PRACTICAL NO : 02

Title : Assignment to install and configure Google App Engine STEPS TO INSTALL & CONFIGURE :

Created By :Ms. Manisha Vilas Khadse

Before begin

1. Create a Google Cloud Platform project, if you don't have one already.

2. Make sure that Python 2.7 is installed on your system: python -V Note: As of Cloud SDK version 206.0.0, the gcloud CLI has experimental support for running using a Python 3.4+ interpreter (run gcloud topic startup for exclusions and more information on configuring your Python interpreter). All other Cloud SDK tools still requirea Python 2.7 interpreter.

3. Download the archive file best suited to your operating system. Most machines will run the 64-bit package. If you'd like to check, run uname -m to verify if you're running a 64-bit system. Platform Package Size SHA256 Checksum Linux For 64-bit google-cloud-sdk25.6 b1c87fc9451598a76cf66978dd8aa06482bfced639b56cf31559dc2c7f8b7b90 229.0.0-linuxMB For 32-bit google-cloud-sdk25.2 ee8c45f8018d0fee92b07c32cc6d8c891241da0b88bfe289d4e58e6746c3f668 229.0.0-linuxMB (x86) x86.tar.gz Alternatively, to download the Linux 64-bit archive file from your command-line, run: curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloudsdk-229.0.0-linux-x86_64.tar.gzcurl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloudsdk-229.0.0-linux-x86.tar.gz

4. Extract the archive to any location on your file system; preferably, your Home folder. On Linux, you can extract the archive file by running this command: tar zxvf [ARCHIVE_FILE] google-cloud-sdk

5. If you're having trouble getting the gcloud command to work, ensure your $PATH is defined appropriately. Use the install script to add Cloud SDK tools to your path. You will also be able to opt-in to command-completion for your bash shell and usage statistics collection during the installation process.

Run the script using this command:

./google-cloud-sdk/install.sh

Restart your terminal for the changes to take effect. Alternatively, you can call Cloud SDK after extracting the downloaded archive by invoking its executables via the full path.

Initialize the SDK Use the gcloud init command to perform several common SDK setup tasks. These include authorizing the SDK tools to access Google Cloud Platform using your user account credentials and setting up the default SDK configuration.

To initialize the SDK:

1. Run the following at a command prompt: gcloud init Note: To prevent the command from launching a web browser, use gcloud init -- console- only instead.
To authorize without a web browser and non-interactively, create a service account with the appropriate scopes using the Google Cloud Platform Console and use gcloud auth activate-service-account with the corresponding JSON key file.
2. Accept the option to log in using your Google user account: To continue, you must log in. Would you like to log in (Y/n)? Y
3. In your browser, log in to your Google user account when prompted and click Allow to grant permission to access Google Cloud Platform resources.
4. At the command prompt, select a Cloud Platform project from the list of those where you have Owner, Editor or Viewer permissions:

Pick cloud project to use:
[1] [my-project-1]
[2] [my-project-2] ..

. Please enter your numeric choice: If you only have one project, gcloud init selects it for you.

5. If you have the Google Compute Engine API enabled, gcloud init allows you to choose adefault Compute Engine zone:

Which compute zone would you like to use as project default?

[1] [asia-east1-a]
[2] [asia-east1-b] ..

. [14] Do not use default zone Please enter your numeric choice:

gcloud init confirms that you have complete the setup steps successfully:gcloud has now been configured!

You can use [gcloud config] to change more gcloud settings. Your active configuration is: [default]

Run core gcloud commands

Run these gcloud commands to view information about your SDK installation:

1. To list accounts whose credentials are stored on the local system:gcloud auth list gcloud displays a list of credentialed accounts: Credentialed Accounts ACTIVE ACCO UNT * example-user1@gmail.com exampleuser-2@gmail.com

2. To list the properties in your active SDK configuration:

gcloud config list gcloud displays the list of properties: [core] account = example-user1@gmail.com disable_usage_reporting = False project = example-project

3. To view information about your Cloud SDK installation and the active SDK configuration: gcloud info gcloud displays a summary of information about your Cloud SDK installation. This includes information about your system, the installed SDK components, the active user account and current project, and the properties in the active SDK configuration.

4. To view information about gcloud commands and other topics from the command line: gcloud help For example, to view the help for gcloud compute instances create: gcloud help compute instances create gcloud displays a help topic that contains a description of the command, a list of commandflags and arguments, and examples of how to use it.

How to Run Program:
Now as we have finished installing app engine, now it's time to createand upload an app. In this case we will be taking example of a "HELLO WORLD" app in python.

1. As we already have made sure that we have python installed in our system, It will be easier for usto clone existing code and deploy it rather than creating our own so we will use python docs- sample. Run the command "git clone https://github.com/GoogleCloudPlatform/pythondocs- samples".

2. cd python docs- samples/appengine/standard/hello_world

3. dev_appserver.py app.yaml It will run and give you the url of default and admin.
If you go to the link of default you see the text hello world like this.

Output :

OR

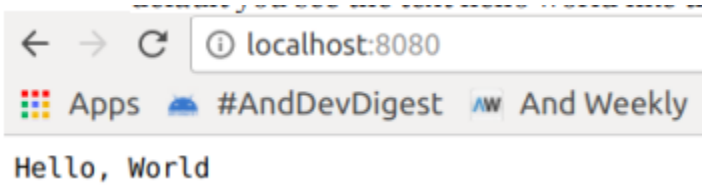
Google App Engine, Installing it the right way in ubuntu.

1. Make sure you have python installed in your ubuntu system. run the command “python -V” and most probably you will get “Python 2.7.6” or above.

2. Crul https://sdk.cloud.google.com and use bash to run the commands by typing this command curl https://sdk.cloud.google.com | bash
3. Whenever you get to choose directories just hit enter, “YEAH IT WILL BE FINE”.
4. Follow the instructions in the installation process.
5. Then run gcloud init
6. Follow the installation instructions as they are very straight forward.
7. Choose the account you want to use for google app engine.
8. Choose the project with numeric choice (don’t use textual, you might make mistake). If you do not already have a google app engine project create a app engine project by following this link. https://console.cloud.google.com/start
9. Enable google api by pressing Y in the command line prompt. Now as we have finished installing appengine, now it’s time to create and upload an app. In this case we will be taking example of a “HELLO WORLD” app in python.

1. As we already have made sure that we have python installed in our system, It will be easier for us to clone existing code and deploy it rather than creating our own so we will use pythondocs-sample. Run the command “git clone https://github.com/GoogleCloudPlatform/python-docssamples”.

2. cd to hello world sample by typing the command “ cd pythondocs-samples/appengine/standard/hello_world”.
3. Then run the command “dev_appserver.py app.yml”. It will run and give you the url of default and admin. If you go to the link of default you see the text hello world like this.



This is how you run the python app in your local server. But what we have to do is hosting the app in google app engine. To do so Now let’s follow the following instructions.

1. Run the command Ctrl + C .
2. Being in the same working directory hello-world runt he command gcloud app deploy
3. Select the project you want to deploy the app , press Y and enter to continue. after that you will get the console output “Deployed service[default] to [Your web url for appengine] ”
4. If you copy and paste the url, you will see the hello world in the browser too.