**Practical : 9**

**Aim: - Google App Engine(GAE)**

**Submitted by: Samip Shah**

**Enrollment No: 15012011045**



**GANPAT UNIVERSITY**

**U. V. Patel College of Engineering**

**Computer Engineering Department**

* **Purpose Of The Project:**

Understand how the Google App Engine (GAE) works, what are the different components that are essential to GAE, get familiar with such components and how to host a web application over GAE.

* **Definition**

Google App Engine (GAE) is a service for developing and hosting Web applications in Google's data centers, belonging to the platform as a service (PaaS) category of cloud computing. Web applications hosted on GAE are sandboxed and run across multiple servers for redundancy and allowing for scaling of resources according to the traffic requirements of the moment. App Engine automatically allocates additional resources to the servers to accommodate increased load.

* **Introduction**

Google App Engine is Google's platform as a service offering that allows developers and businesses to build and run applications using Google's advanced infrastructure. These applications are required to be written in one of a few supported languages, namely: Java, Python, PHP and Go. It also requires the use of Google query language and that the database used is Google Big Table. Applications must abide by these standards, so applications either must be developed with GAE in mind or else modified to meet the requirements.

GAE is a platform, so it provides all of the required elements to run and host Web applications, be it on mobile or Web. Without this all-in feature, developers would have to source their own servers, database software and the APIs that would make all of them work properly together, not to mention the entire configuration that must be done. GAE takes this burden off the developers so they can concentrate on the app front end and functionality, driving better user experience.

* **Benefits**

For All Language Communities

Out of the box, App Engine supports Node.js, Java, Ruby, C#, Go, Python, and PHP. Developers from these language communities can be productive immediately in a familiar environment: just add code.

### Just Add Code

Nothing gets between you and shipping high-quality code. Offload infrastructure concerns like scaling your app up or down to handle traffic, load balancing, health-checking and healing your instances, and applying updates to the underlying OS—all fully managed on your behalf by Google.

### Infrastructure When You Need It

Enjoy a fully managed developer experience, or drop down into infrastructure for high-control when you need it. SSH directly into your instances to deploy custom code, manage containers, or integrate App Engine into your DevOps process. Specify your app’s CPU and memory requirements, and App Engine will automatically provision the infrastructure on your behalf.

### Growing Ecosystem of Services

App Engine is designed so that one can tap into a growing ecosystem of managed services, just an API call away. Choose from [SQL and NoSQL databases](https://cloud.google.com/products/storage/), [monitoring & diagnostics](https://cloud.google.com/stackdriver/) tools, popular services like [Cloud Pub/Sub](https://cloud.google.com/pubsub/), and a range of [great developer tools](https://cloud.google.com/products/tools/) on GCP(Google Cloud Platform).

* **Features**

Open & Flexible

Custom runtimes allow you to bring any library and framework to App Engine by supplying a Docker container

Fully Managed

A fully managed environment lets you focus on code while App Engine manages infrastructure concerns

Monitoring, Logging & Diagnostics

[Google Stackdriver](https://cloud.google.com/stackdriver/) gives you powerful application diagnostics to debug and monitor the health and performance of your app

Application Versioning

Easily host different versions of your app, easily create development, test, staging, and production environments

Traffic Splitting

Route incoming requests to different app versions, A/B test and do incremental feature rollouts

Application Security

Help safeguard your application by defining access rules with App Engine firewall and leverage managed SSL/TLS certificates\* by default on your custom domain at no additional cost

* **GAE Architecture**

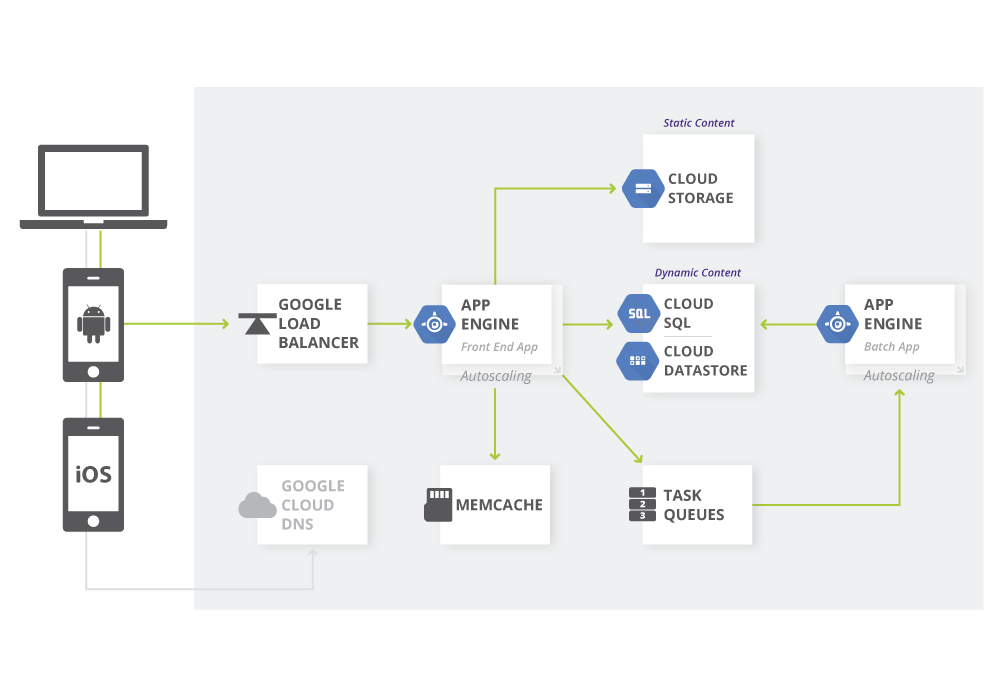


Figure 1Web Apps in the Google App Engine Standard Environment

App Engine is Google’s PaaS platform, a robust development environment for applications written in Java, Python, PHP and Go. The SDK for App Engine supports development and deployment of the application to the cloud. App Engine supports multiple application versions, which enables easy rollout of new application features and traffic splitting to support A/B testing.

The Memcache and Task Queue services are integrated in the App Engine standard environment. [Memcache](https://cloud.google.com/appengine/docs/standard/python/memcache/) is an in-memory cache shared across the App Engine instances. This provides extremely high speed access to information cached by the web server (e.g. authentication or account information).

[Task Queues](https://cloud.google.com/appengine/docs/standard/python/taskqueue/) provide a mechanism to offload longer running tasks to backend servers, freeing the front end servers to service new user requests. Finally, App Engine features a built-in load balancer (provided by the Google Load Balancer) which provides transparent Layer 3 and Layer 7 load balancing to applications.

## Cloud Endpoints Components

### Extensible Service Proxy

The Extensible Service Proxy is a NGINX-based proxy that runs in front of the backend and injects Cloud Endpoints functionality such as authentication, monitoring and logging. The proxy retrieves a service configuration from Google Service Management and uses it to perform validation on incoming requests.

The proxy is designed to be deployed in a containerized environment and validate JWTs and Google ID tokens. It employs a variety of techniques such as heavy caching and asynchronous calls to remain lightweight and highly performant.

### Google Service Control

[Service Control](https://cloud.google.com/service-control/) applies API management rules at runtime such as key authentication, monitoring, and logging. It has two main features:

* Check - verifies authentication and API keys, and indicates whether a call should be permitted
* Report - notifies the systems of record for logging and monitoring

### Google Service Management

[Cloud Endpoints](https://cloud.google.com/endpoints/) uses the OpenAPI specification to describe an API. Deploying this specification to Service Management - typically using Google Cloud SDK - configures the API management rules. Other configuration related tasks also happen here, such as sharing your API with other developers, enabling/disabling the API in different projects, and generating API keys.

### Google Cloud SDK

[Cloud SDK](https://cloud.google.com/sdk/) provides the gcloud command-line application that can be used to make calls to various services of Google Cloud Platform. The gcloud command-line tool is also used to deploy your API Configuration to Service Management.

### Google Cloud Console

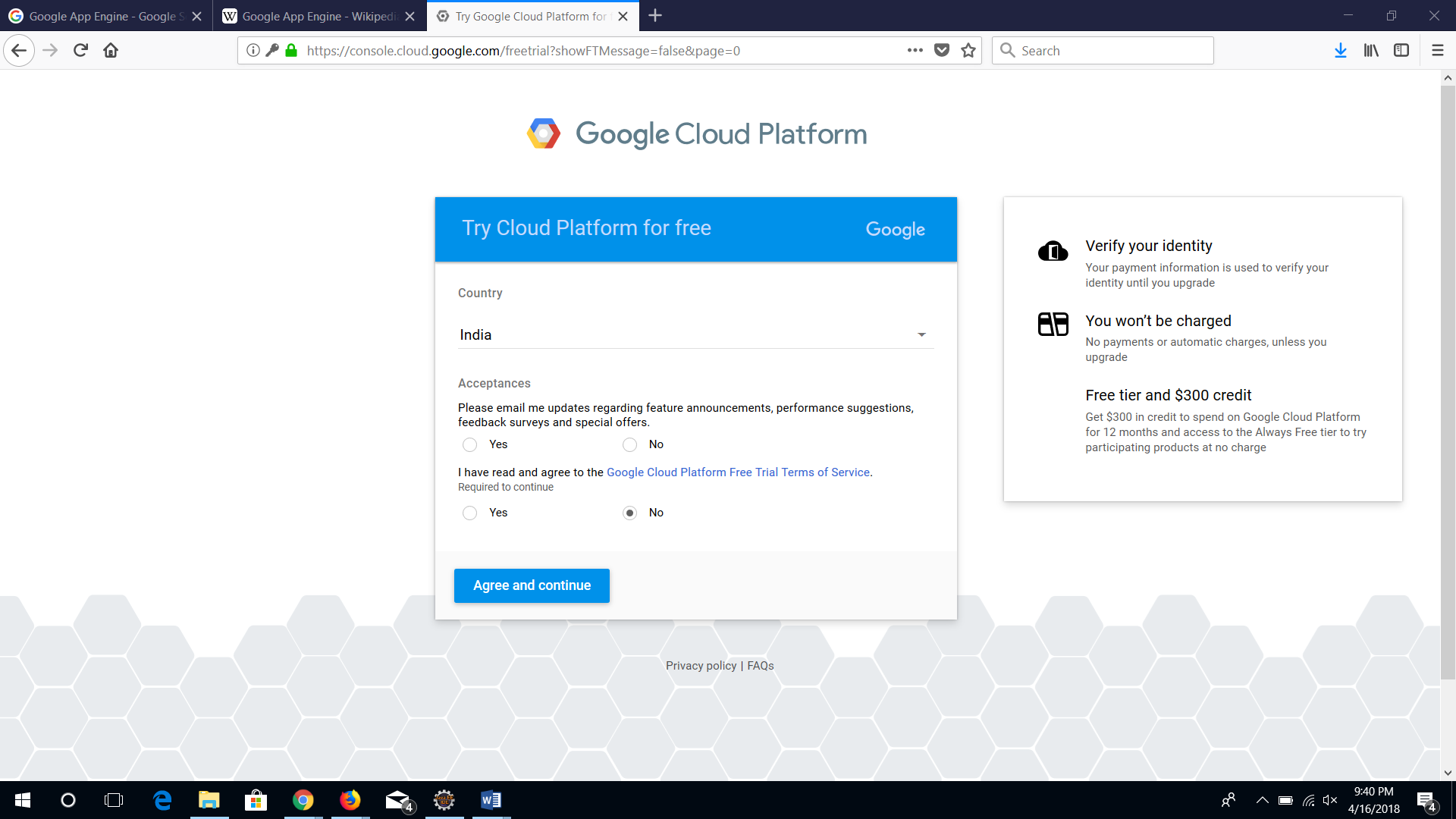
[Cloud Console](https://cloud.google.com/cloud-console/) is the graphical user interface for Google Cloud Platform. Cloud Endpoints uses Cloud Console to expose monitoring and logging data that are sent from the proxy and recorded by Service Control and share APIs with other developers, and for them to generate API keys to call the API.

* **Advantages**
* Will automatically handle and balance all instances and data centers for web applications.
* Provides support for 4 popular programming languages: PHP, Python, Java, and Go.
* Will automatically scale data storage or instances to meet the needs of the programmer.
* Provides App Engine SDK to produce applications locally.
* App Engine can be integrated into multiple IDEs.
* Has a secured Internet Infrastructure to store code and application data safely.
* **Disadvantages**
* If Google servers go down, applications will not run properly.
* Must be careful to read the long list of policies. Violation of Google AdSense policies are common and have resulted in bans.
* Everything is handled by Google: company data information and pricing changes are all in Google's hands.
* Not flexible for future opt-out. If you leave GAE, it is not possible to port your app and you will need to develop your application from scratch again.
* **How to Configure and Host a webapp using Java**

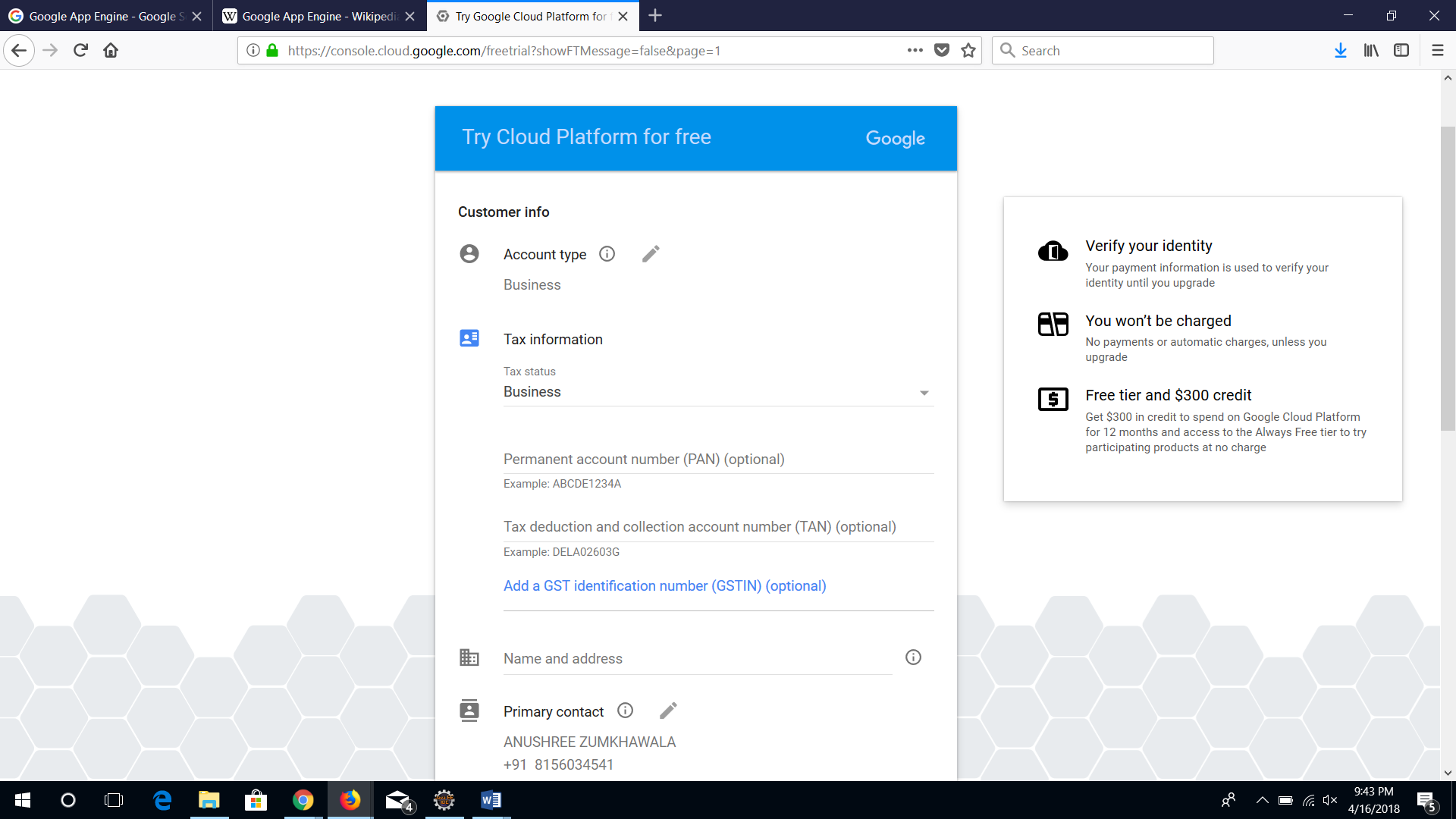
**Steps:-**

1. **Google Cloud Platform Account Creation**

First open Google Cloud Platform and login with your gmail account then it will ask you to start your free trial once you click that you will be asked to agree the terms and conditions, accep that .

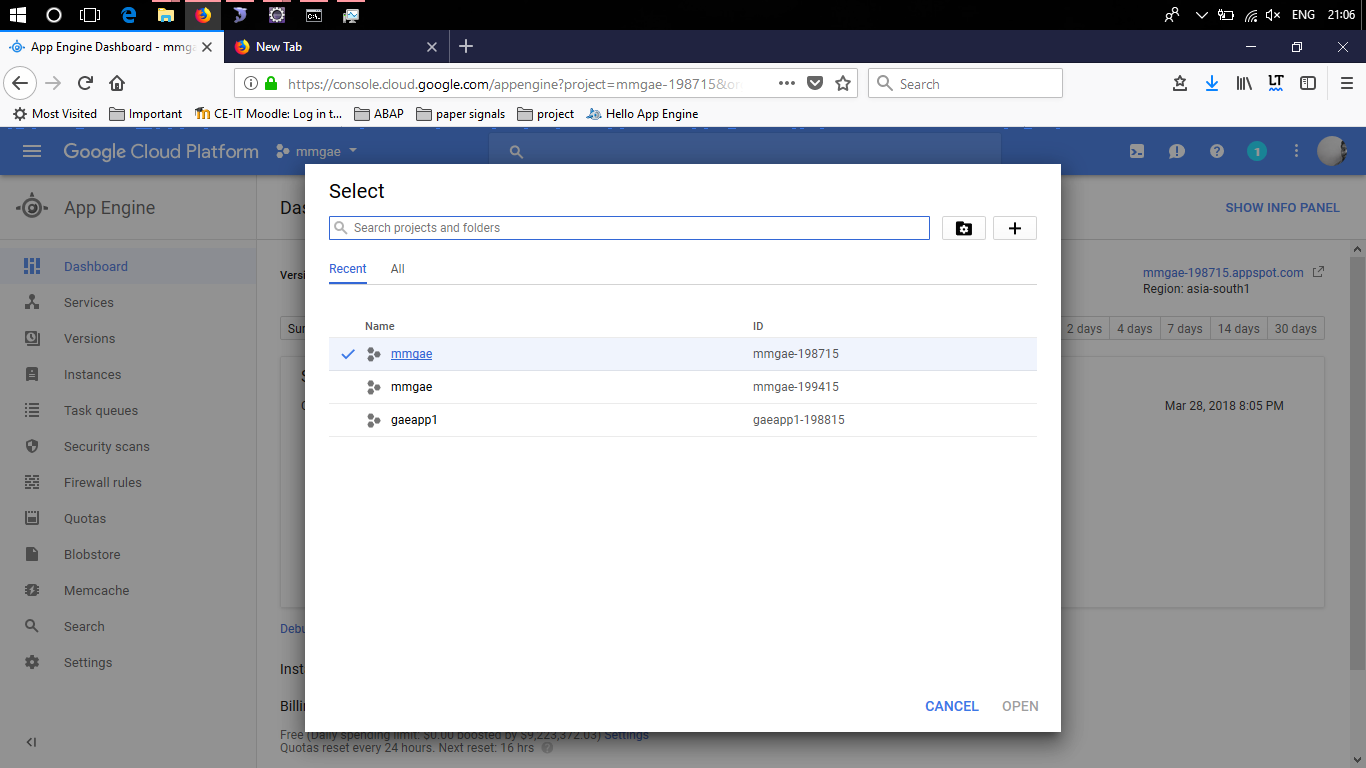


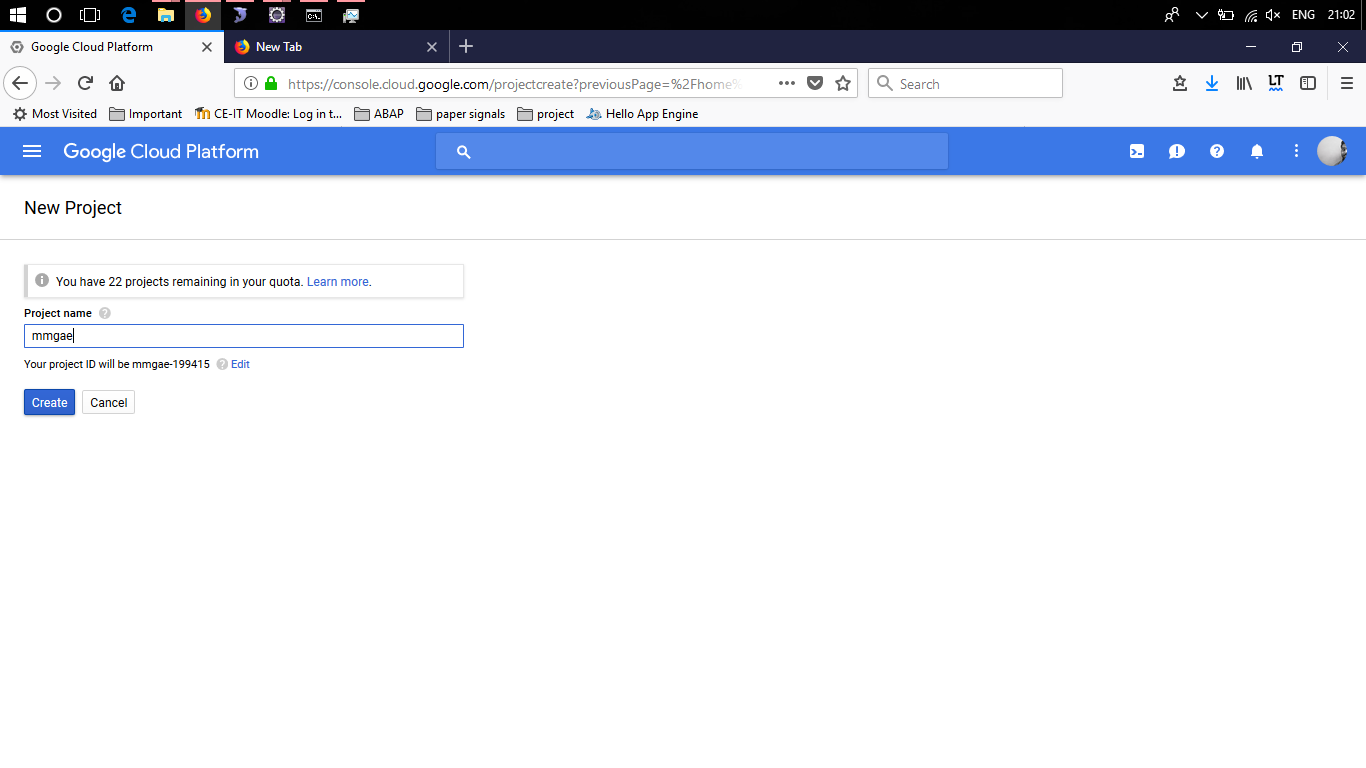
Then It will ask you to fill a form which will require personal details and debit/credit card information in case you exceed your trial limit i.e 20,000 credits or $300.



1. **Project Creation**

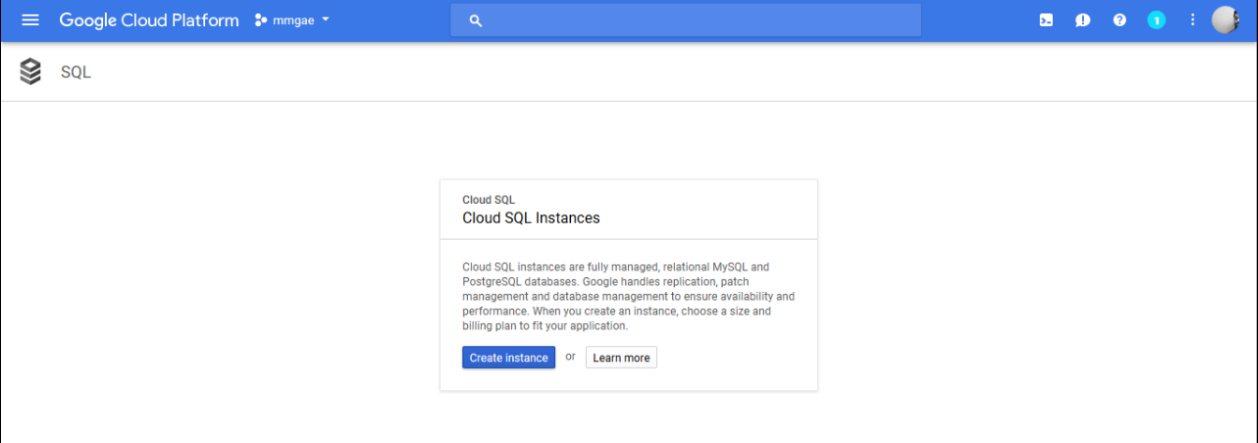
Once you have created your account, create a new App Engine project, which will require you to give it a name and according to it a project id will be generated which will be useful if you try to connect to app engine using SSL or Socket Factory.

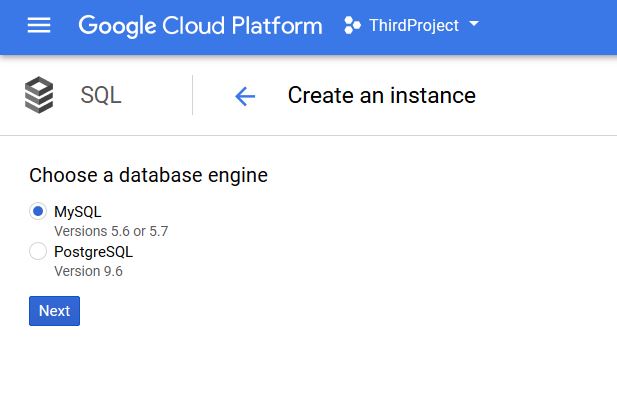
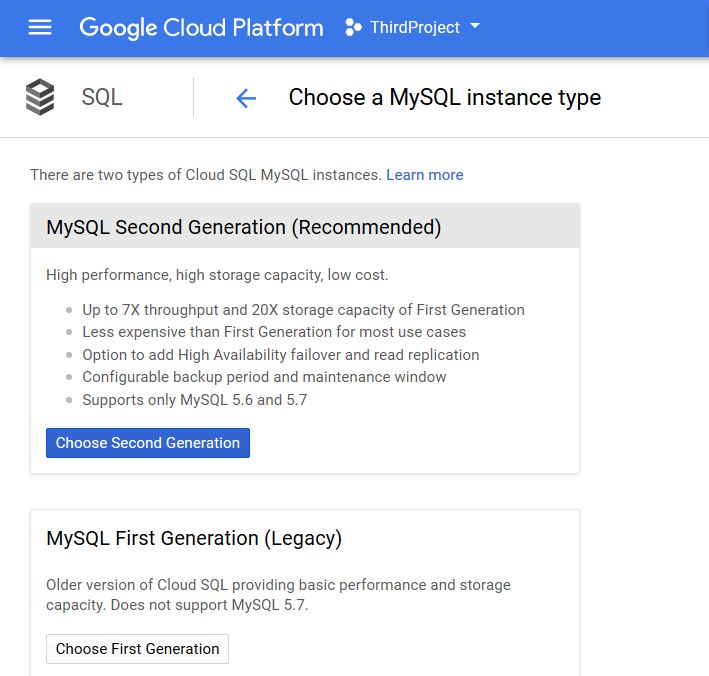




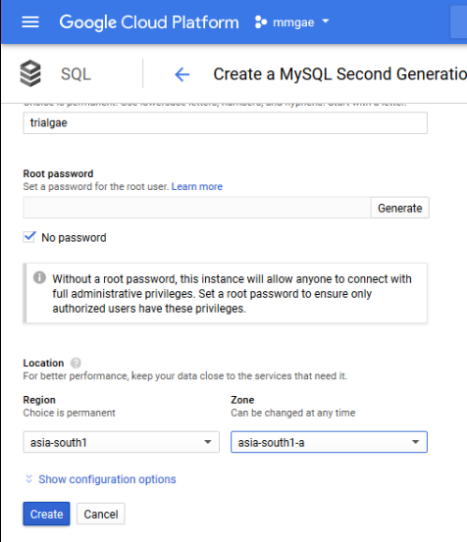
1. **Now Create Sql Instance**

Navigate to Cloud Sql from the navigation bar present on the left and then click Create Instance, then give it a name and you will get a IPv4 address this we will need to connect our mysql database to cloud sql. Choose Mysql.



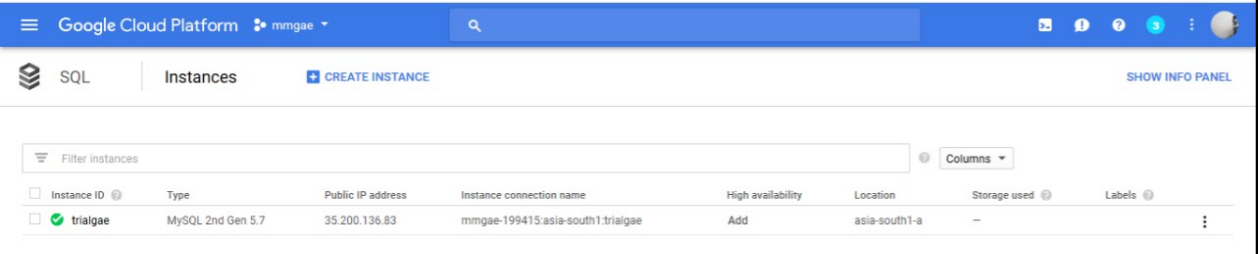
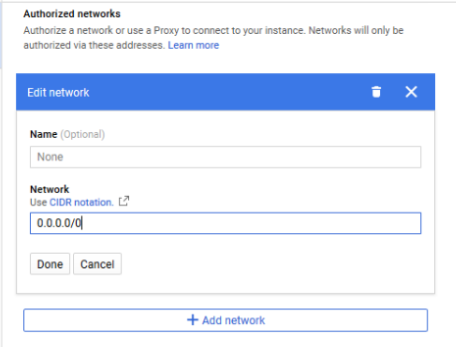
 

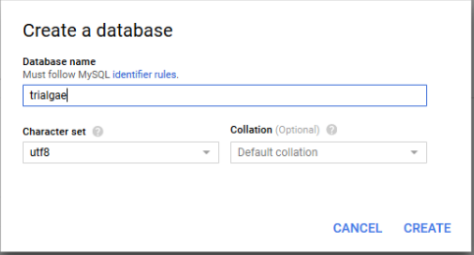
After this you would be directed towards a new page where you will need to give an instance id and password so that not everyone can have access to your SQL instance. Also choose a location nearest to your location for better performance.



1. **Creating Database and Setting Up Authorization**

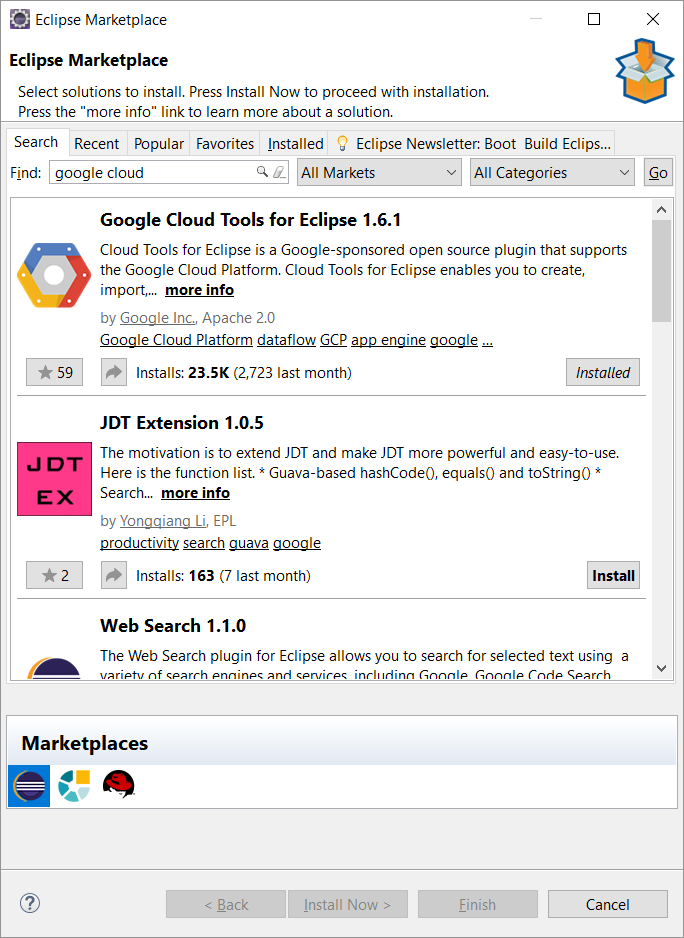
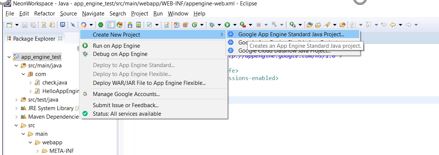
Now we have our IPv4 address and now we will create a database and set authorization to 0.0.0.0/0 so that everyone can access it. It is usually used for protection of data from outside sources and maintaining privacy. Finally save it and your SQL instance is ready.





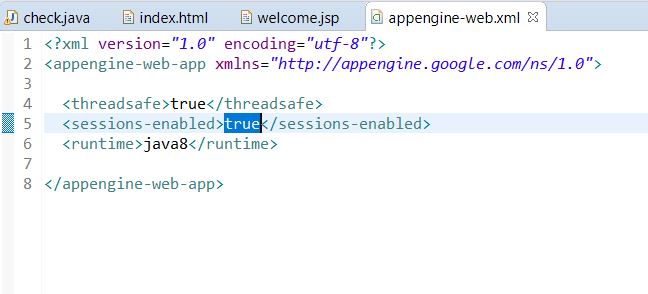
1. **Setting Up Eclipse**

First download the Google Cloud Platform Tools plugin from Eclipse Marketplace. Then you will get an icon in toolbar press that and create new App Engine Standard Project.

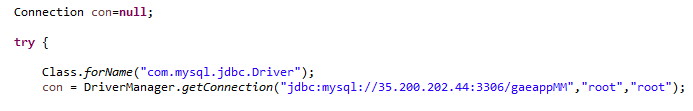
 

1. **Creating Webapp**

You can convert your existing project to app engine project or create a new app engine standard project as we say above. Create a Web app as you make any other web application but make sure if you are using session turn it on in appengine-web.xml.

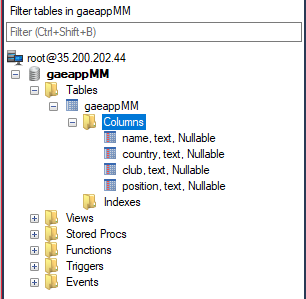
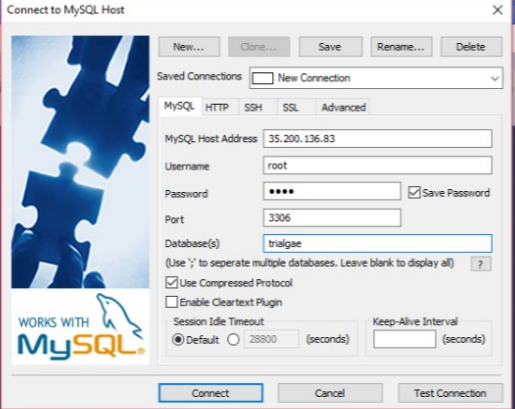


There is just a minor change in Connection Url, instead of localhost you need to give the IPv4 address of your SQl instance and their username and password respectively



1. **Setting up SQlYog (Mysql tool)**

Now when you open your SQLYog or MySQl Workbench click on new connection and put the SQL instance Ipv4 address in the host address and username and password in their respective fields. Now you can create your table.



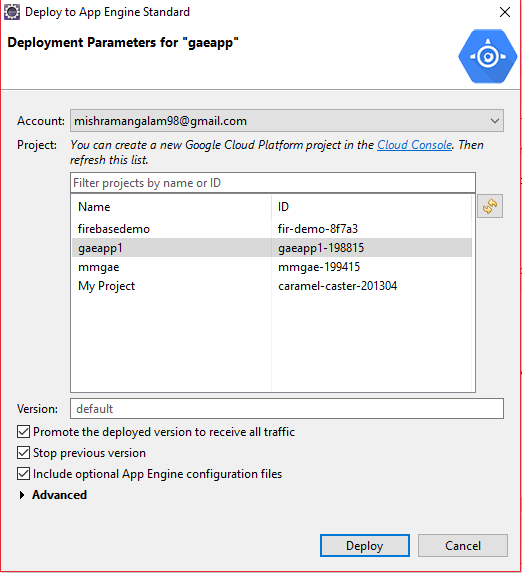
1. **Deploying To App Engine**

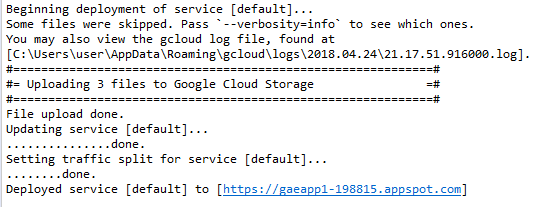
Now once your code is ready all that is left is to deploy the project on App Engine. However, initially if you want to check for errors Google Cloud Platform provides an App engine server where you can test your application and once you are sure deploy on APP Engine.

For deploying just right-click on project and select deploy to App Engine Standard..

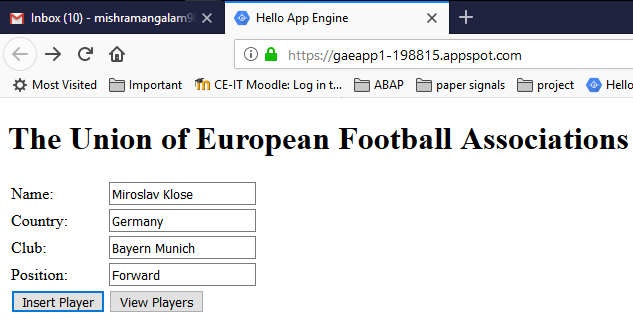
A Dialog Box will be opened where if you are using it for the first time it will ask you to login in your gmail account if it doesn’t then might already be logged in. Now select the App Engine Project you want your web Application to run on in this case ThirdProject and click Deploy.

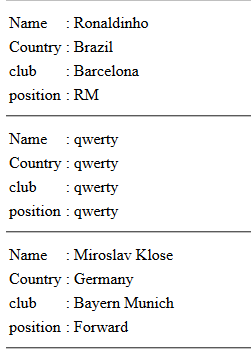
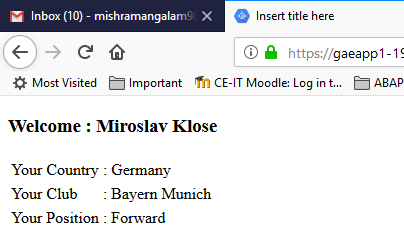
Now some Files would be uploaded to google cloud storage and then after the process is complete you will get an url on clicking which you can see your project running on Google App Enigne.





1. **Check Output**



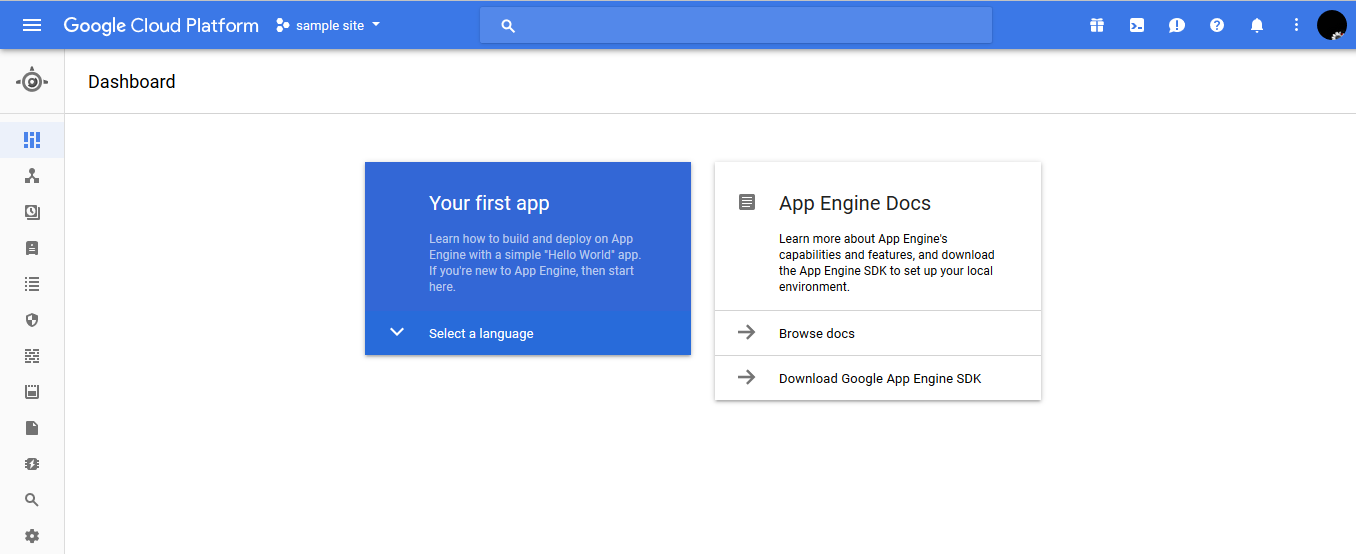


**Deploying a website with database on Google App Engine using Python.**

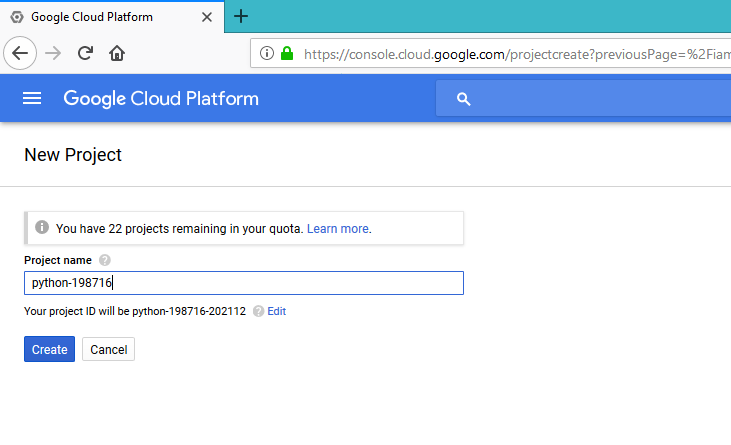
## Pre-Requirements

* Python 2.7
* Google App Engine Launcher
* Google Cloud SDK

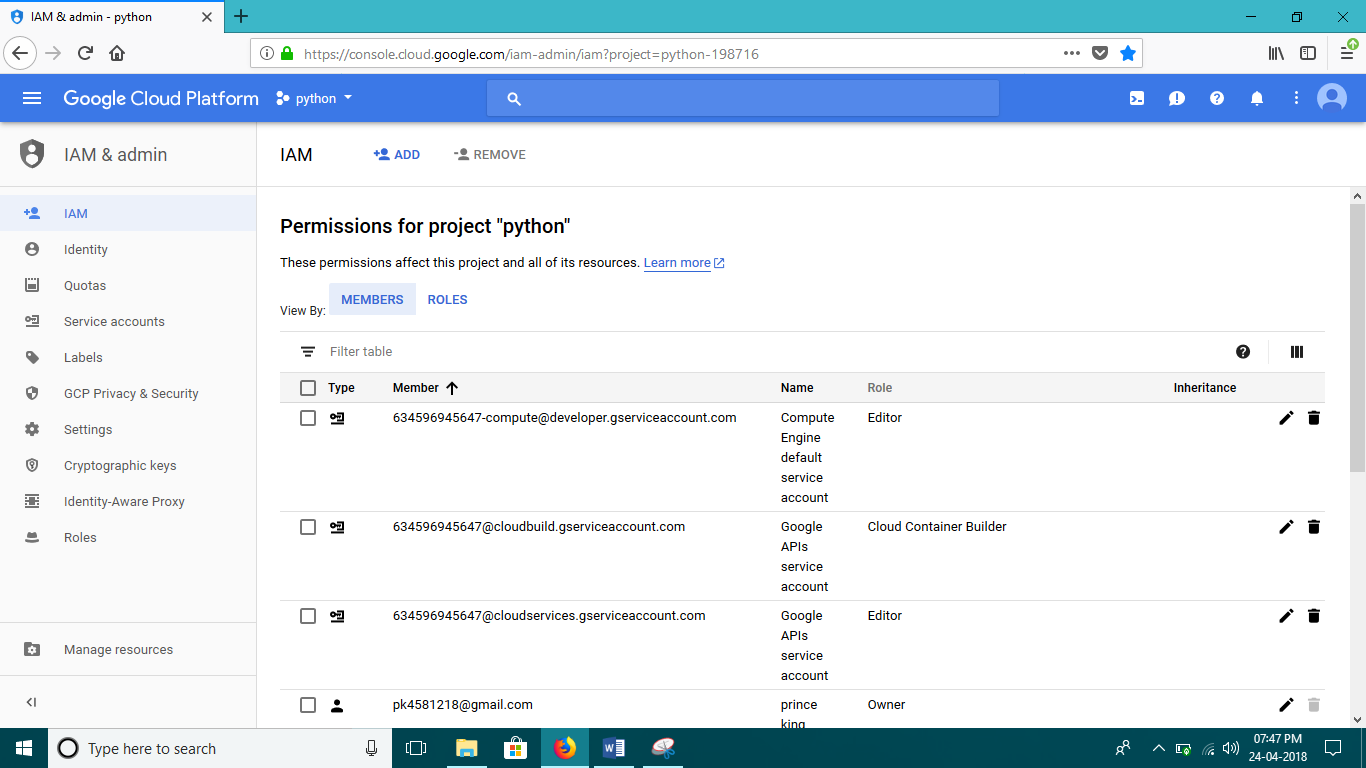
1. **Sign up with your existing Google account and you can see this HOME PAGE.**

****

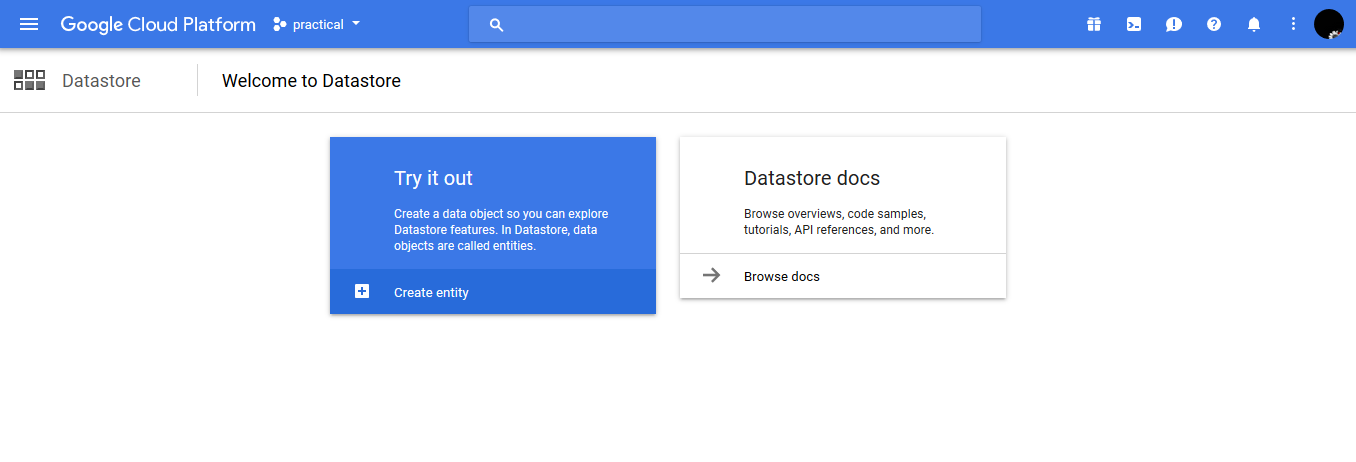
1. **Now for create new project click on “Create Project” here. Give Project Name and Project ID and Create it.**



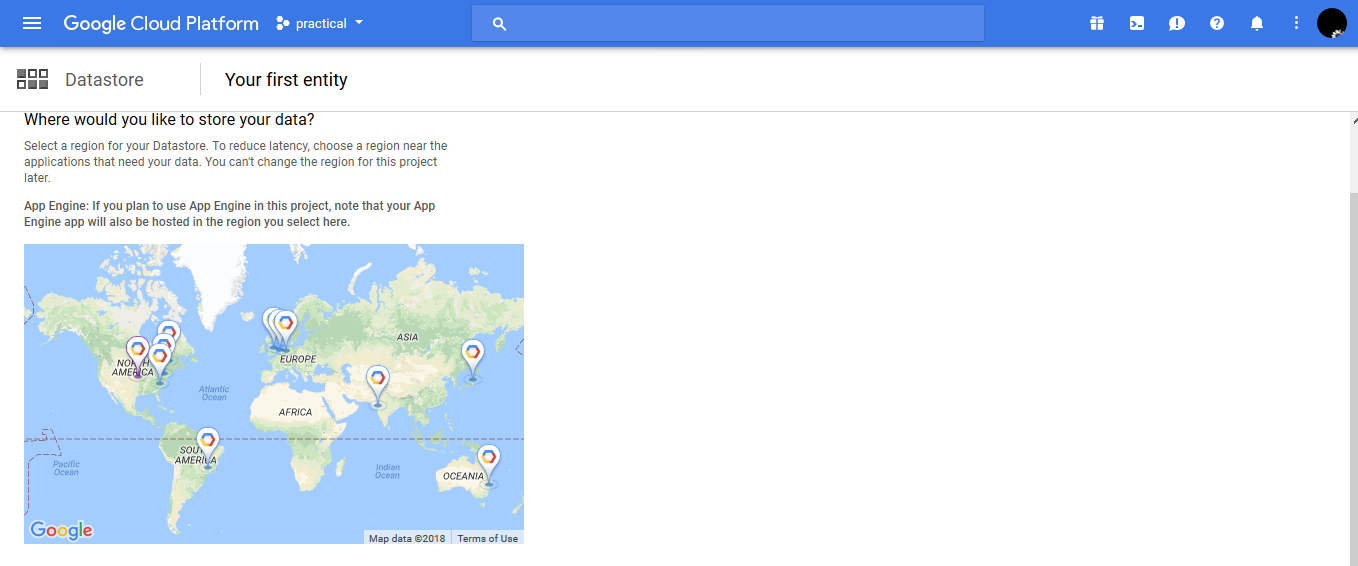
1. **Now you can see Home Page of your Project.**



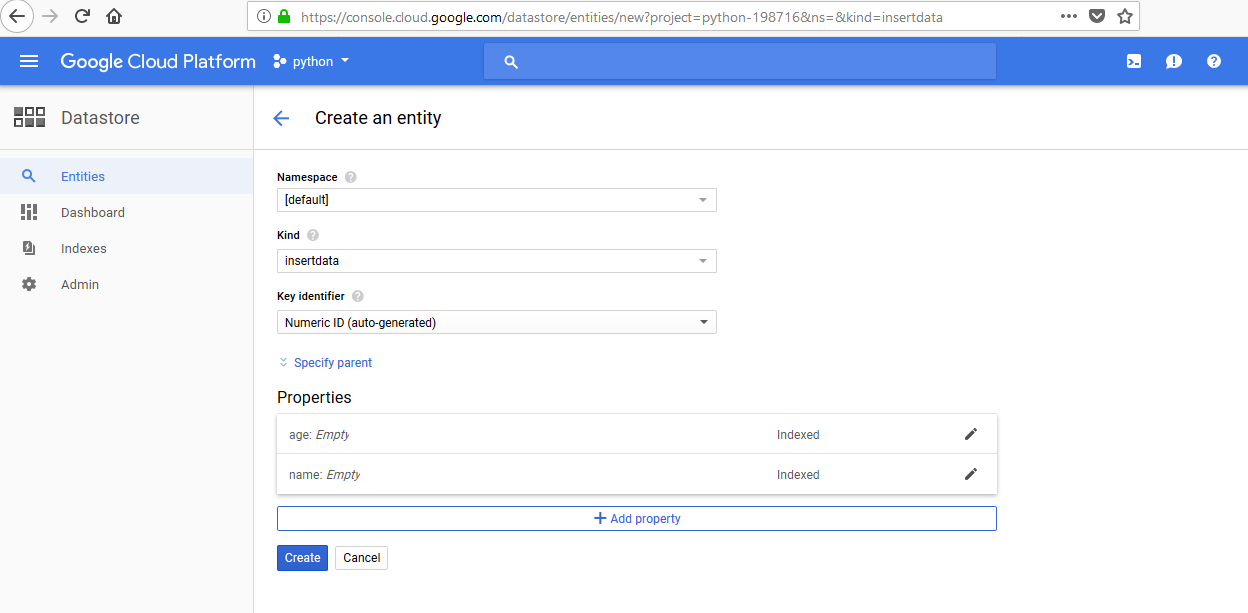
1. **Now go to the Menu and click on Data store.**



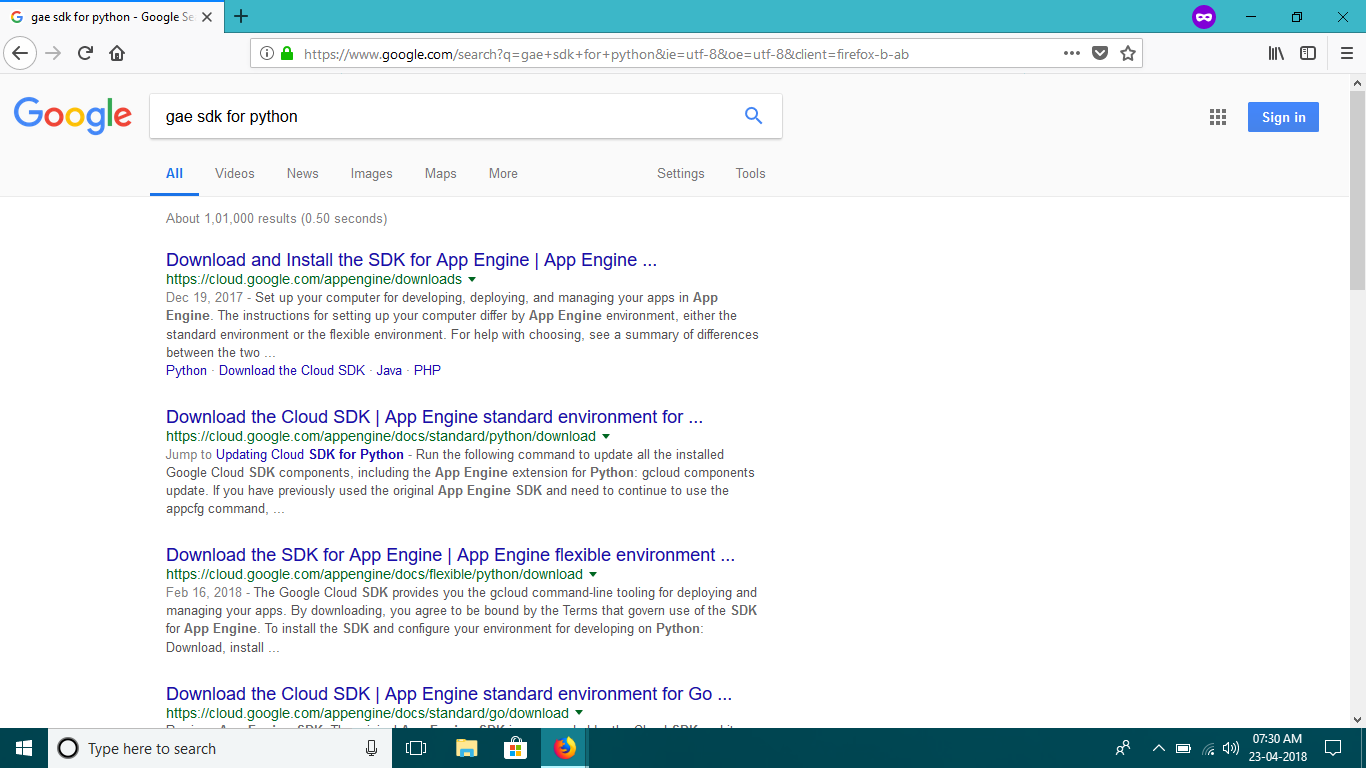
1. **Give region.**

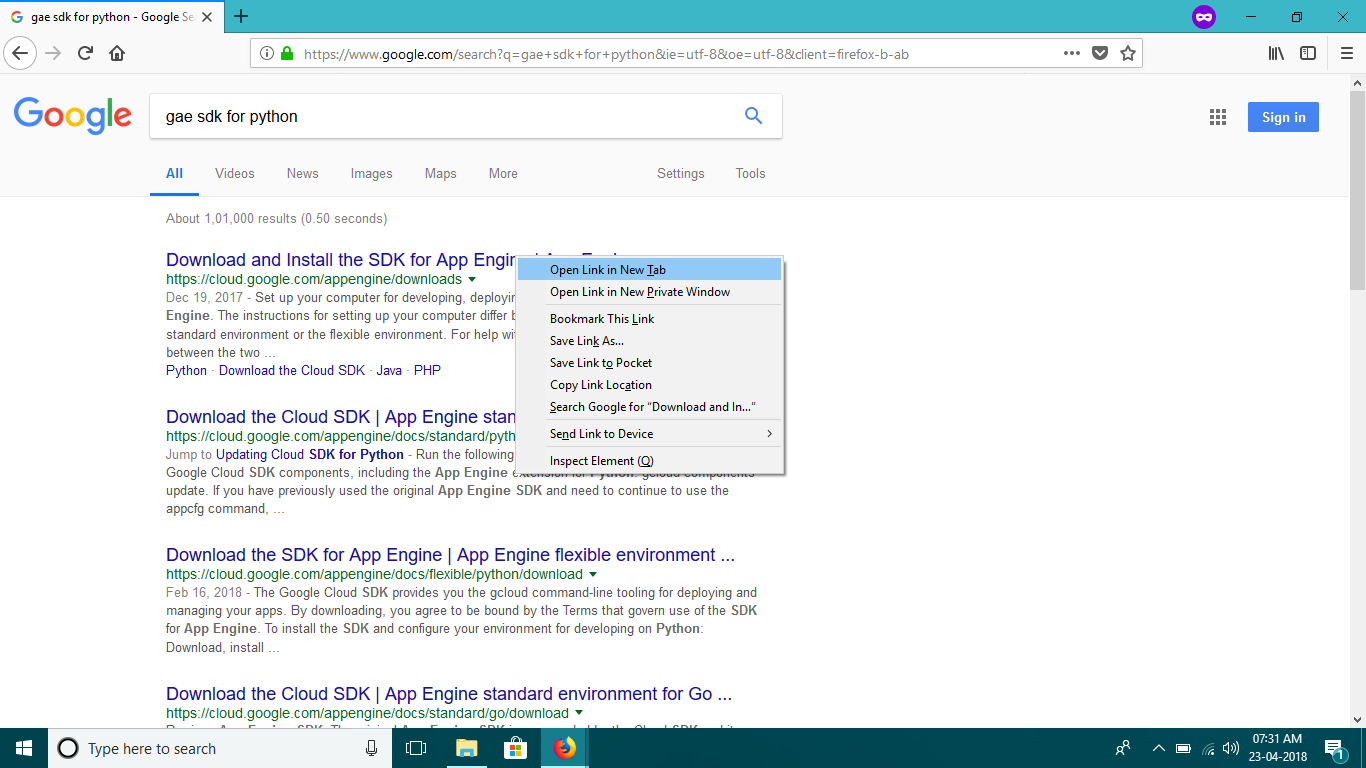


1. **Here keep namespace default and give name in “Kind” and Click on “Create Entity”. You can see the “+Add property” button and you can add your columns.**

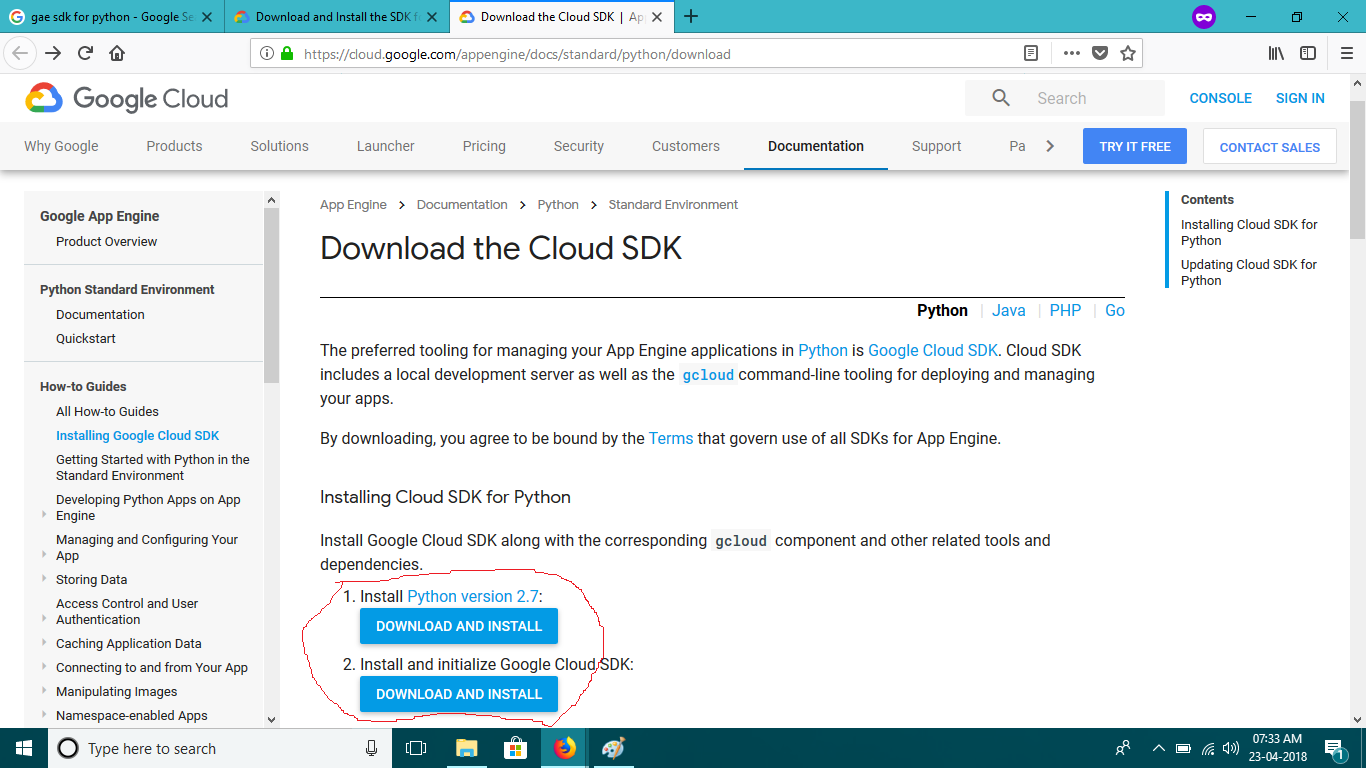


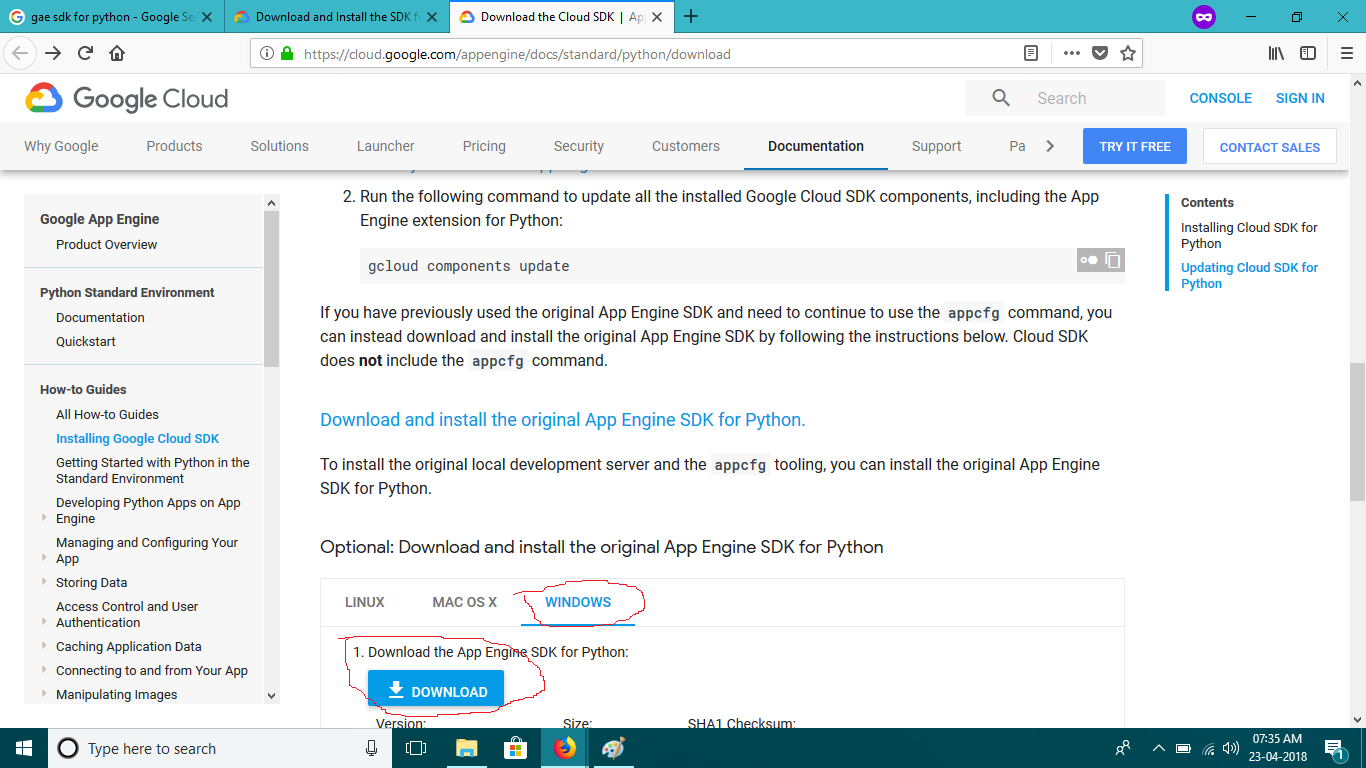
1. **Now open new tab and search ”gae sdk for python” and then follow steps as bellow.**



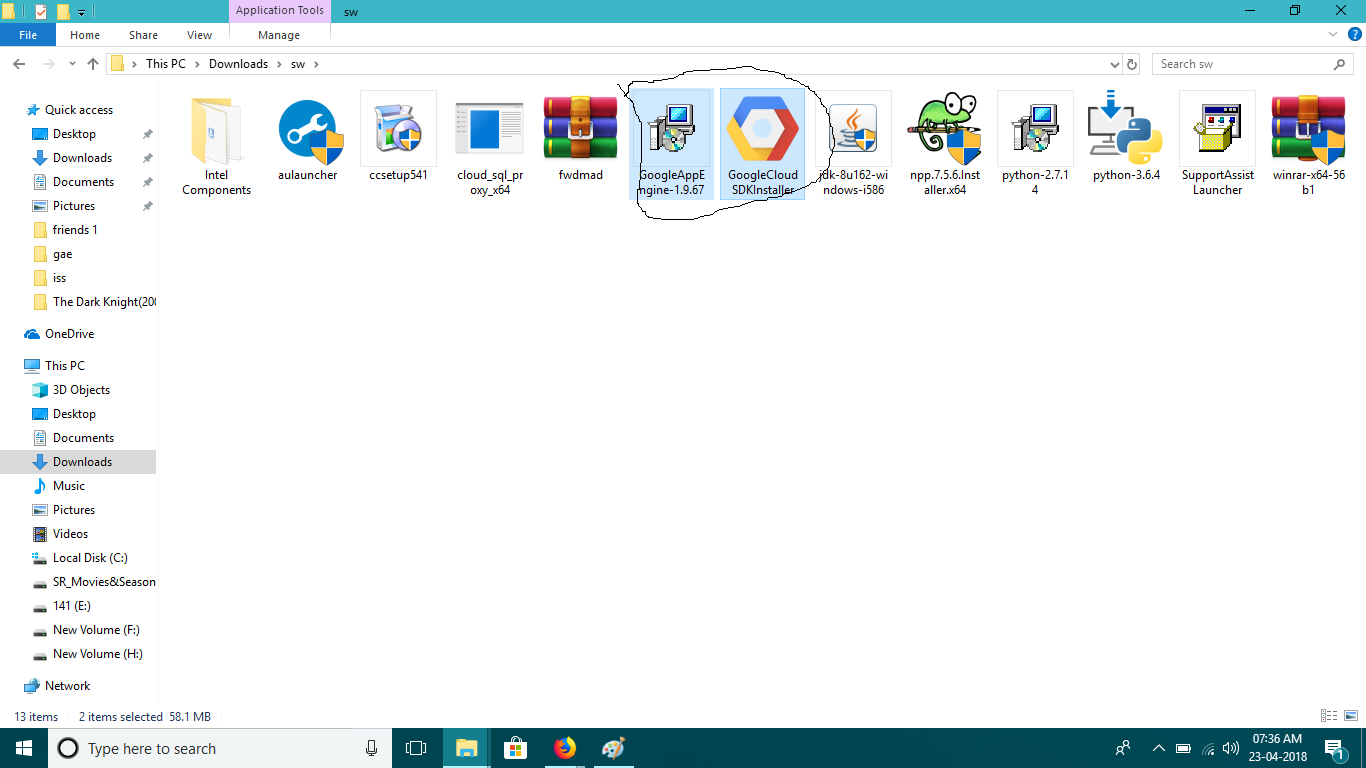


1. **In new tab download and install “python 2.7” , “Google Cloud SDK” and “Google App Engine SDK for Python”**

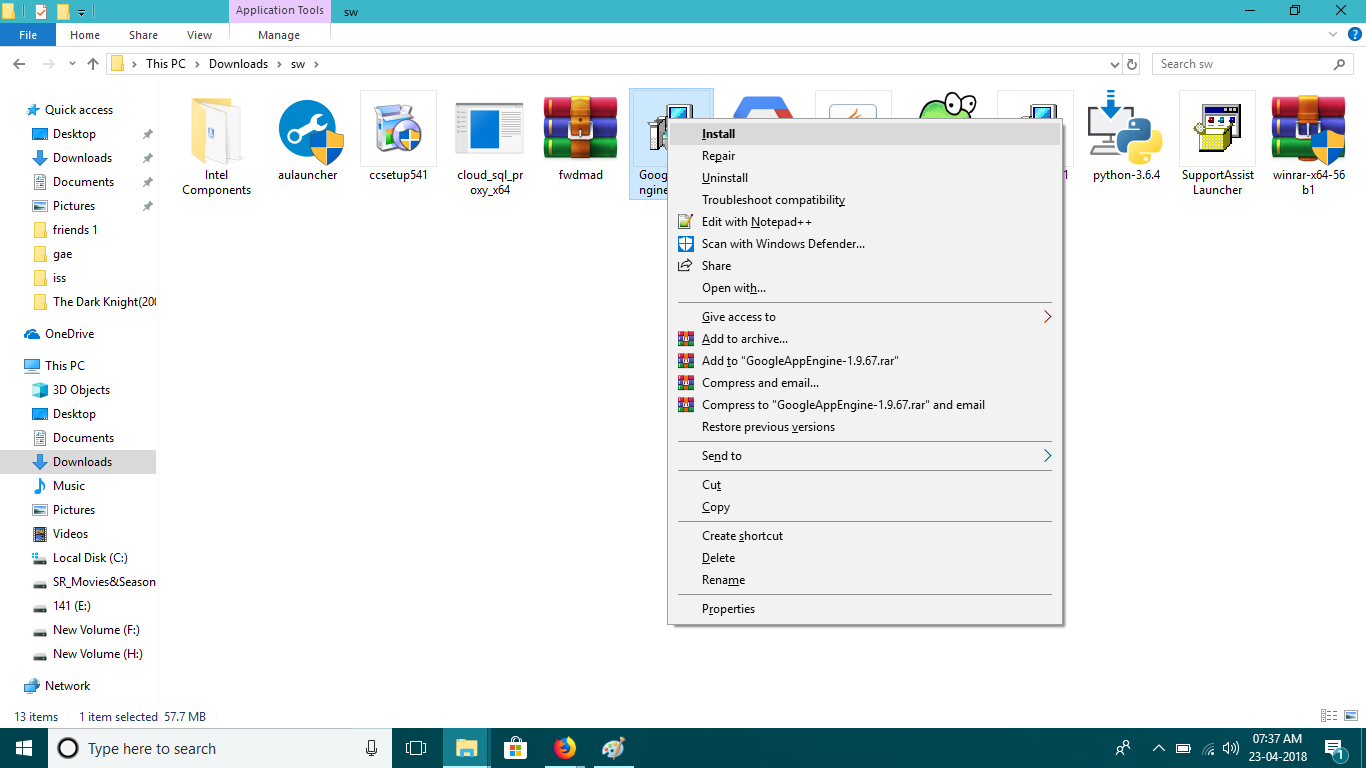


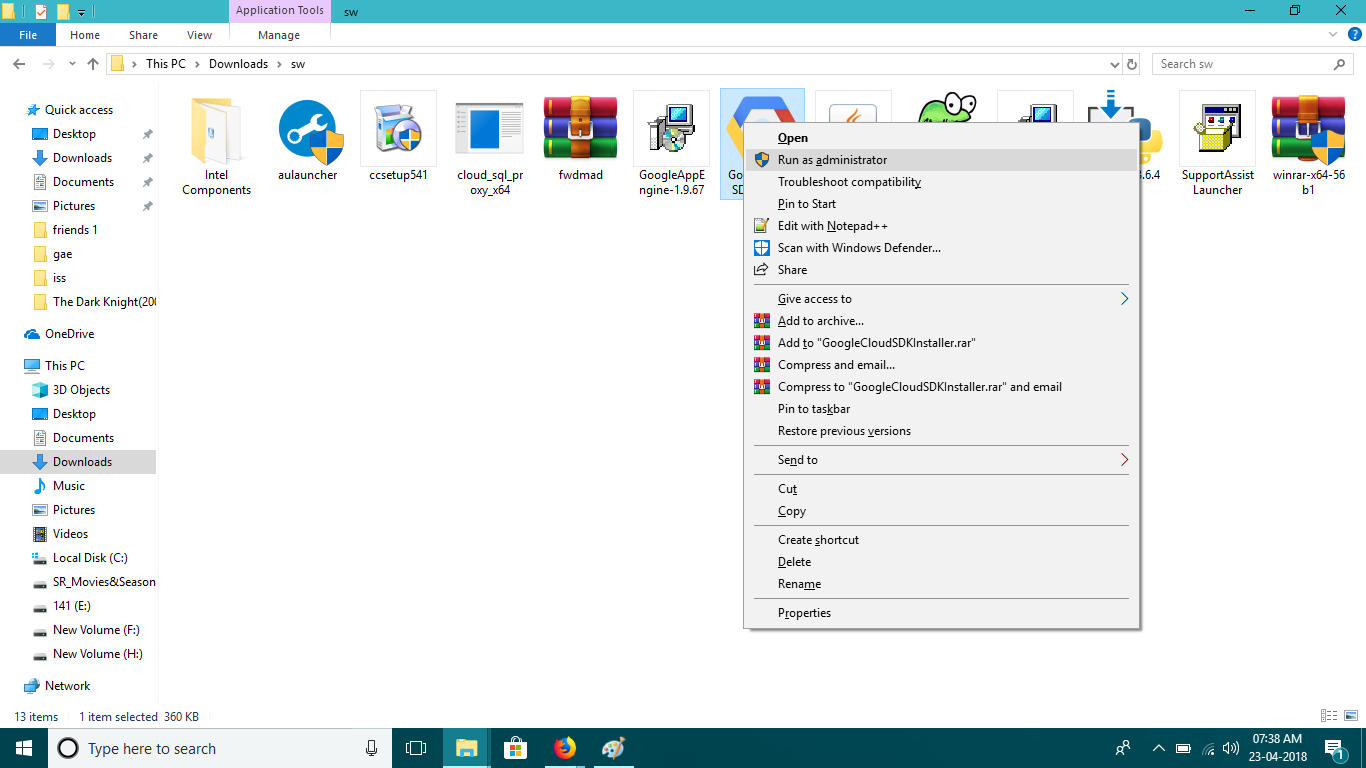


1. **Open File explore go to download you can see both items.**

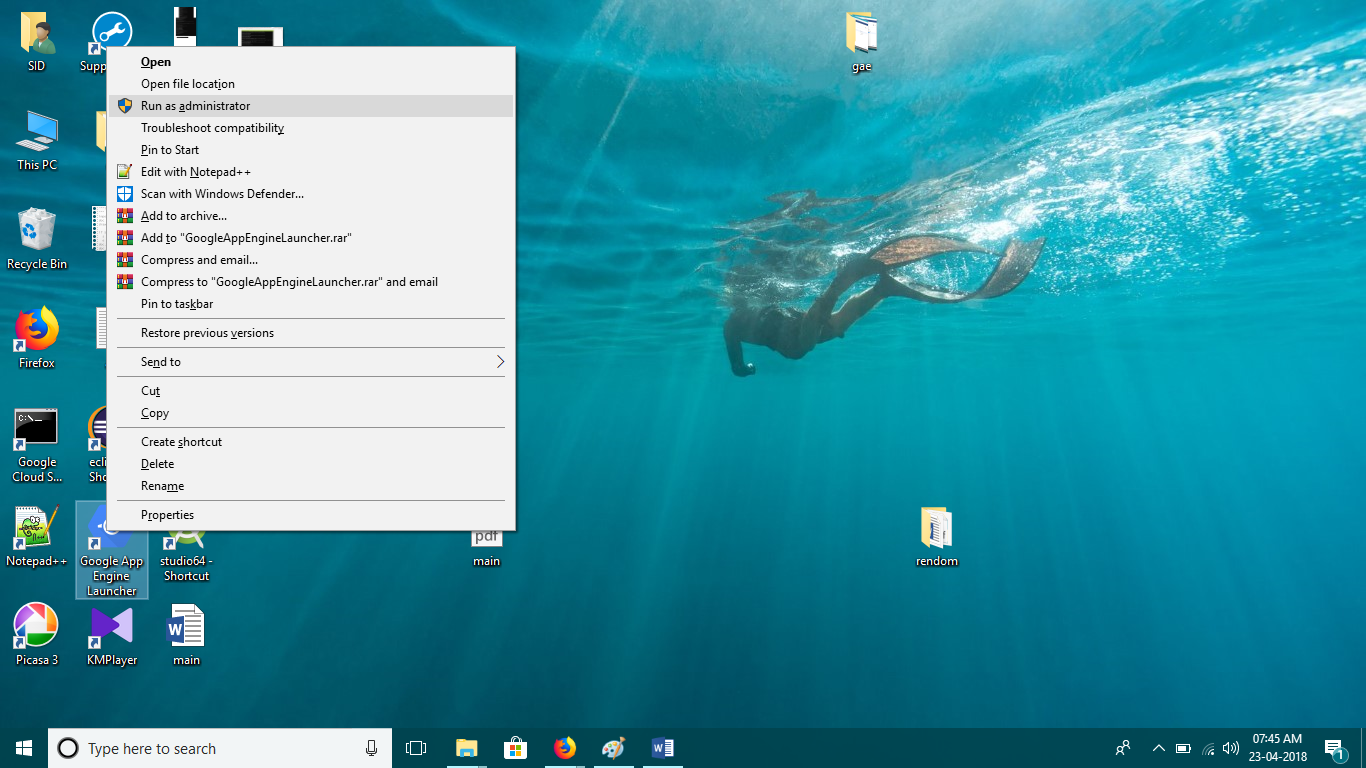


1. **Install them both.**

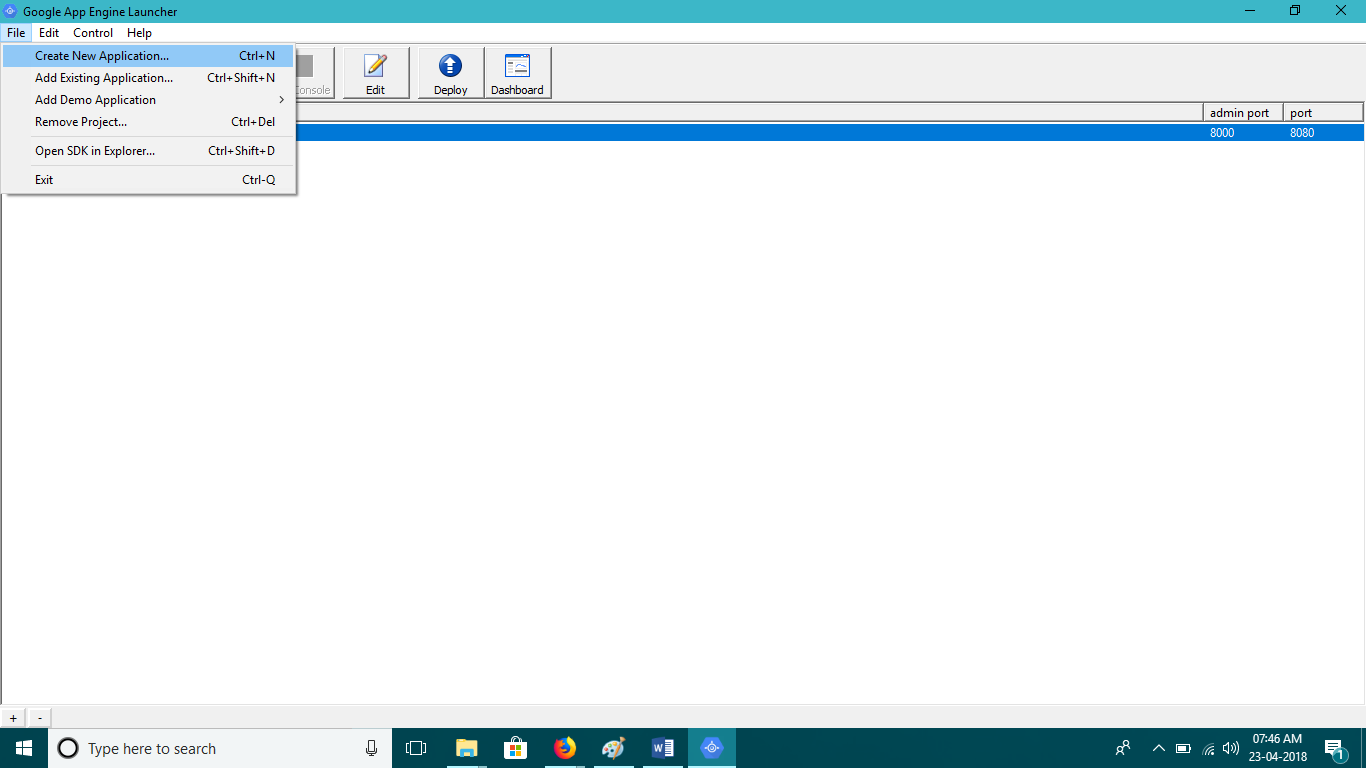




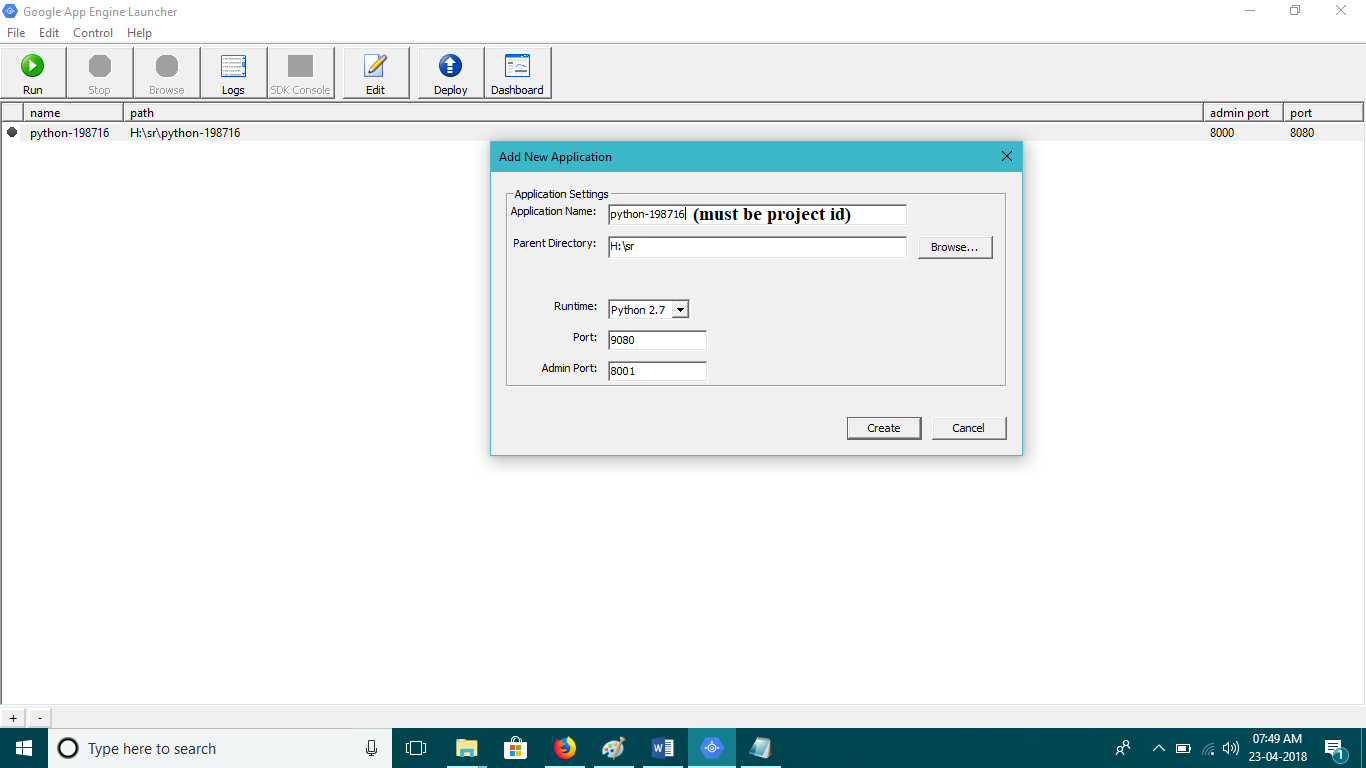
1. **After finishing installation run “google app engine launcher”**



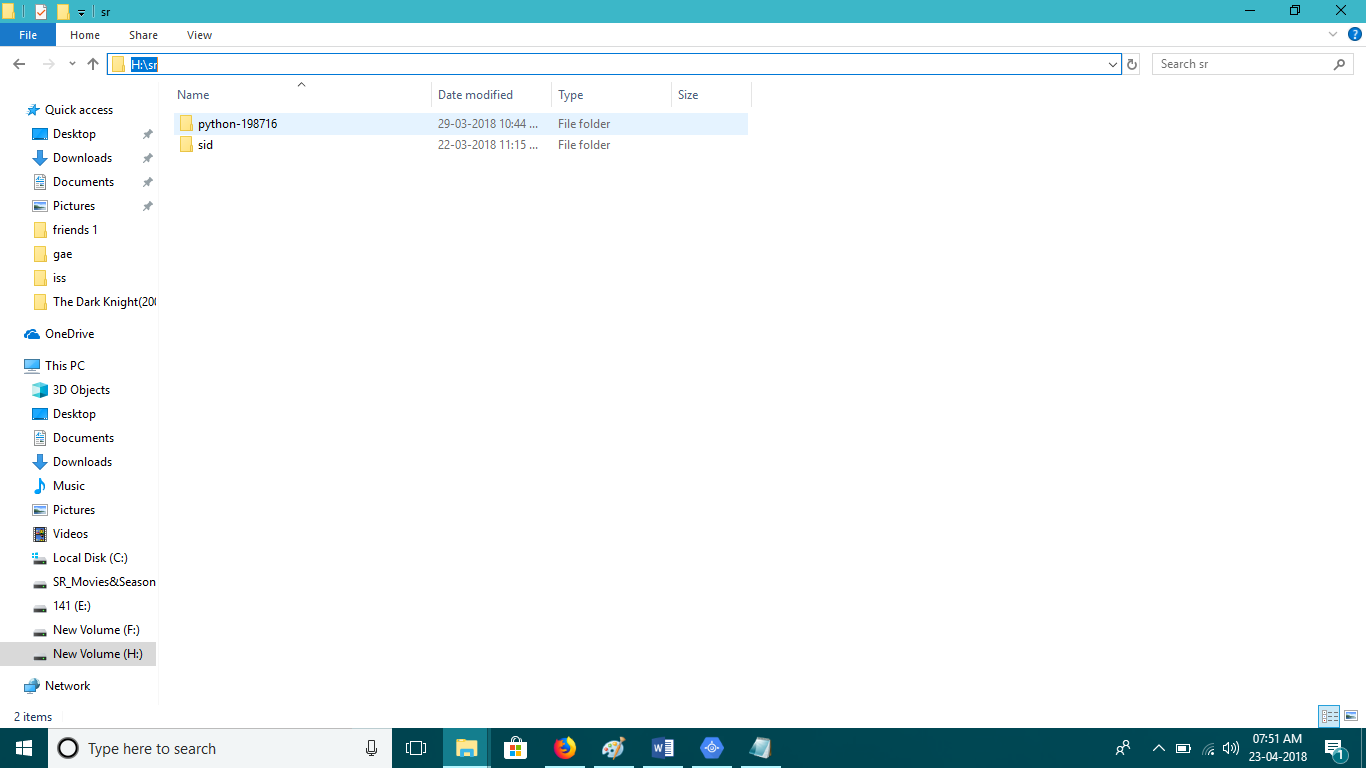
1. **After that go to file🡪Create new application”**



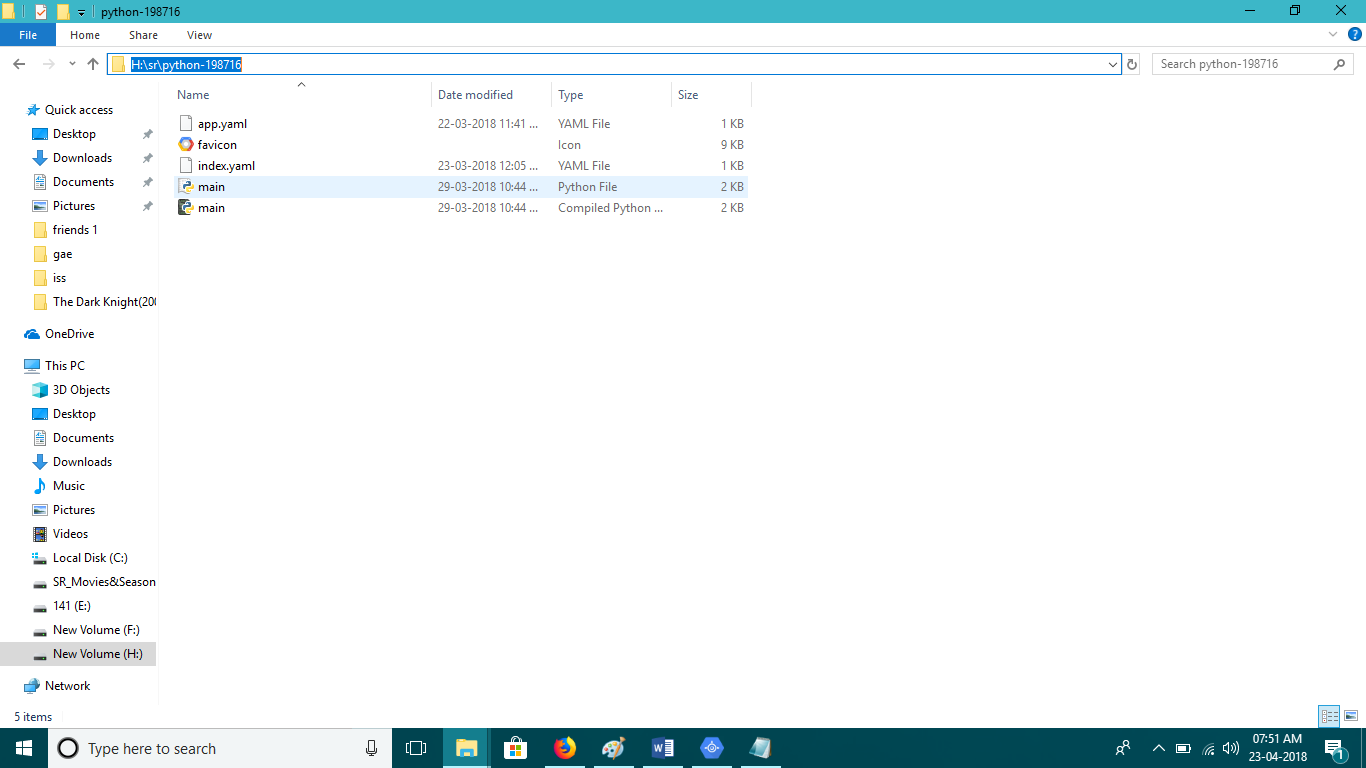
1. **Give application name that name must me your project id and choose give a path to project by clicking on “Browser” then click on “CREATE”. you can choose port number according to your need.**



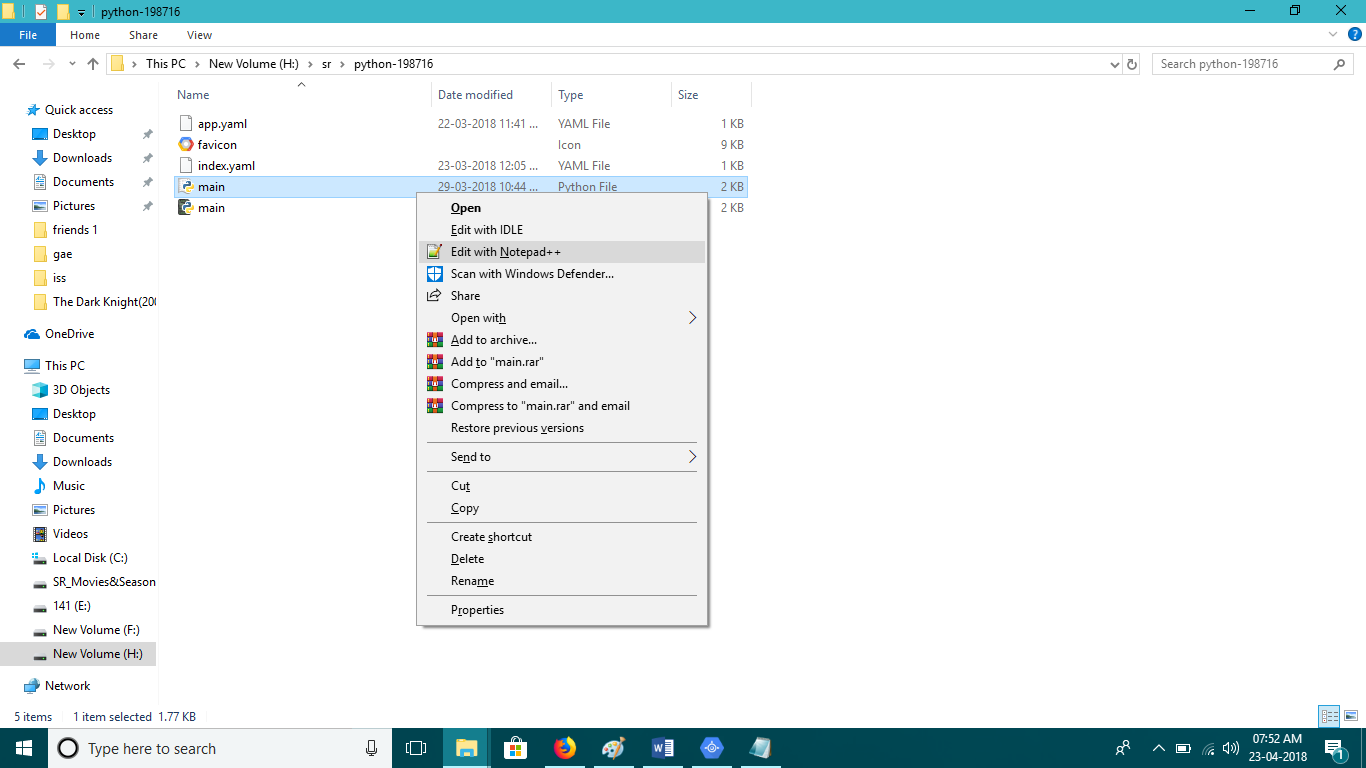
1. **Go to that path you will see the folder name “application name”.**



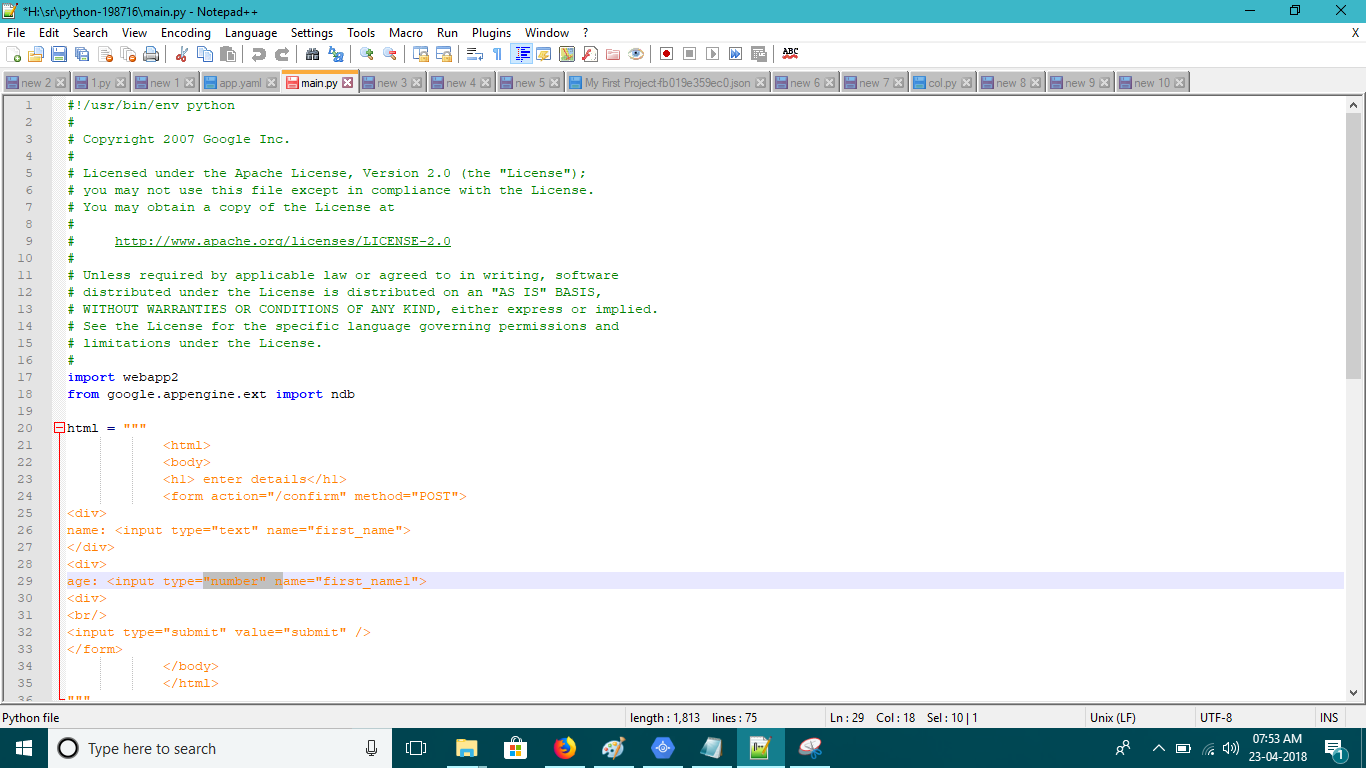
1. **Click on that folder you will get to see some files as bellow.**

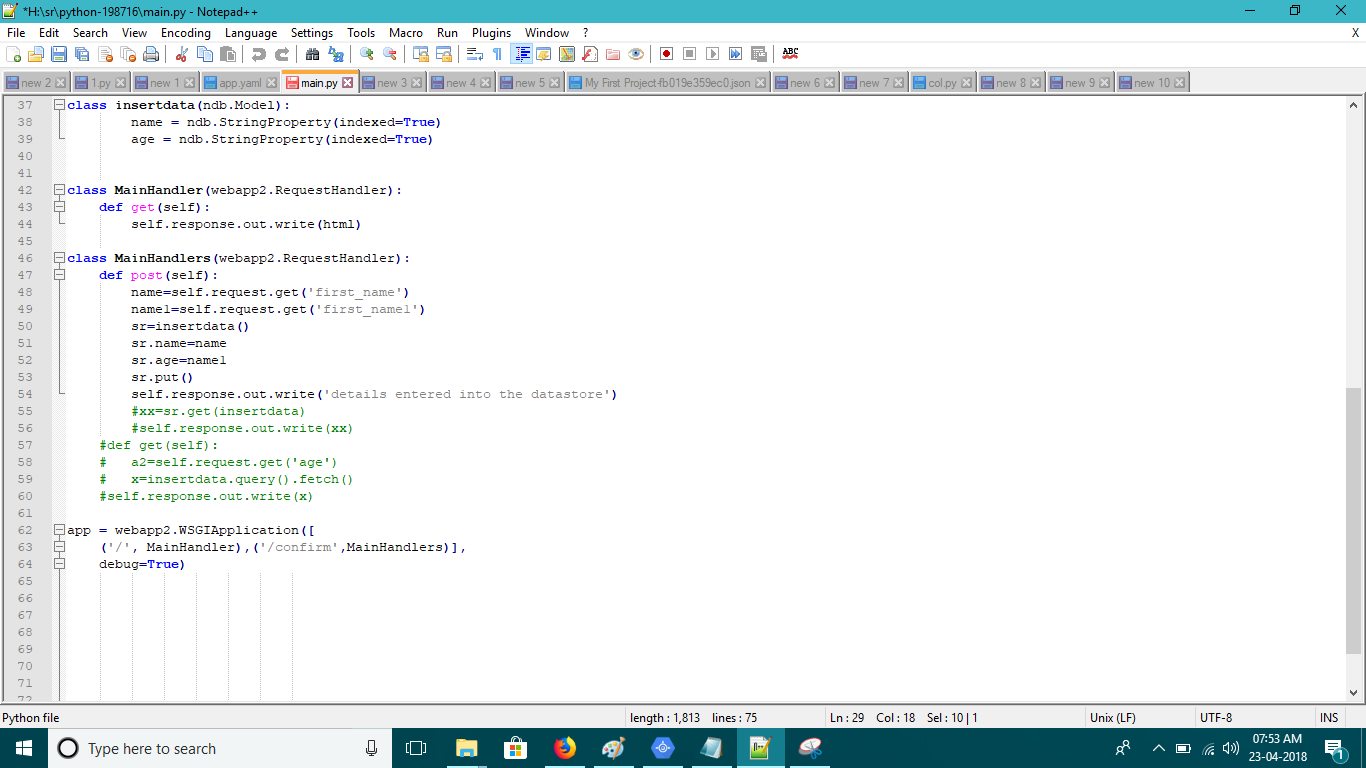


1. **Do left click on “main.py” and open it.**

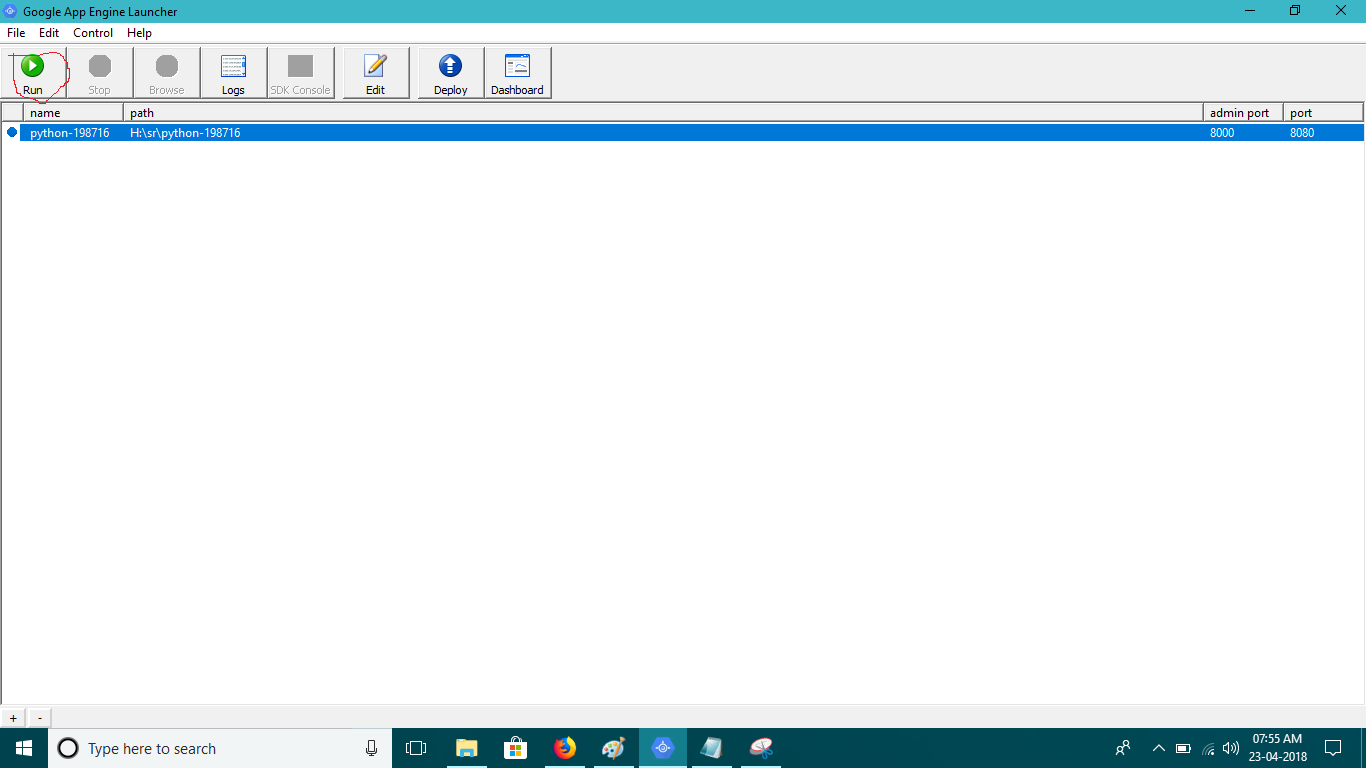


1. **Then you can write your code .**

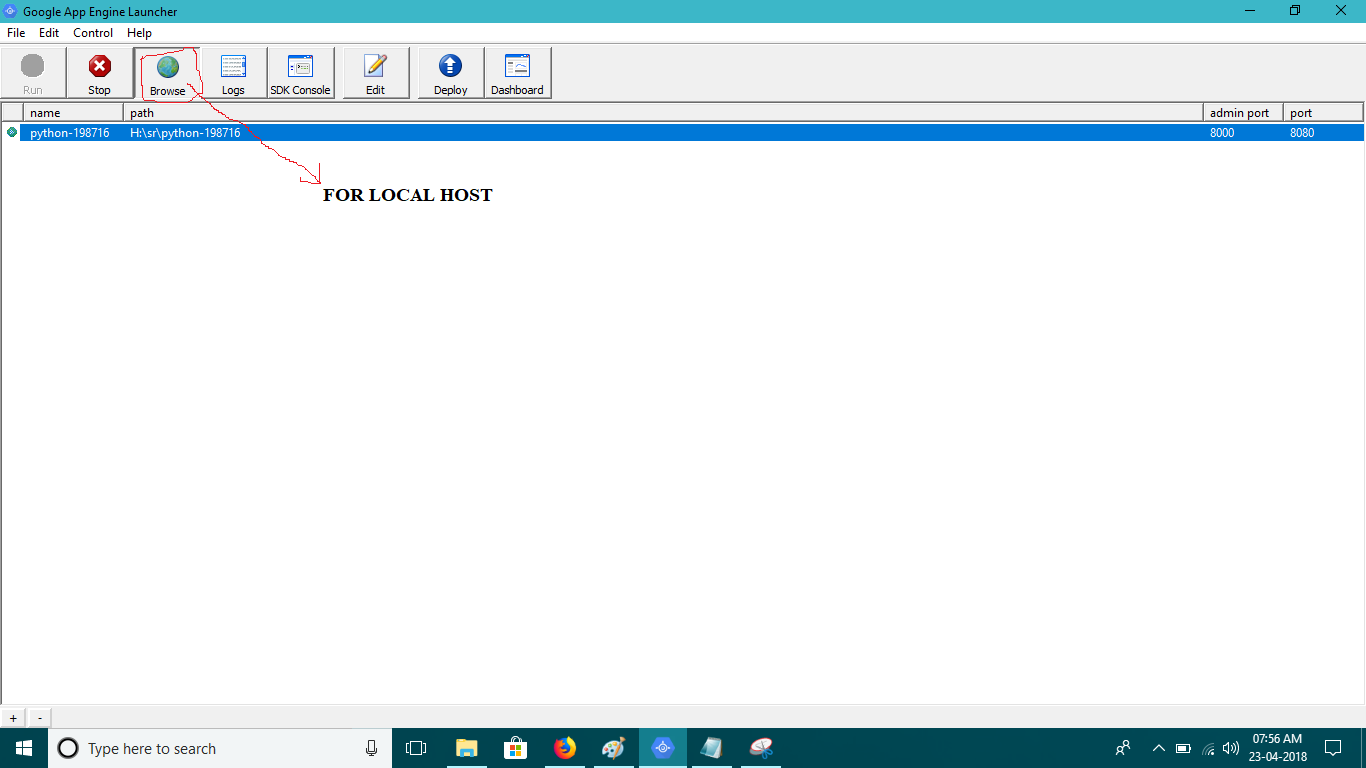




1. **Go to app launcher select the app you want to run and then click on “Run”.**



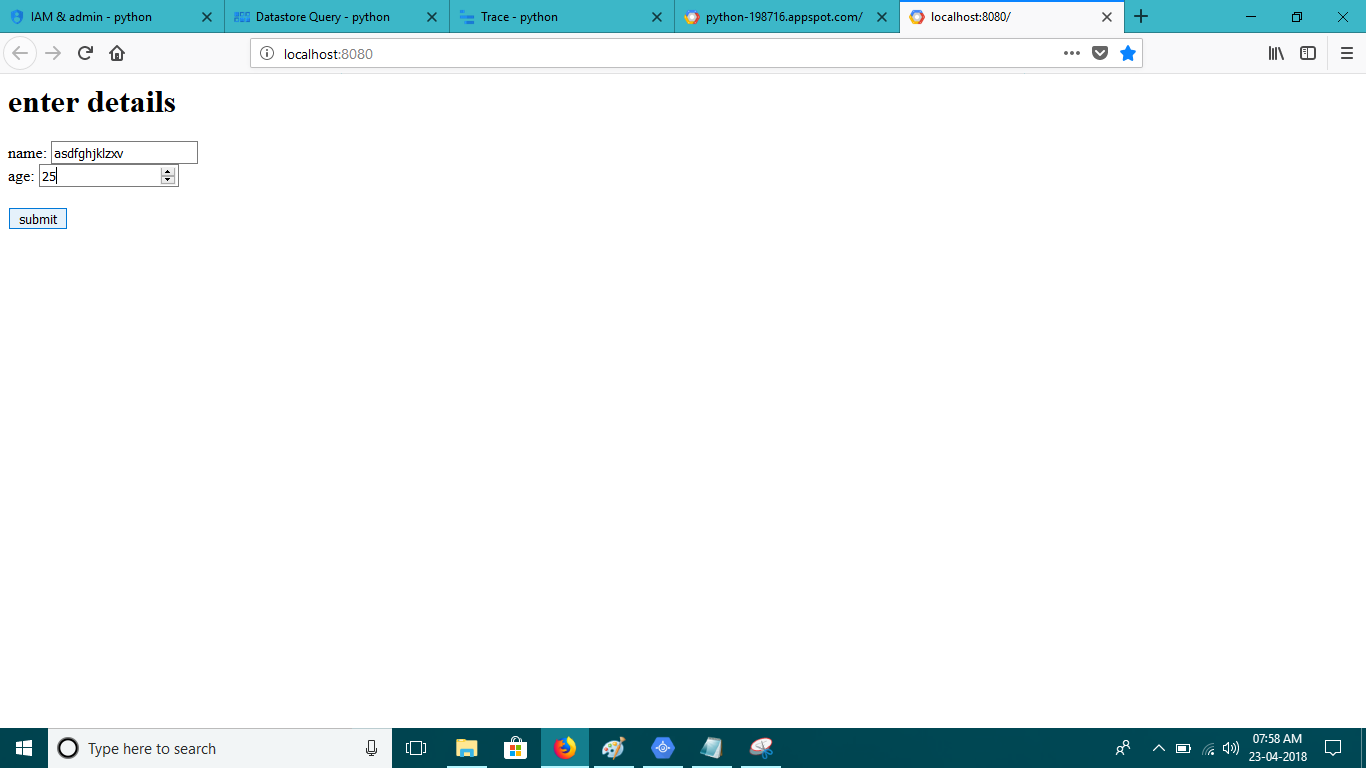
1. **If you want to deploy your app on local host then click on “Browse”**

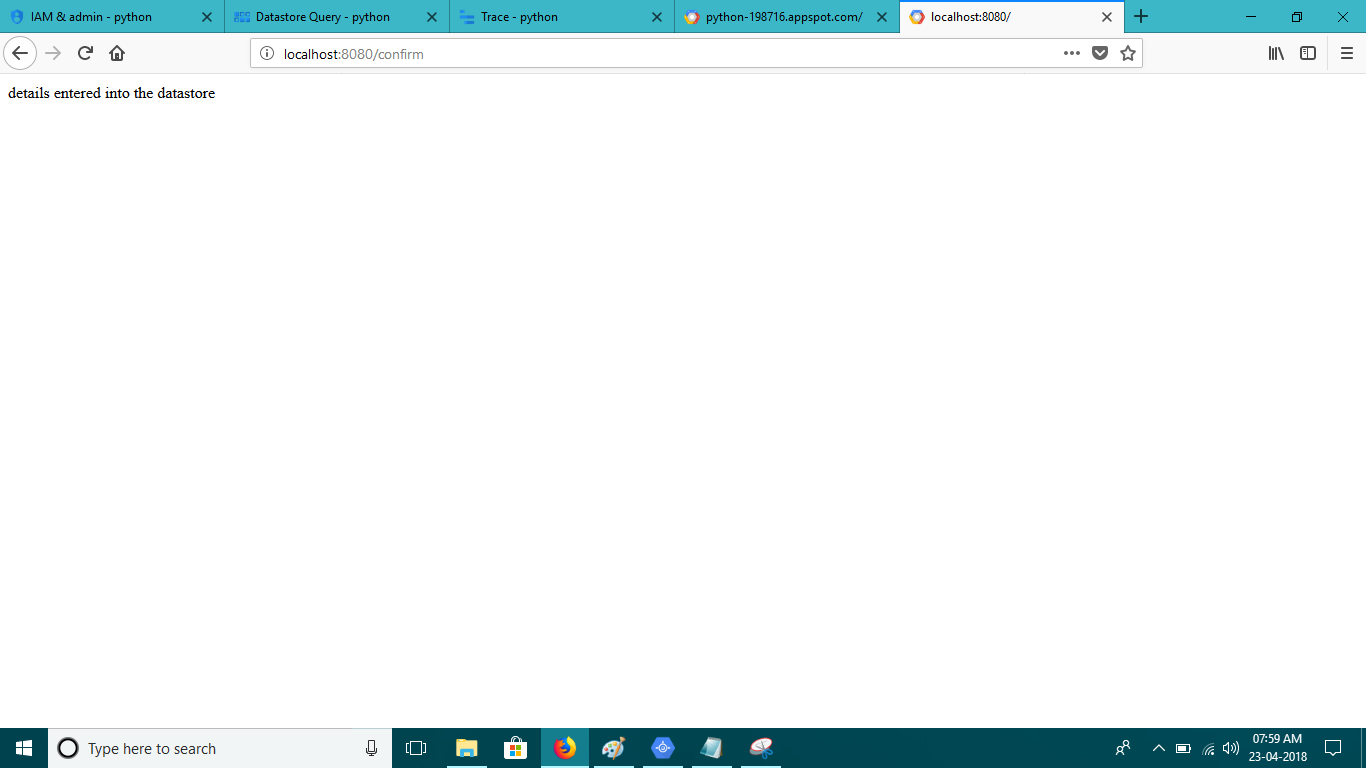


1. **Tab will be open in your browser.**

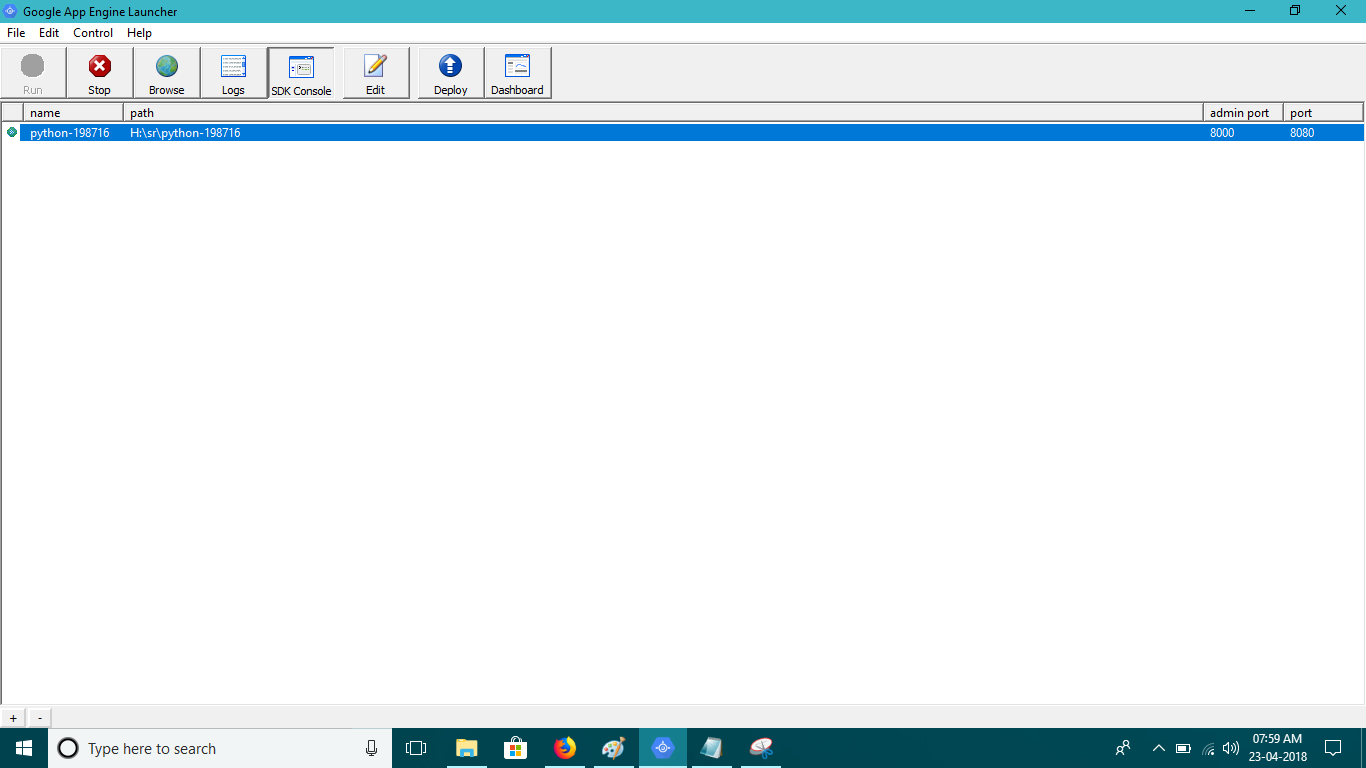


1. **Fill details and click on “submit”.**

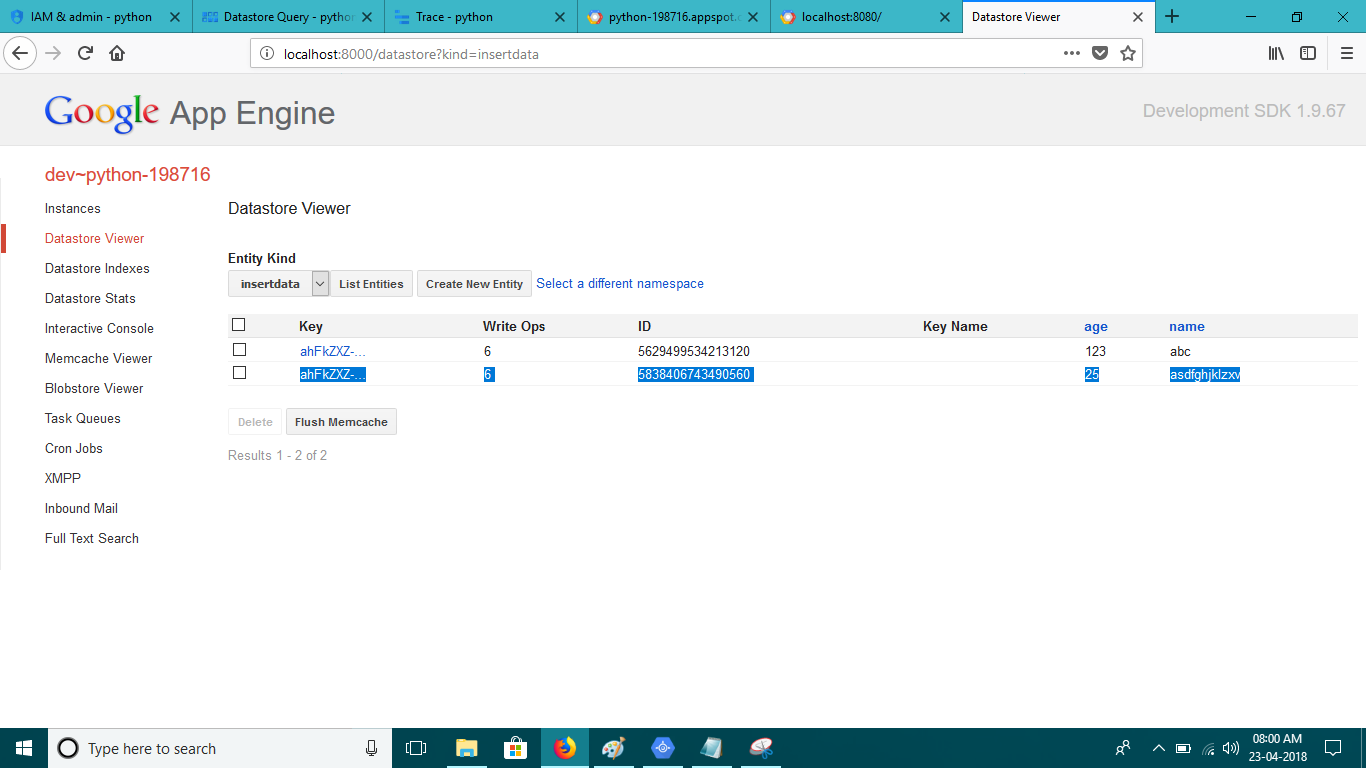




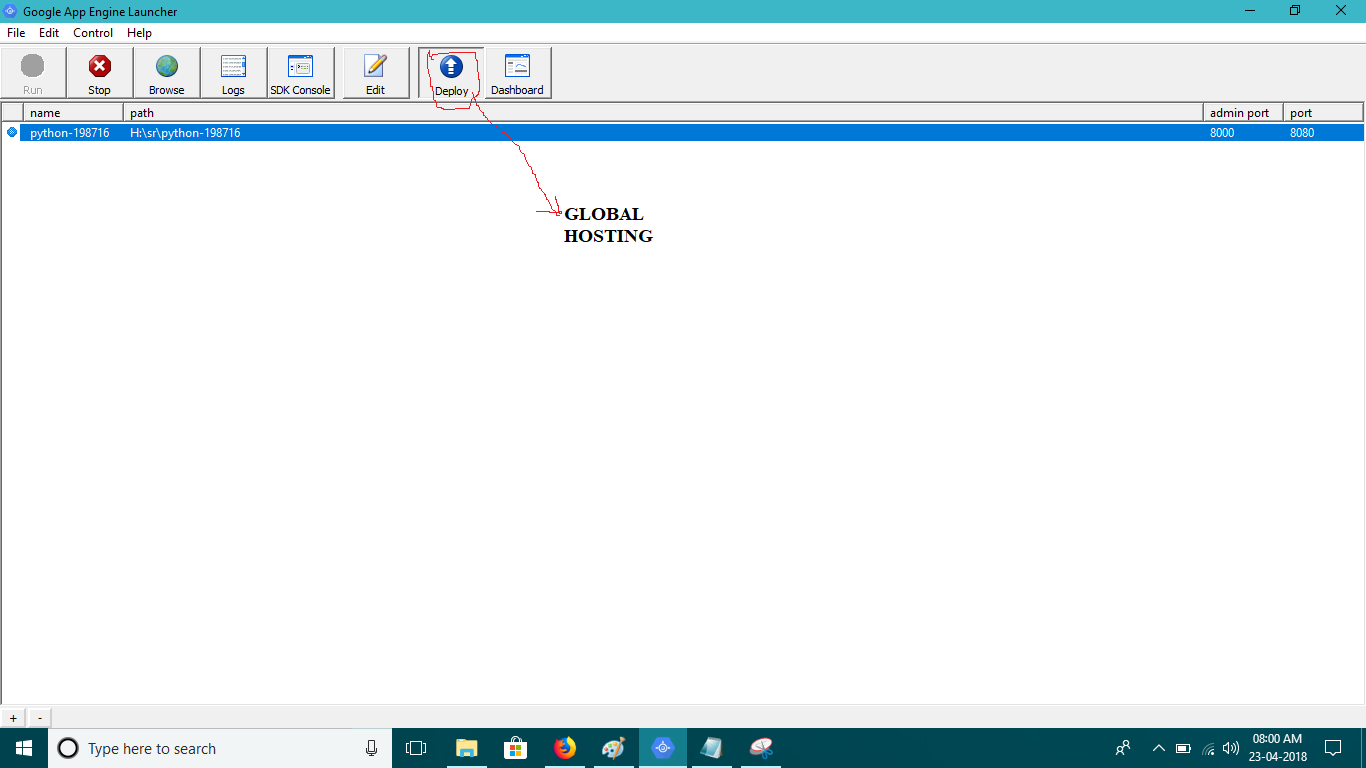
1. **If you wish to see the database of local host application then click on “SDK Console”.**



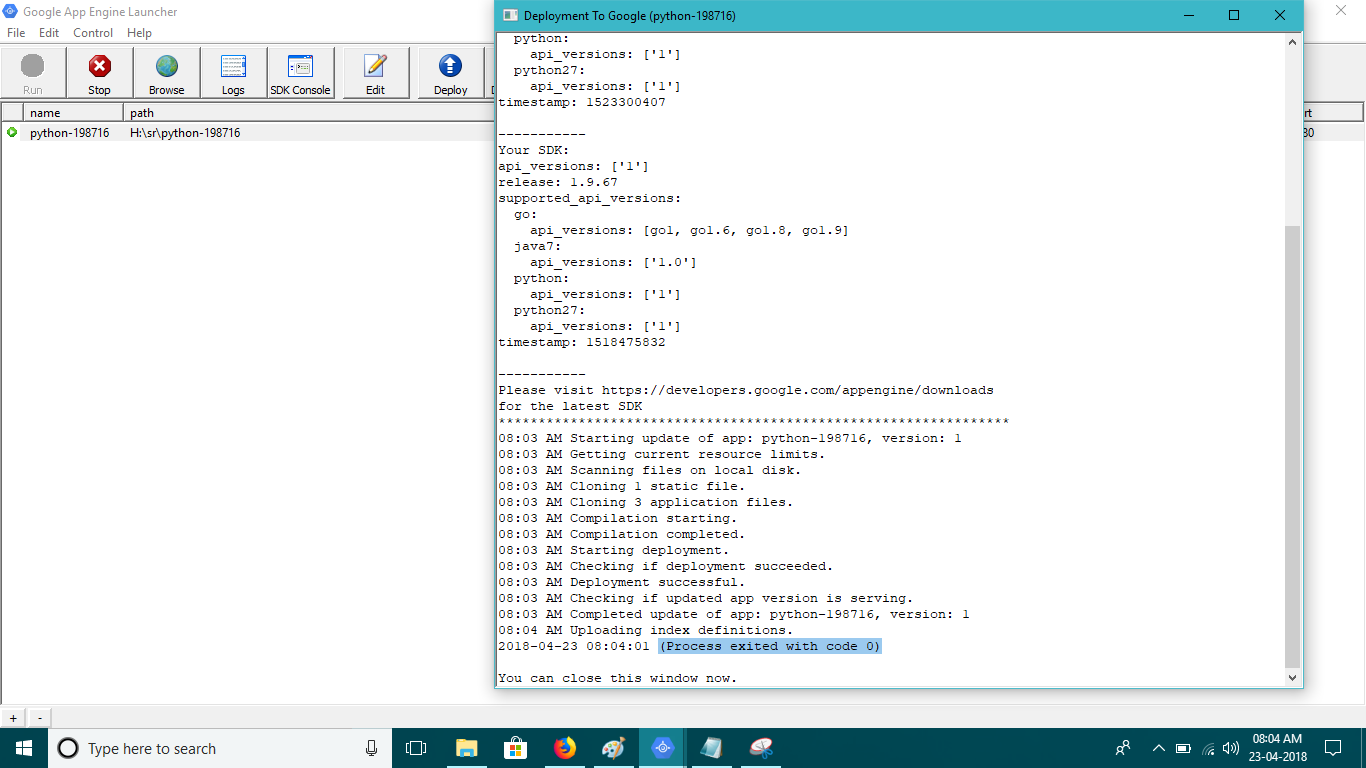
1. **New tab will be open in your browse go to DataStore Viewer” select your kind name(in normal word “your table name”).**



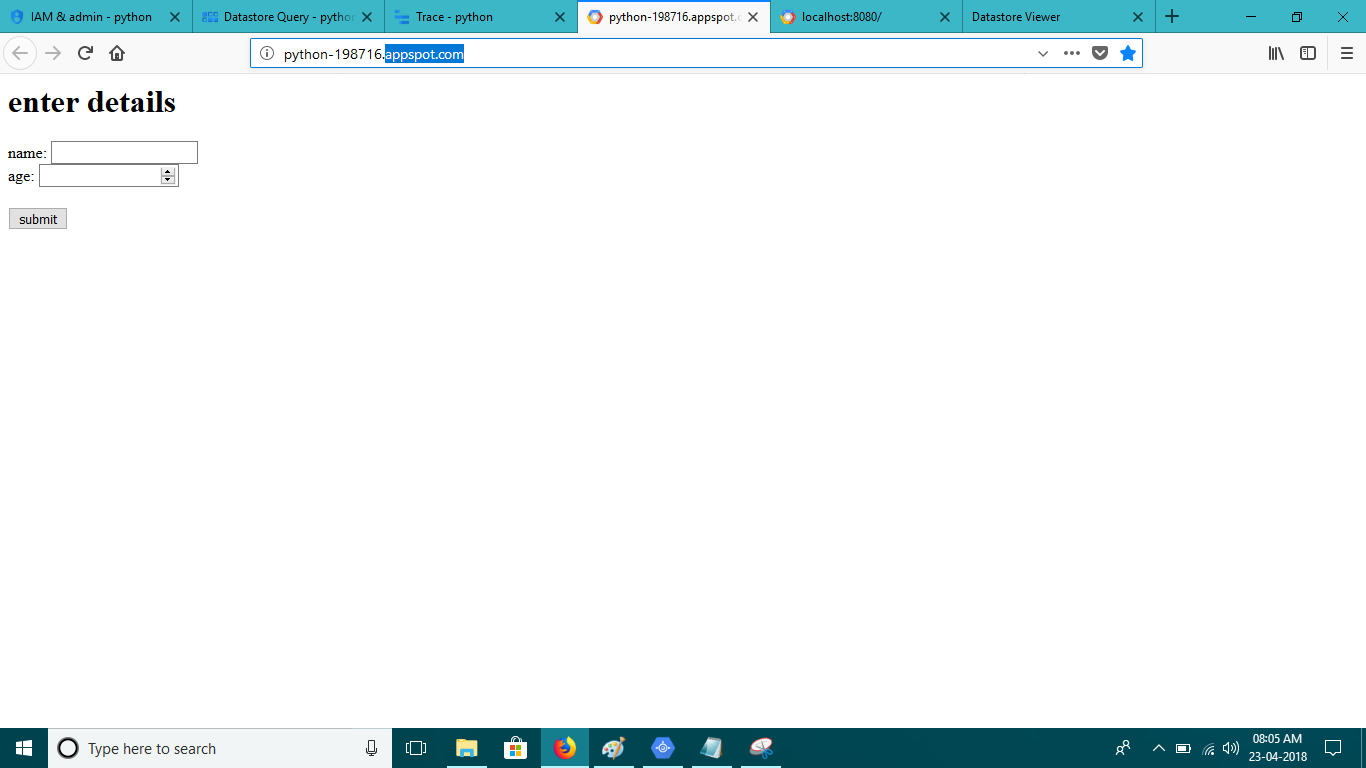
1. **For deploy app on global level then click on “Deploy”.**



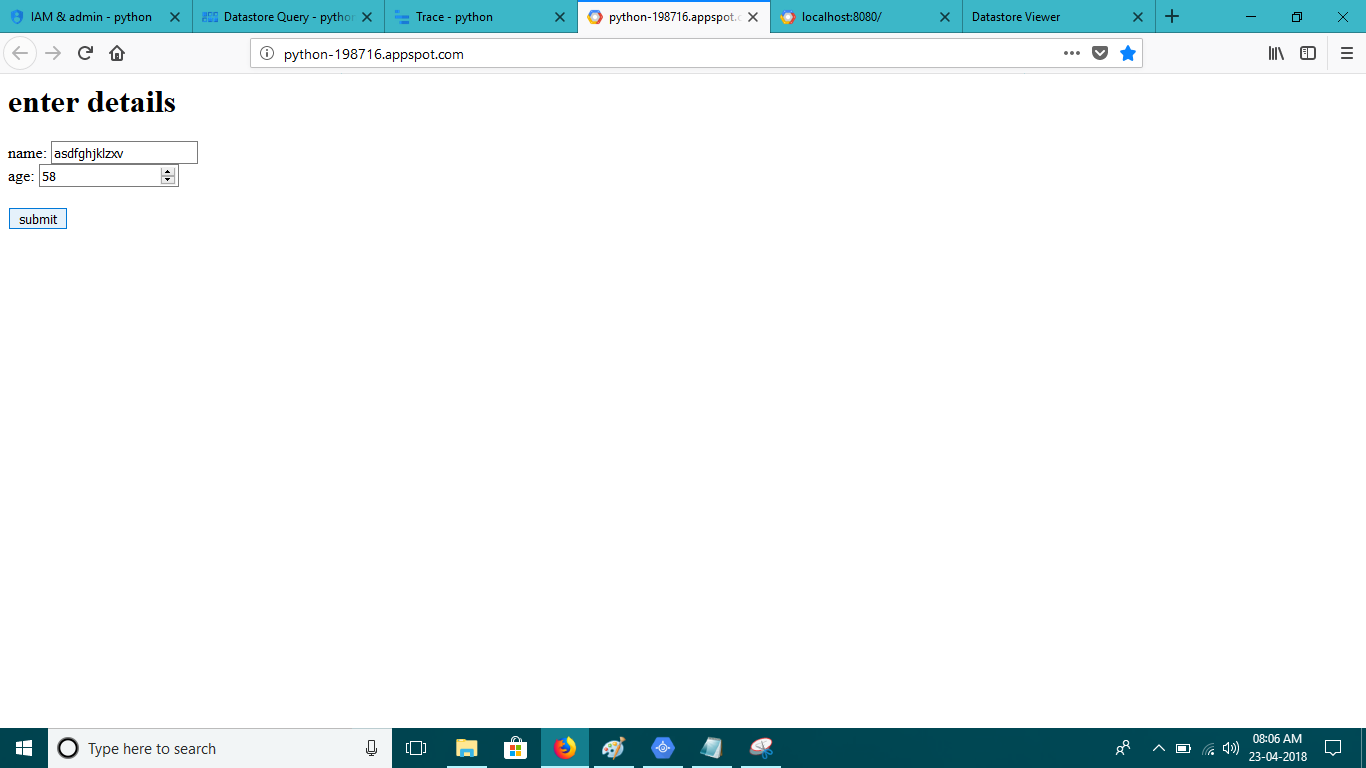
1. **New window will open and start processing on your action, it will take some time and high internet connection. If “process existed with code 0” show up then your application is successfully deploy otherwise there must be some error.**

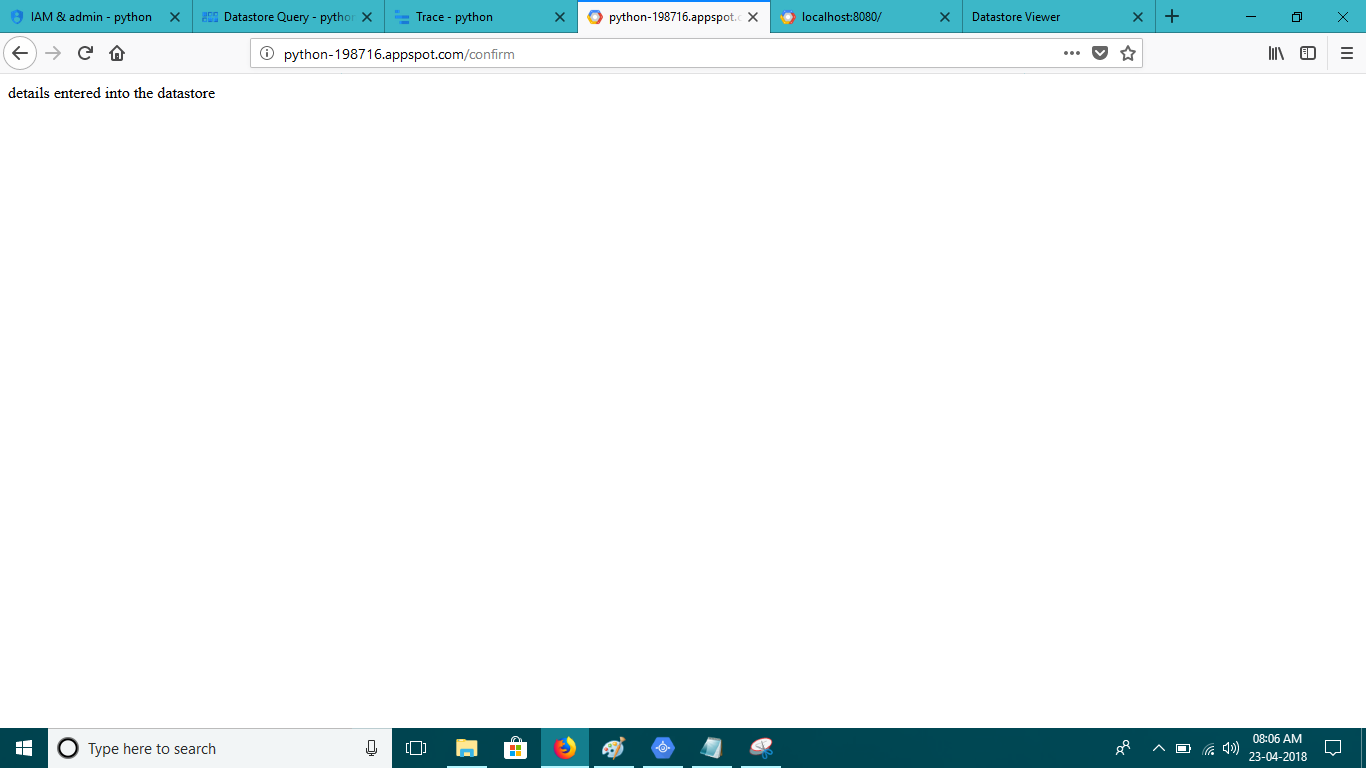


1. **Search “your project id”.appspot.com. here our project id is “python-198716” and .appspot.com is your domain and its default.**

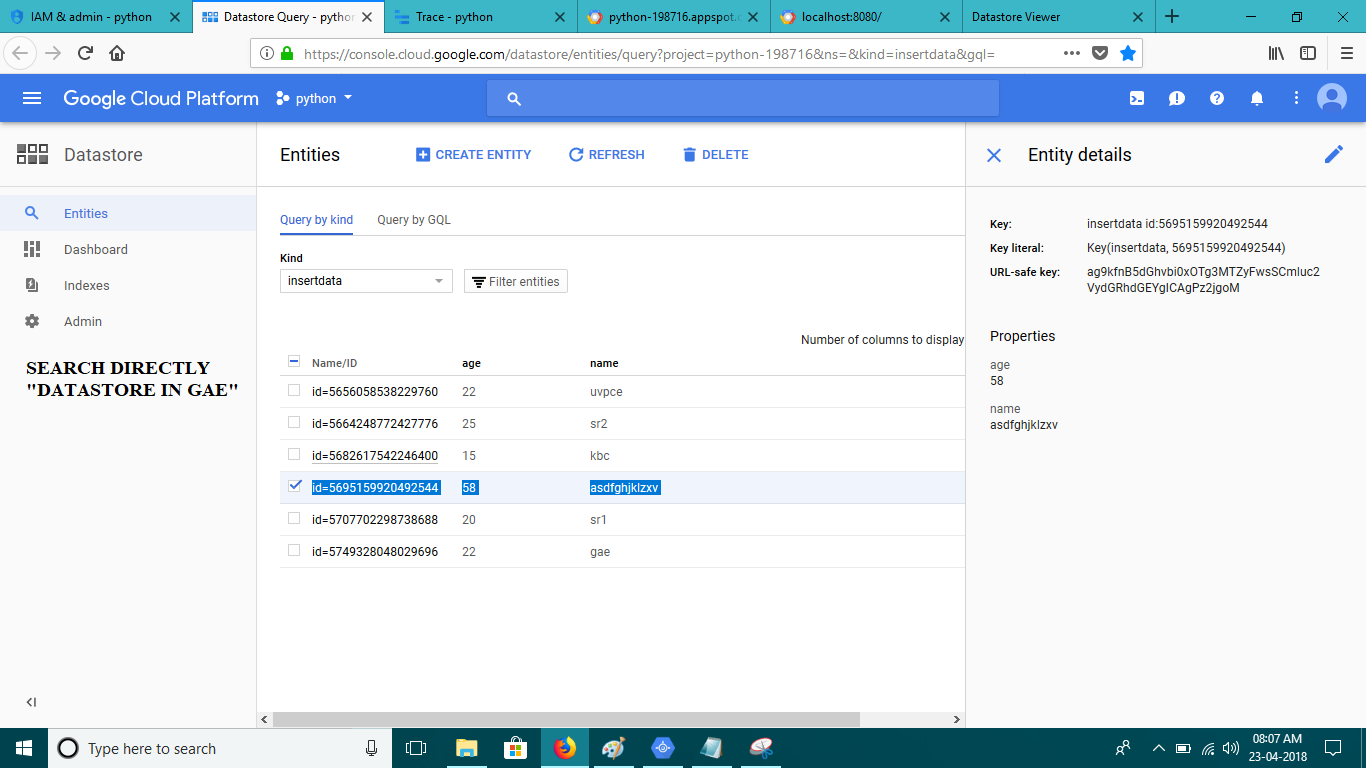


1. **Fill the details and click on “submit”.**

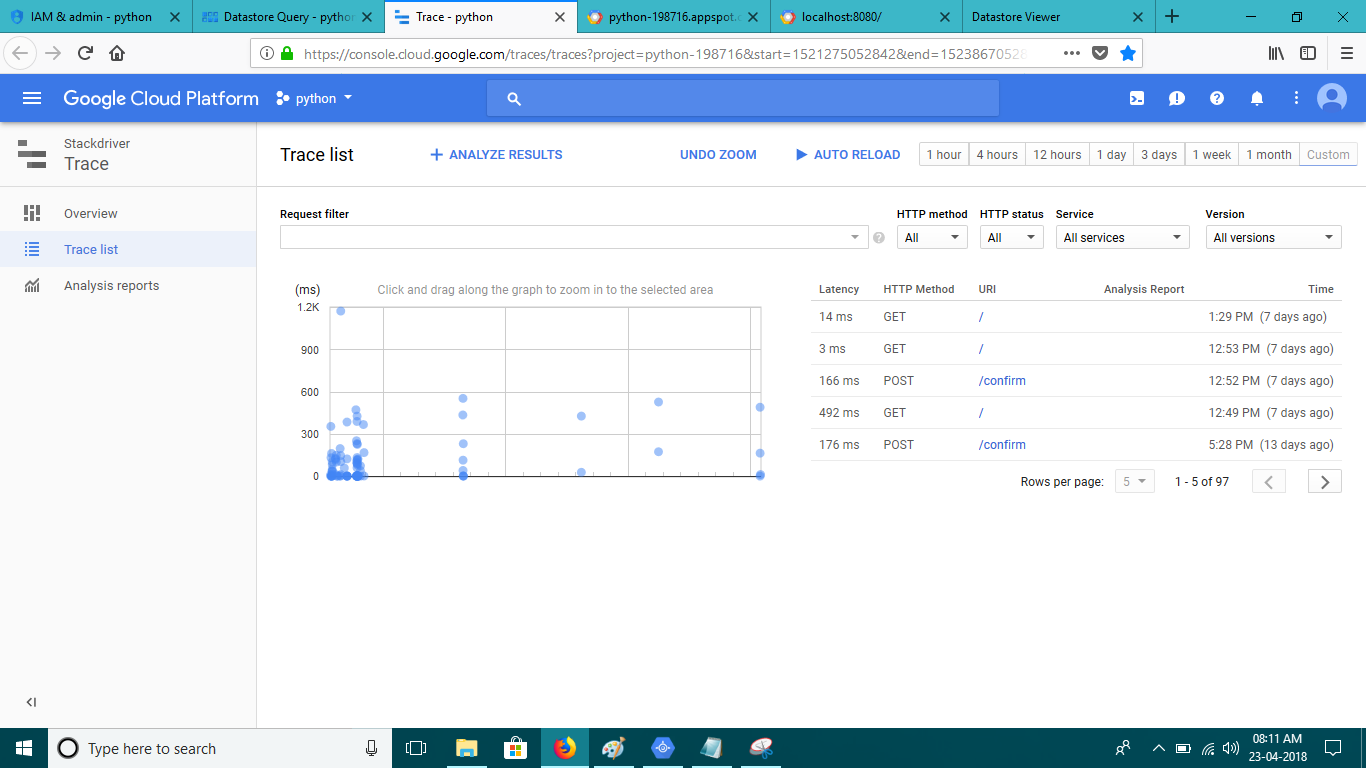




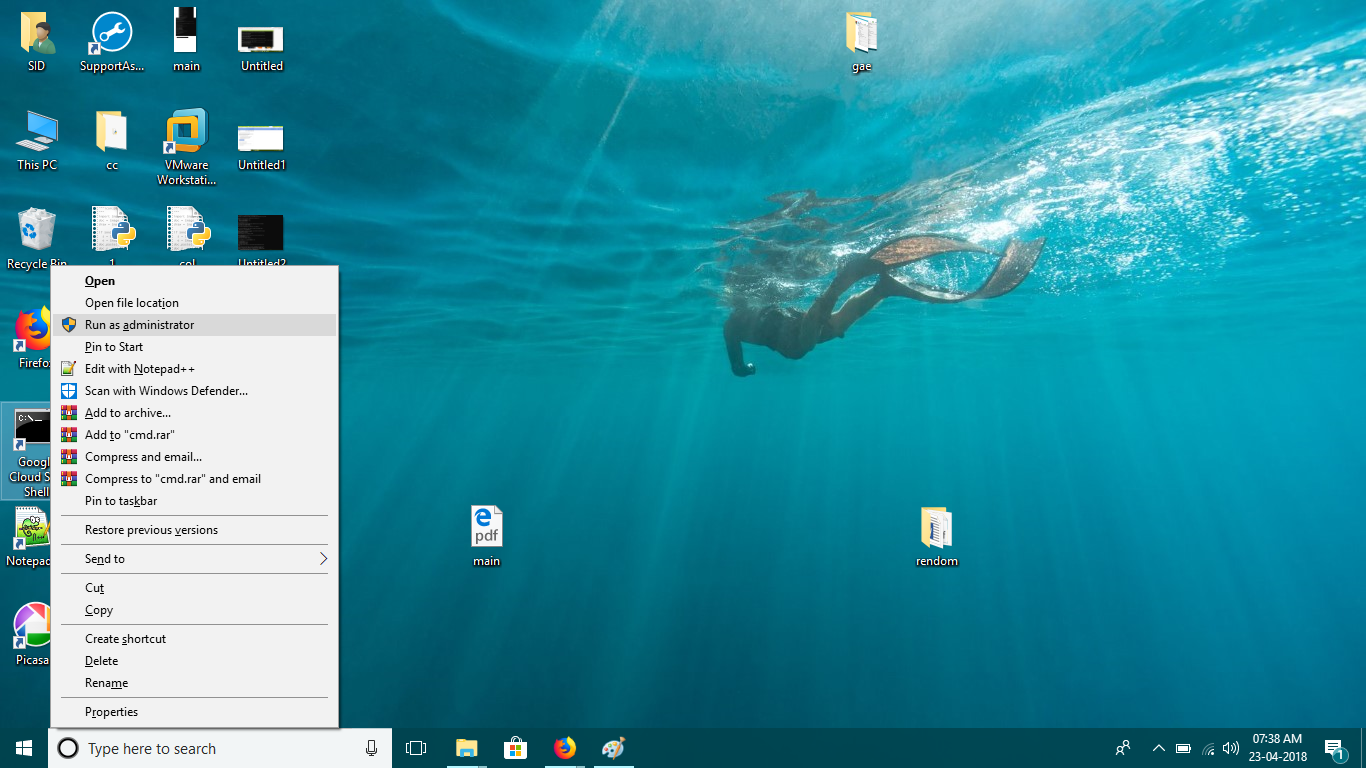
1. **Now go to DataStore select your kind name you will get to see the details you entered. In case if you don’t find Datastore search if on google in same window and different tab.it will auto config.**



1. **Go to trace you will see the full details about your app. It will be monitor by GAE.**



1. **Open SDK.**

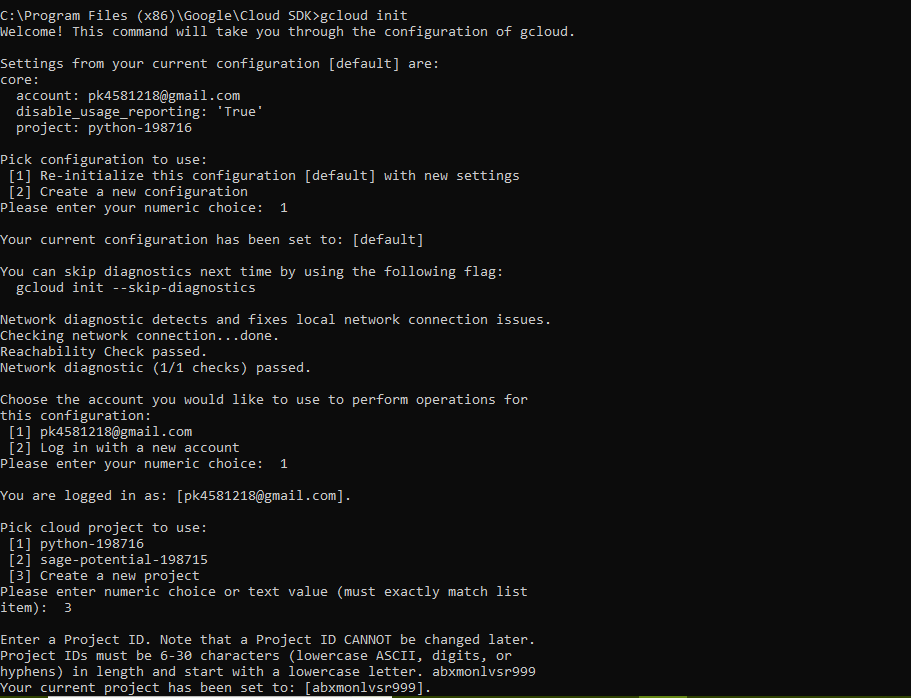


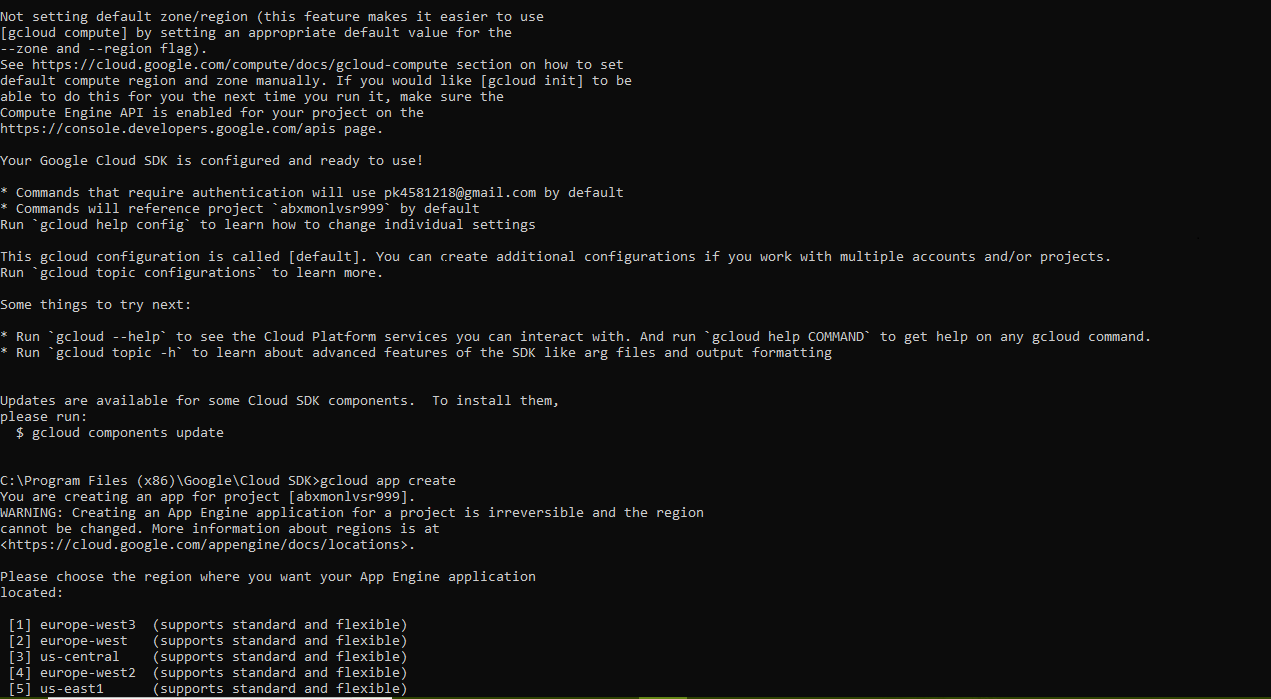
1. **New window will open.**



1. **Start Wring command as bellow to create new project and app and choose account to work in.**

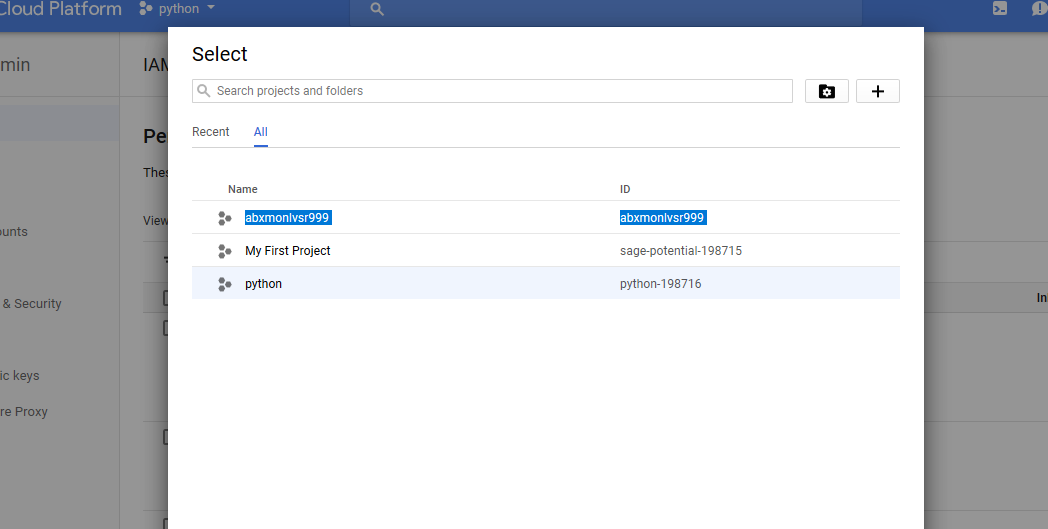
**Go forward according to your and your app’s need.**







1. **Go to your GAE account. Select all project and you will get to see a new project that is just created by you using Google Cloud SDK. If you want to start work in that project then select it and then start making app and follow step number 11 to 28.**



* **References**

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

<https://cloud.google.com/solutions/architecture/webapp>

<https://www.techopedia.com/definition/31267/google-app-engine-gae>