**Title :**To demonstrate the use of PaaS(Platform as a Service) tools.

**Aim :**To demonstrate the use of following PaaS tools:

1) Cloud Foundry

2) GoogleApp Engine

3) OpenShift

**Objective:** To study and implement use of different PaaS tools.

To implement and test programs on real cloud environment.

**Theory :**

* **Google App Engine:**

**Google App Engine** (often referred to as **GAE** or simply **App Engine**) is a platform as a service (PaaS) cloud computing platform for developing and hosting web applications in Google-managed data centers. Applications are sandboxed and run across multiple servers. App Engine offers automatic scaling for web applications—as the number of requests increases for an application, App Engine automatically allocates more resources for the web application to handle the additional demand

# The App Engine Standard Environment

# The App Engine Standard environment makes it easy to build and deploy an application that runs reliably even under heavy load and with large amounts of data. It includes the following features:

1. Persistent storage with queries, sorting, and transactions.
2. Automatic scaling and load balancing.
3. Asynchronous task queues for performing work outside the scope of a request.
4. Scheduled tasks for triggering events at specified times or regular intervals.
5. Integration with other Google cloud services and APIs.

## The Standard Environment Development Kit

## Software Development Kits (SDKs) for App Engine are available in all supported languages. Each SDK includes:

* All of the APIs and libraries available to App Engine.
* A simulated, secure sandbox environment, that emulates all of the App Engine services on your local computer.
* Deployment tools that allow you to upload your application to the cloud and manage different versions of your application.
* **Google App Engine Simple Guide**

This guide help you to developed the cloud environment, Installation steps and deployment of your application on Google app engine.

**Step 1:** Set Up Development Environment

1. **Editor** - we recommend Sublime Text (It’s optional )
2. **Python 2.7** - check your version by running python -V and if it's not 2.7.\*, or download download Python 2.7.9 for your OS and install it.
3. **SDK** - download and install Google App Engine SDK for Python

**Step 2:** Set app Google app engine simple hello word app on LocalHost.

1. Open Google app engine Launcher.

2. File -> create new application ->choose application name as you like -> choose parent directory to save the project. ->select port as 8080 and admin port default. ->create.

3. Now you can see your application on Google App Engine Launcher. Select your project -> Click on run button

4. Open any browser -> type URL (**localhost :8080**)

**Step 3:** Set app Google app engine simple hello word app on (Google app engine appstore)

Creating project at Google app engine console using Gmail login.

1. Go here: https://console.developers.google.com
2. Select **Create project**
3. Click **Edit** to view the *Project ID* field
4. Choose a *unique* project id(or choose default)
5. Give your project a name (doesn't have to be unique)
6. Click **Create**

**Step 4:** Change in the **.ymal**file.Open the project folder, open .ymal file in text editor and change the *Project ID*  as you generate in step 3

**Step 5:** Now run the application again as on browser: wwww.application-name.appspot.com

**Input :** UserName and Password

**Output :** Login Successfully

**Platform :**  Windows 7 64bit / Windows 8 64bit

**Software :** **Editor** -Recommend Sublime Text

**Python 2.7**

GoogleAppEngine **SDK**

**Conclusion:**

Thus we have successfully completed and studied PaaS tool (Google App Engine) and implemented on real cloud environment.

**Mathematical Model**

Let’s **S= {I,O,F,Sc,Fc}**

Where,

I=Input

O=Output

F=Function set’s

Sc=Success cases

Fc=Failure cases

**Input:** I={I1,I2,I3…….In}

I1=Username

I2=Password

**Output:**O={O1,O2,O3…….On}

O1=successfully login on account

O2=successfully logout on account

**Function set’s**: F= {F1, F2, F3……Fn}

F1=Handling the current user request [RequestHandler()].

F2=Check current user and validate it. [get\_current\_user()]

**Failure cases**: Fc={Fc1,Fc2,Fc3……Fcn}

Fc1=User does not entered correct user name and password.

Fc2= User unable to login on account.

Fc3=User unable to login on account

**Success cases**: Sc= {Sc1,Sc2,Sc3….Scn}

Sc1=User entered correct user name and password.

Sc2=User successfully login on account.

Sc3=User successfully logout on account.

**FAQ’s:**

# Q.1) what are the different App Engine Standard Environment?

# Ans->Standard Environment supported by app engine are Python2.7,Java7,PHP,Go,Node.js

**Q.2) what are the features supported by the GAE?**

**Ans->**

1. Persistent storage with queries, sorting, and transactions.
2. Automatic scaling and load balancing.
3. Asynchronous task queues for performing work outside the scope of a request.
4. Scheduled tasks for triggering events at specified times or regular intervals.
5. Integration with other Google cloud services and APIs.

**Q.3)GAE are is opensorce or free?**

**Ans->**Google App Engine is free up to a certain level of consumed resources. Fees are charged for additional storage, bandwidth, or instance hours required by the application

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Code (main.py) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

import webapp2

fromgoogle.appengine.api import users

classMainHandler(webapp2.RequestHandler):

def get(self):

self.response.write('Hello world!')

classMyHandler(webapp2.RequestHandler):

def get(self):

user = users.get\_current\_user()

if user:

greeting = ('Welcome, %s! (<a href="%s">sign out</a>)' %

(user.nickname(), users.create\_logout\_url('/')))

else:

greeting = ('<a href="%s">Sign in or register</a>.' %

users.create\_login\_url('/'))

self.response.out.write('<html><body>%s</body></html>' % greeting)

app = webapp2.WSGIApplication([

('/', MyHandler),

('/', MainHandler),

], debug=True)

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Code (app.yaml) \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

application: deductive-star-125712

version: 1

runtime: python27

api\_version: 1

threadsafe: yes

handlers:

- url: /favicon\.ico

static\_files: favicon.ico

upload: favicon\.ico

- url: .\*

script: main.app

libraries:

- name: webapp2

version: "2.5.2"