



Experiment No. 5

Title:

Implement and configure Google App Engine to deploy Python Program application.

Objective:

To learn Google App Engine

Tools used:

Internet , GCP , python app , Github

Prerequisite:

Understanding of Google Cloud , python , Uses of Github

Theory:

App Engine

App Engine is a fully managed, serverless platform for developing and hosting web applications at scale. You can choose from several popular languages, libraries, and frameworks to develop your apps, and then let App Engine take care of provisioning servers and scaling your app instances based on demand.

Python on Google App Engine

App Engine offers you a choice between two Python language environments. Both environments have the same code-centric developer workflow, scale quickly and efficiently to handle increasing demand, and enable you to use Google's proven serving technology to build your web, mobile and IoT applications quickly and with minimal operational overhead. While the two environments have a lot in common, they differ in a few important ways.

Applications run in a secure, sandboxed environment, allowing App Engine standard environment to distribute requests across multiple servers, and scaling servers to meet traffic demands. Your application runs within its own secure, reliable environment that is independent of the hardware, operating system, or physical location of the server.



NUTAN COLLEGE OF ENGINEERING & RESEARCH (NCER)
Department of Computer Science & Engineering (CSE)

This hands-on lab shows you how to create a small App Engine application that displays a short message.

What you'll learn

In this lab you'll do the following with a Python app:

Clone/download

Test

Update

Test

Deploy

Before you click the Start Lab button

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

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

To complete this lab, you need:

Access to a standard internet browser (Chrome browser recommended).

Note: Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

Time to complete the lab---remember, once you start, you cannot pause a lab.

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

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

NUTAN COLLEGE OF ENGINEERING & RESEARCH (NCER)
Department of Computer Science & Engineering (CSE)

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 the **Lab Details** panel with the following:
 - The **Open Google Console** button
 - Time remaining
 - The temporary credentials that you must use for this lab
 - Other information, if needed, to step through this lab
2. Click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.

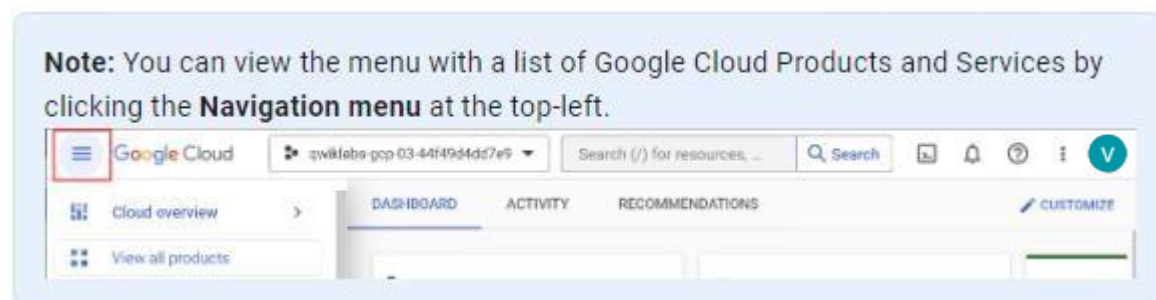
Tip: Arrange the tabs in separate windows, side-by-side.

Note: If you see the **Choose an account** dialog, click **Use Another Account**.

3. If necessary, copy the **Username** from the **Lab Details** panel and paste it into the **Signin** dialog. Click **Next**.
4. Copy the **Password** from the **Lab Details** panel and paste it into the **Welcome** dialog. Click **Next**.

Important: You must use the credentials from the left panel. Do not use your Google Cloud Skills Boost credentials. **Note:** Using your own Google Cloud account for this lab may incur extra charges.


5. Click through the subsequent pages:
 - Accept the terms and conditions.
 - Do not add recovery options or two-factor authentication (because this is a temporary account).
 - Do not sign up for free trials.After a few moments, the Cloud Console opens in this tab.



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

Activate Cloud Shell

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

1. Click **Activate Cloud Shell**  at the top of the Google Cloud console.
When you are connected, you are already authenticated, and the project is set to your **PROJECT_ID**. The output contains a line that declares the **PROJECT_ID** for this session:

```
Your Cloud Platform project in this session is set to YOUR_PROJECT_ID
```

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

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

2. (Optional) You can list the active account name with this command:
gcloud auth list

Copied!

```
content_copy
```

3. Click **Authorize**.
4. Your output should now look like this:

Output:

```
ACTIVE: *  
  
ACCOUNT: student-01-xxxxxxxxxxxx@qwiklabs.net
```

To set the active account, run:

```
$ gcloud config set account 'ACCOUNT'
```

5. (Optional) You can list the project ID with this command:
gcloud config list project

Copied!

content_copy

Output:

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

Example output:

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

Note: For full documentation of gcloud, in Google Cloud, refer to the gcloud CLI overview guide.

```
gcloud config set compute/region "REGION"
```

Copied!

content_copy

Task 1. 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 **Navigation menu**, click **APIs & Services > Library**.
2. Type "App Engine Admin API" in the search box.
3. Click the **App Engine Admin API** card.
4. Click **Enable**. If there is no prompt to enable the API, then it is already enabled and no action is needed.

Task 2. 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:

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

Copied!

content_copy

2. Go to the directory that contains the sample code:

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

Copied!

content_copy

Task 3. Test the application

Test the application using the Google Cloud development server (`dev_appserver.py`), which is included with the preinstalled App Engine SDK.

1. From within your helloworld directory where the app's `app.yaml` configuration file is located, start the Google Cloud development server with the following command:

```
dev_appserver.py app.yaml
```

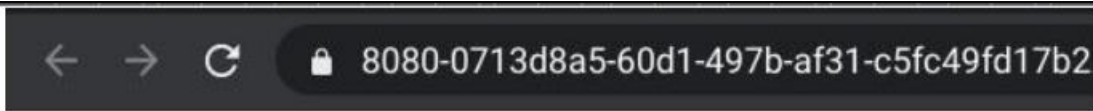
Copied!

content_copy

The development server is now running and listening for requests on port 8080.

2. View the results by clicking the **Web preview** (🌐) > **Preview on port 8080**.

You'll see this in a new browser window:



Hello World!

Tas

k 4. 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
```

Copied!

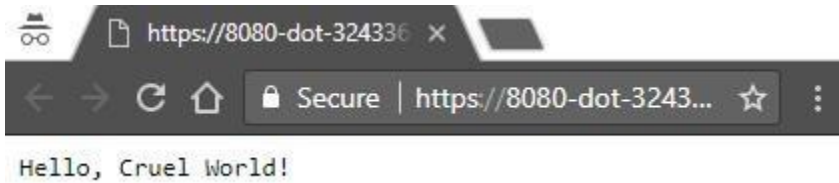
```
content_copy
```

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

Copied!

```
content_copy
```

4. Change "Hello World!" to "Hello, Cruel World!".
5. Save the file with CTRL-S and exit with CTRL-X.
6. Reload the Hello World! Browser or click the **Web Preview** (🌐) > **Preview on port8080** to see the results.



Task 5. 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:
`gcloud app deploy`

Copied! content_copy

2. Enter the number that represents your region: `<REGION>`
 3. The App Engine application will then be created.
- Example output:

```
Creating App Engine application in project [qwiklabs-gcp-233dca09c0ab577b] and region  
["REGION"]....done.
```

```
Services to deploy:
```

```
descriptor:  [/home/gcpstaging8134_student/python-docs-  
samples/appengine/standard/hello_world/app.yaml]
```



```
$ gcloud app logs tail -s default
```

To view your application in the web browser run:

```
$ gcloud app browse
```

Note: If you receive an error as "Unable to retrieve P4SA" while deploying the app, then re-run the above command.

Task 6. View your application

- To launch your browser enter the following command, then click on the link it provides:
gcloud app browse

Copied!

content_copy

Example output (note that your link will be different):

Did not detect your browser. Go to this link to view your app:

<https://qwiklabs-gcp-233dca09c0ab577b.appspot.com>



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



NUTAN COLLEGE OF ENGINEERING & RESEARCH (NCER)
Department of Computer Science & Engineering (CSE)

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

Deploy your app.

Check my progress

Conclusion:

In this exercise, we have successfully deployed a Python application using Google App Engine, a powerful Platform as a Service (PaaS) offering from Google Cloud Platform (GCP). We have leveraged GitHub for version control and source code management, demonstrating the effective use of cloud-based tools in a real-world development workflow. This hands-on experience has deepened my understanding of PaaS and its practical applications, particularly in the context of deploying Python applications on the cloud. The knowledge and skills acquired will be invaluable in future cloud computing endeavors.