Dockerfiles
root@KaliLinux:~/Downloads# touch myimage
root@KaliLinux:~/Downloads# gedit myimage
root@KaliLinux:~/Downloads# cat myimage
Hello...From my first docker file
root@KaliLinux:~/Downloads# gedit myimage
root@KaliLinux:~/Downloads# gedit myimage
root@KaliLinux:~/Downloads# cat myimage
FROM ubuntu

CMD ["echo","Hello...From my first docker file"]
root@KaliLinux:~/Downloads# docker build -t myimage:latest
"docker build" requires exactly 1 argument.
See 'docker build --help'.

Usage: docker build [OPTIONS] PATH | URL | -
      Build an image from a Dockerfile
root@KaliLinux:~/Downloads# docker build -t "myimage" .
Sending build context to Docker daemon 487.5MB

```

```

root@KaliLinux:~/Downloads/Dockerfile# docker tag myimage:latest raj9367/assignment_5:myfirstimage
root@KaliLinux:~/Downloads/Dockerfile# docker push raj9367/assignment_5:myfirstimage
The push refers to repository [docker.io/raj9367/assignment_5]
d4dfa212623: Pushed
cba97cc5811c: Pushed
0c78fac124da: Pushed
myfirstimage: digest: sha256:a276f8deb5da0863f621d3a3636202f9eb022a77219f2ebfb90752082b375e6 size: 943
root@KaliLinux:~/Downloads/Dockerfile# 
```

Enable

Link a source provider and run a build to see build results here.

: of the most recent tags here.

Docker commands

To push a new tag to this repository:

`docker push raj9367/assignment_5:tagname`

Tags and Scans

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
myfirstimage	Ubuntu	a few seconds ago	a few seconds...

Recent builds

Link a source provider and run a build to see build results here.



Explore

Docker Editions
Containers
Plugins
Baseline

Account

My Content
Billing

Publish

Publisher Center

Resources

Docker Blog
Feedback
Documentation
Hub Release Notes
Forums

Support

Feedback
Documentation
Hub Release Notes
Forums

```

^Croot@KaliLinux:~/Downloads/Dockerfile# docker rmi -f raj9367/assignment_5:myfirstimage
Untagged: raj9367/assignment_5:myfirstimage
Untagged: raj9367/assignment_5@sha256:a276f8deb5da0863f621d3a3636202f9eb022a77219f2ebfb90752082b375e6
Deleted: sha256:8872216dad46211ad19e2e286e65a5bec41ac86c36b1b2146a8774a2db609c40
root@KaliLinux:~/Downloads/Dockerfile# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest 8e428cff54c8 4 days ago 72.9MB
hello-world latest d1165f221234 3 weeks ago 13.3kB
root@KaliLinux:~/Downloads/Dockerfile# docker pull raj9367/assignment_5:myfirstimage
myfirstimage: Pulling from raj9367/assignment_5
04a5f4cda3ee: Already exists
ff496a88c8ed: Already exists
0ce83f459fe7: Already exists
Digest: sha256:a276f8deb5da0863f621d3a3636202f9eb022a77219f2ebfb90752082b375e6
Status: Downloaded newer image for raj9367/assignment_5:myfirstimage
docker.io/raj9367/assignment_5:myfirstimage
root@KaliLinux:~/Downloads/Dockerfile# docker run raj9367/assignment_5:myfirstimage
Hello ...! from my first docker image
root@KaliLinux:~/Downloads/Dockerfile# 
```

Assignment No 6

Title: Setup Single Node Kubernetes Cluster with Minikube.

Theory:

¶ What is Kubernetes?

Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.

The name Kubernetes originates from Greek, meaning helmsman or pilot. Google open-sourced the Kubernetes project in 2014. Kubernetes combines [over 15 years of Google's experience](#) running production workloads at scale with best-of-breed ideas and practices from the community

¶ Why you need Kubernetes and what it can do

Containers are a good way to bundle and run your applications. In a production environment, you need to manage the containers that run the applications and ensure that there is no downtime. For example, if a container goes down, another container needs to start. Wouldn't it be easier if this behavior was handled by a system?

That's how Kubernetes comes to the rescue! Kubernetes provides you with a framework to run distributed systems resiliently. It takes care of scaling and failover for your application, provides deployment patterns, and more. For example, Kubernetes can easily manage a canary deployment for your system.

Kubernetes provides you with:

- **Service discovery and load balancing**
- **Storage orchestration**

- **Automated rollouts and rollbacks**
- **Automatic bin packing**
- **Self-healing**
- **Secret and configuration management**

Kubernetes Components

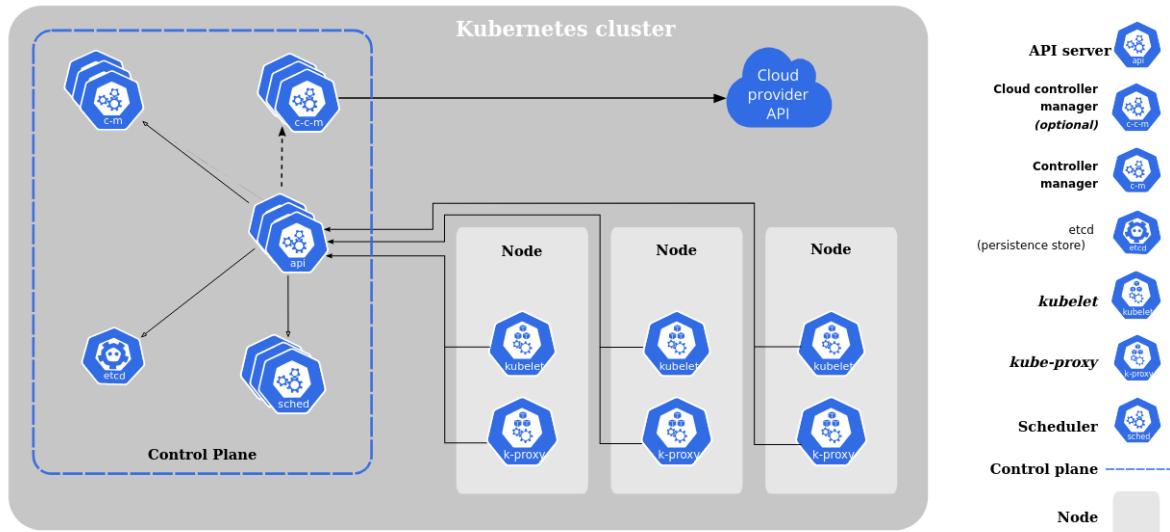
When you deploy Kubernetes, you get a cluster.

A Kubernetes cluster consists of a set of worker machines, called [nodes](#), that run containerized applications. Every cluster has at least one worker node.

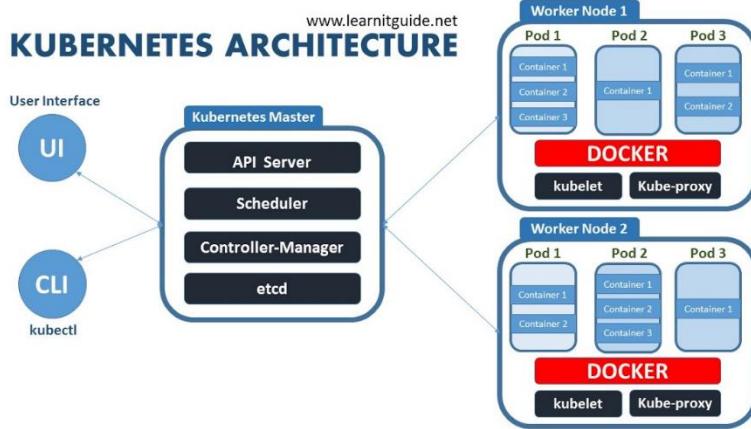
The worker node(s) host the [Pods](#) that are the components of the application workload. The [control plane](#) manages the worker nodes and the Pods in the cluster. In production environments, the control plane usually runs across multiple computers and a cluster usually runs multiple nodes, providing fault-tolerance and high availability.

This document outlines the various components you need to have a complete and working Kubernetes cluster.

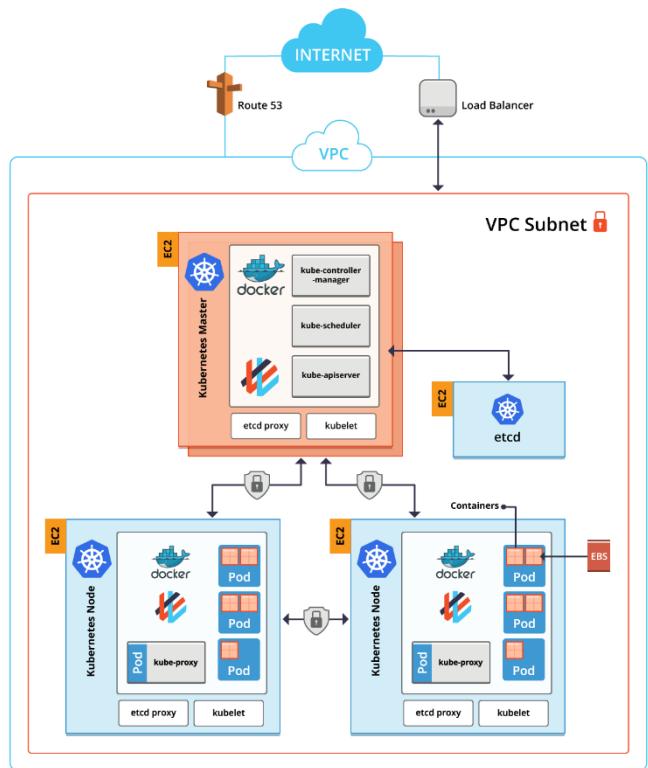
Here's the diagram of a Kubernetes cluster with all the components tied together.



🔗 Kubernetes Architecture



Kubernetes Architecture on AWS:



Control Plane Components

The control plane's components make global decisions about the cluster (for example, scheduling), as well as detecting and responding to cluster events (for example, starting up a new [pod](#) when a deployment's replicas field is unsatisfied).

Control plane components can be run on any machine in the cluster. However, for simplicity, setup scripts typically start all control plane components on the same machine, and do not run user containers on this machine. See [Building High-Availability Clusters](#) for an example multi-master-VM setup.

kube-apiserver

The API server is a component of the Kubernetes [control plane](#) that exposes the Kubernetes API. The API server is the front end for the Kubernetes control plane.

The main implementation of a Kubernetes API server is [kube-apiserver](#). kube-apiserver is designed to scale horizontally—that is, it scales by deploying more instances. You can run several instances of kube-apiserver and balance traffic between those instances.

etcd

Consistent and highly-available key value store used as Kubernetes' backing store for all cluster data.

If your Kubernetes cluster uses etcd as its backing store, make sure you have a [back up](#) plan for those data.

You can find in-depth information about etcd in the official [documentation](#).

kube-scheduler

Control plane component that watches for newly created [Pods](#) with no assigned [node](#), and selects a node for them to run on.

Factors taken into account for scheduling decisions include: individual and collective resource requirements, hardware/software/policy constraints, affinity and anti-affinity specifications, data locality, inter-workload interference, and deadlines.

kube-controller-manager

Control Plane component that runs [controller](#) processes.

Logically, each [controller](#) is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

These controllers include:

- Node controller: Responsible for noticing and responding when nodes go down.
- Replication controller: Responsible for maintaining the correct number of pods for every replication controller object in the system.
- Endpoints controller: Populates the Endpoints object (that is, joins Services & Pods).
- Service Account & Token controllers: Create default accounts and API access tokens for new namespaces.

cloud-controller-manager

A Kubernetes [control plane](#) component that embeds cloud-specific control logic. The cloud controller manager lets you link your cluster into your cloud provider's API, and separates out the components that interact with that cloud platform from components that just interact with your cluster.

The cloud-controller-manager only runs controllers that are specific to your cloud provider. If you are running Kubernetes on your own premises, or in a learning environment inside your own PC, the cluster does not have a cloud controller manager.

As with the kube-controller-manager, the cloud-controller-manager combines several logically independent control loops into a single binary that you run as a single process. You can scale horizontally (run more than one copy) to improve performance or to help tolerate failures.

The following controllers can have cloud provider dependencies:

- Node controller: For checking the cloud provider to determine if a node has been deleted in the cloud after it stops responding
- Route controller: For setting up routes in the underlying cloud infrastructure
- Service controller: For creating, updating and deleting cloud provider load balancers

Node Components

Node components run on every node, maintaining running pods and providing the Kubernetes runtime environment.

kubelet

An agent that runs on each [node](#) in the cluster. It makes sure that [containers](#) are running in a [Pod](#).

The kubelet takes a set of PodSpecs that are provided through various mechanisms and ensures that the containers described in those PodSpecs are running and healthy. The kubelet doesn't manage containers which were not created by Kubernetes.

kube-proxy

kube-proxy is a network proxy that runs on each [node](#) in your cluster, implementing part of the Kubernetes [Service](#) concept.

[kube-proxy](#) maintains network rules on nodes. These network rules allow network communication to your Pods from network sessions inside or outside of your cluster.

kube-proxy uses the operating system packet filtering layer if there is one and it's available. Otherwise, kube-proxy forwards the traffic itself.

Container runtime

The container runtime is the software that is responsible for running containers.

Kubernetes supports several container runtimes: [Docker](#), [containerd](#), [CRI-O](#), and any implementation of the [Kubernetes CRI \(Container Runtime Interface\)](#).

Pods

Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

A *Pod* (as in a pod of whales or pea pod) is a group of one or more [containers](#), with shared storage/network resources, and a specification for how to run the containers. A Pod's contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific "logical host": it contains one or more application containers which are relatively tightly coupled. In non-cloud contexts, applications executed on the same physical or virtual machine are analogous to cloud applications executed on the same logical host.

Using Pods

Usually you don't need to create Pods directly, even singleton Pods. Instead, create them using workload resources such as [Deployment](#) or [Job](#). If your Pods need to track state, consider the [StatefulSet](#) resource.

Pods in a Kubernetes cluster are used in two main ways:

- **Pods that run a single container.** The "one-container-per-Pod" model is the most common Kubernetes use case; in this case, you can think of a Pod as a wrapper around a single container; Kubernetes manages Pods rather than managing the containers directly.
- **Pods that run multiple containers that need to work together.** A Pod can encapsulate an application composed of multiple co-located containers that are tightly coupled and need to share resources. These co-located containers form a single cohesive unit of service—for example, one container serving data stored in a shared volume to the public, while a separate *sidecar* container refreshes or updates those files. The Pod wraps these containers, storage resources, and an ephemeral network identity together as a single unit.

Service

An abstract way to expose an application running on a set of [Pods](#) as a network service.

With Kubernetes you don't need to modify your application to use an unfamiliar service discovery mechanism. Kubernetes gives Pods their own IP addresses and a single DNS name for a set of Pods, and can load-balance across them.

Ingress

FEATURE STATE: Kubernetes v1.19 [stable]

An API object that manages external access to the services in a cluster, typically HTTP.

Ingress may provide load balancing, SSL termination and name-based virtual hosting.

Terminology

For clarity, this guide defines the following terms:

- **Node:** A worker machine in Kubernetes, part of a cluster.
- **Cluster:** A set of Nodes that run containerized applications managed by Kubernetes. For this example, and in most common Kubernetes deployments, nodes in the cluster are not part of the public internet.
- **Edge router:** A router that enforces the firewall policy for your cluster. This could be a gateway managed by a cloud provider or a physical piece of hardware.
- **Cluster network:** A set of links, logical or physical, that facilitate communication within a cluster according to the Kubernetes [networking model](#).
- **Service:** A Kubernetes [Service](#) that identifies a set of Pods using [label](#) selectors. Unless mentioned otherwise, Services are assumed to have virtual IPs only routable within the cluster network.

What is Ingress?

[Ingress](#) exposes HTTP and HTTPS routes from outside the cluster to [services](#) within the cluster. Traffic routing is controlled by rules defined on the Ingress resource.

Here is a simple example where an Ingress sends all its traffic to one Service:



An Ingress may be configured to give Services externally-reachable URLs, load balance traffic, terminate SSL / TLS, and offer name-based virtual hosting. An [Ingress controller](#) is responsible for fulfilling the Ingress, usually with a load balancer, though it may also configure your edge router or additional frontends to help handle the traffic.

An Ingress does not expose arbitrary ports or protocols. Exposing services other than HTTP and HTTPS to the internet typically uses a service of type [Service.Type=NodePort](#) or [Service.Type=LoadBalancer](#).

Try Kubernetes:

There are many ways to install Kubernetes.

1) Manual means that you manually set up everything from the networking bits over downloading, configuring and launching components such as etcd or kube-apiserver. You've got full control over what goes where, but it might take you some time and it's error-prone. The ultimate reference in this area is Kelsey Hightower's [Kubernetes The Hard Way](#) (KTHW).

2) Installer are CLI tools that leverage templates and/or automation tools such as Terraform and Ansible. You typically have a lot control over what is going on, but less than in the manual approach. Examples include:

- OpenCredo's Kubernetes from scratch to AWS with Terraform and Ansible
- [kubeadm](#) (bare metal installation, a building block for other installers)
- [kops](#), mainly AWS (AWS also provide service for kubernetes: EKS – Elastic Kubernets Service)
- [kubicorn](#)
- [kubespray](#) (Ansible-based, both bare-metal and cloud)
- OpenShift [advanced install](#) based on Ansible

3) Hosted effectively means little to no installation effort on your end. For example:

- [Azure Container Service](#) (ACS) with Kubernetes
 - [Google Container Engine](#) (GKE)
 - [IBM Bluemix Container Service](#)
 - [OpenShift Online](#) (OSO)
- 4) [Minikube](#) or [Minishift](#) – Virtualized Environment for Kubernetes

Setup Single Node Kubernetes Cluster with Minikube

- 1) Installing Minikube -> <https://kubernetes.io/docs/tasks/tools/install-minikube/>
Install Virtualbox latest edition.
- 2) Setting up Minikube on virtualbox -> <https://kubernetes.io/docs/tasks/tools/install-minikube/>

You will need to keep the minikube in the PATH both on Windows/Linux

Use the following instruction to setup single node Kubernetes cluster

```
minikube start --driver=virtualbox
```

You can also set custom configuration like

```
minikube start --driver=virtualbox --cpus=2 --memory=4096m
```

- 3) Open the minikube dashboard with following command. It will take 2-10 mins depending on your bandwidth.

```
minikube dashboard
```

- 4) The dashboard will open in your default browser

The screenshot shows the Kubernetes Dashboard's Overview page. On the left, a sidebar lists Cluster, Cluster Roles, Namespaces, Nodes, Persistent Volumes, Service Accounts, Storage Classes, Workloads (with Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, and Replica Sets), and Services. The main area has two sections: 'Discovery and Load Balancing' (Services) and 'Config and Storage' (Secrets). The Services section displays a table with one row for 'kubernetes'. The table columns are Name, Namespace, Labels, Cluster IP, Internal Endpoints, External Endpoints, and Created. The kubernetes entry has the following details:

Name	Namespace	Labels	Cluster IP	Internal Endpoints	External Endpoints	Created
kubernetes	default	component: apiserver provider: kubernetes	10.96.0.1	kubernetes:443 TCP kubernetes:0 - TCP	-	4 minutes ago

The Secrets section is currently empty.

5) On the right side corner you will see a symbol for , click on it, and go to -> Create from form

The dialog has three tabs: 'Create from input', 'Create from file', and 'Create from form', with 'Create from form' selected. The fields are as follows:

- App name ***: A text input field with a character count indicator '0 / 24'.
- Container image ***: A text input field with a placeholder 'Enter the URL of a public image on any registry, or a private image hosted on Docker Hub or Google Container Registry.' and a 'Learn more' link.
- Number of pods ***: A text input field containing '1'.
- Service ***: A dropdown menu set to 'None'.

Below the form are three buttons: 'Deploy' (disabled), 'Cancel', and 'Show advanced options'.

6) Enter the details as below

App Name: nginx

Container image: nginx

Number of pods: 1

Service: External

Port: 80

Target Port: 80

Protocol: TCP

The screenshot shows a web-based form for creating a new application deployment. The form has three tabs at the top: 'Create from input', 'Create from file', and 'Create from form'. The 'Create from form' tab is selected and highlighted with a blue underline.

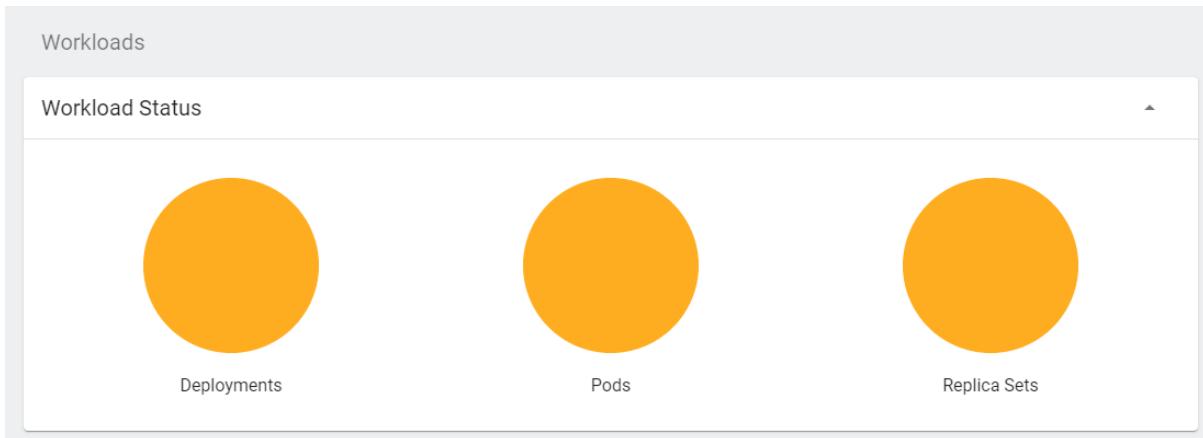
The form fields are as follows:

- App name ***: nginx (5 / 24 characters used)
- Container Image ***: nginx
- Number of pods ***: 1
- Service ***: External
- Port ***: 80
- Target port ***: 80
- Protocol ***: TCP

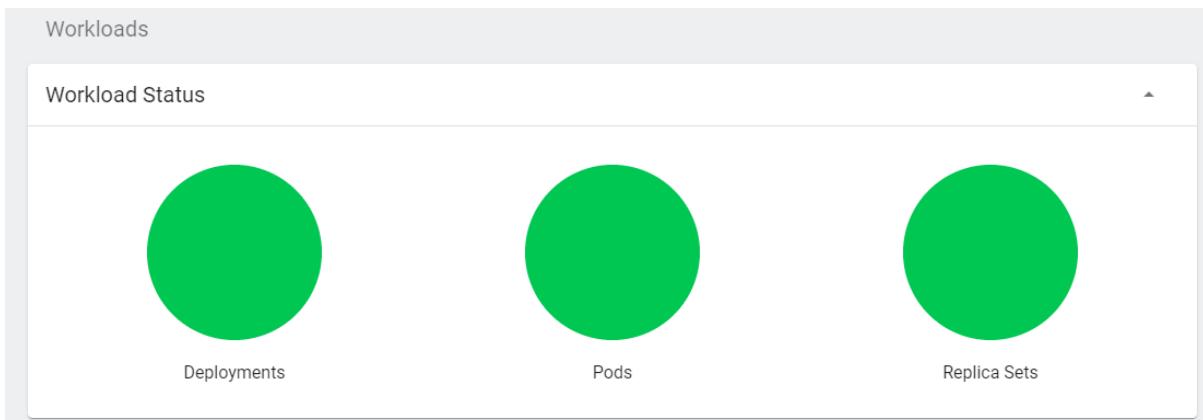
A small trash can icon is located next to the Protocol dropdown menu.

Click Deploy

7) Now you will see something like this,



Wait for it to turn green.



You will see the details for deployment below.

8) Access the nginx application in your browser with following command.

```
minikube service nginx
```

The nginx default page will open in browser and you will see the service details as well.

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

References for further study:

- 1) <https://kubernetes.io/docs/home/>
- 2) you need to learn Kubernetes RIGHT NOW!! - Networkchuck
- 3) techworld with Nana
- 4) Kubernetes Concepts Explained in 9 minutes! - KodeKloud
- 5) Kubernetes Tutorial for Beginners | Kubernetes Tutorial | Intellipaat
- 6) Install Kubernetes | Setup Kubernetes Step by Step | Kubernetes Training | Intellipaat

Output :

```
C:\WINDOWS\system32>choco install minikube
Chocolatey v0.10.15
[Pending] Removing incomplete install for 'visualstudio2017-workload-vctools'
Installing the following packages:
minikube
By installing you accept licenses for the packages.

kubernetes-cli v1.20.5 [Approved]
kubernetes-cli package files install completed. Performing other installation steps.
The package kubernetes-cli wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N)o/[P]rint):

Timeout or your choice of '' is not a valid selection.
You must select an answer
Do you want to run the script?([Y]es/[A]ll - yes to all/[N)o/[P]rint): All

Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar.gz to C:\ProgramData\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar to C:\ProgramData\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
ShimGen has successfully created a shim for kubectl.exe
The install of kubernetes-cli was successful.
* 1 nodes stopped.
```

```
C:\WINDOWS\system32>minikube status
minikube
  type: Control Plane
  host: Running
  kubelet: Running
  apiserver: Running
  kubeconfig: Configured
  timeToStop: Nonexistent

C:\WINDOWS\system32>minikube dashboard
* Enabling dashboard ...
  - Using image kubernetesui/dashboard:v2.1.0
  - Using image kubernetesui/metrics-scraper:v1.0.4
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:53444/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
^C
C:\WINDOWS\system32>
C:\WINDOWS\system32>minikube service nginx
|-----|-----|-----|
| NAMESPACE | NAME | TARGET PORT | URL |
|-----|-----|-----|
| default   | nginx | tcp-80-80-19d81/80 | http://192.168.99.100:31971 |
|-----|-----|-----|
* Opening service default/nginx in default browser...

C:\WINDOWS\system32>minikube dashboard
* 1 nodes stopped.

Assignment No ... db_demo.php Assignmetn No... +1
```

```

Administrator: Command Prompt
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:53639/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
`C:\Windows\system32>minikube dashboard
* The control plane node must be running for this command
  - To start a cluster, run: "minikube start"

C:\Windows\system32>minikube start
* minikube v1.18.1 on Microsoft Windows 10 Home Single Language 10.0.19042 Build 19042
* Using the virtualbox driver based on existing profile
* Starting control plane node minikube in cluster minikube
* Restarting existing virtualbox VM for "minikube" ...
* Preparing Kubernetes v1.20.2 on Docker 20.10.3 ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v4
  - Using image kubernetesui/dashboard:v2.1.0
  - Using image kubernetesui/metrics-scraper:v1.0.4
* Enabled addons: storage-provisioner, default-storageclass, dashboard
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default

C:\Windows\system32>kubectl get svc
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP     22m
nginx       LoadBalancer  10.111.131.145  <pending>     80:31971/TCP  13m

C:\Windows\system32>minikube dashboard
* Verifying dashboard health ...
* Launching proxy ...
  * 1 nodes stopped.

Assignment No ... db_demo.php Assignmetn No... +1

```

The screenshot shows the Kubernetes Dashboard 'Create' interface. The URL in the address bar is 127.0.0.1:53444/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/#/create?namespace=default. The form fields are as follows:

- App name:** nginx
- Container image:** nginx
- Number of pods:** 1
- Service:** External
- Port:** 80
- Target port:** 80
- Protocol:** TCP

At the bottom of the form are buttons for **Deploy**, **Cancel**, and **Show advanced options**.

The screenshot shows the Kubernetes Dashboard interface. On the left, a sidebar lists various resources: Workloads (Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets), Service (Ingresses, Services), Config and Storage (Config Maps, Persistent Volume Claims, Secrets, Storage Classes), and Cluster (Cluster Role Bindings, Cluster Roles, Namespaces, Network Policies, Notes). The main area displays the 'Workloads' section with three large green circles representing Deployments, Pods, and Replica Sets. Below this, the 'Deployments' section shows a table with one entry:

Name	Namespace	Labels	Pods	Created	Images
nginx	default	k8s-app: nginx	1 / 1	9 minutes ago	nginx

Below the Deployments table is the 'Pods' section, also showing one entry:

Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx-7f46644dc-9bkhp	default	k8s-app: nginx pod-template-hash: 7f46644dc	minikube	Running	0	-	-	8.000000000000001

The screenshot shows a web browser window displaying the 'Welcome to nginx!' page. The URL bar indicates the page is not secure (Not secure) and shows the IP address 192.168.99.100:31971. The page content is as follows:

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

