Assignment No. Elec 3- C1

Aim

Use of PaaS Tools(OpenShift)

Problem Definition

Lab teacher to frame suitable assignment to demonstrate the use of following PaaS tools: Cloud Foundry(Hint: Use Spring Framework), GoogleApp Engine, OpenShift

Learning Objectives

- To understand platform as a service of cloud computing
- Learn how to use OpenShift

Learning Outcome

- Learnt about working of PaaS
- Implemented a basic application to access data from OpenShift

Software And Hardware Requirements

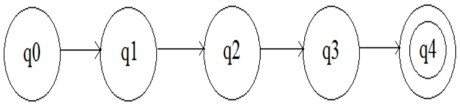
- Latest 64-BIT Version of Linux Operating System
- Eclipse with Python
- Openshift Server

Mathematical Model

Let S be the system of solution set for given problem statement such that,
S = { s, e, X, Y, F, DD, NDD, Su, Fu }
where,
s = start state
such that, s = { Python, MongoDB, RockMongo }
i.e. Application configuration on Cloud server
e = end state

such that, y = { S }
where, S = Service (Database access) received by the user
X = set of input
such that X = { x1 }
where,
x1 = URL of server
such that, x1 = { url | $url\left[a\text{-}z0\text{-}9\right]*$}
Y = set of output
such that Y = { (x_1, y_1), ...(x_i, y_i)}
i.e. Database access(JSON OBJECT)
where, (x_i,y_i) is a key value pair
F = set of function
such that F = { f1, f2, f3 }
where,
f1 = function to connect to OpenShift
f2 = function to retrieve data from OpenShift
f3 = function to print data
DD = Deterministic data
    DD = url
NDD = Nondeterministic data
    NDD = Data at the cloud / OpenShift
    Successfully connected to OpenShift and data of retrieved
Fu = Failure case
    Connection error

State Diagram



where,
q0 = Register on Openshift
q1 = Create Application
q2 = Configure Application
q3 = Request for the service from application
q4 = Access obtained by client

Theory

Platform as a service (PaaS)

It is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app. PaaS can be delivered in two ways: as a public cloud service from a provider, where the consumer controls software deployment and configuration settings, and the provider provides the networks, servers, storage and other services to host the consumers application; or as software installed in private data centers or public infrastructure as a service and managed by internal IT departments.

Below are some of the features that can be included with a PaaS offering:

- Operating system
- Server-side scripting environment
- Database management system
- Server Software
- Support
- Storage
- Network access
- Tools for design and development
- Hosting

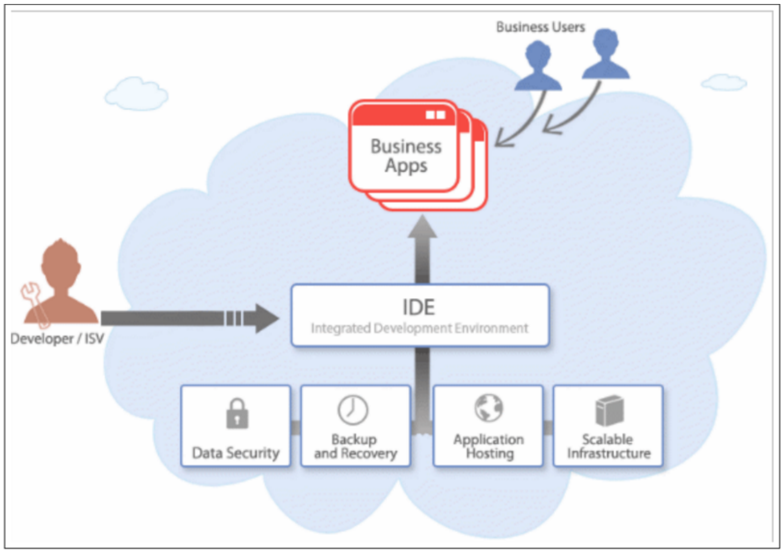


Fig: PaaS Architecture

OpenShift

OpenShift is Red Hats Platform-as-a-Service (PaaS) that allows developers to quickly develop, host, and scale applications in a cloud environment. With OpenShift you have a choice of offerings, including online, on-premise, and open source project options.Online is Red Hats public cloud application development and hosting platform. Online is currently powered by version 2 of the Origin project source code, which is also available under the Apache License.Online supports a variety of languages, frameworks, and databases out of the box. In addition to the built-in languages and services, developers can add other language, database, or middleware components that they need via the OpenShift Cartridge API.

Fig: Openshift Architecture

Program Code

```
//myapp.py

import os
from flask import Flask
from flask import request
import pymongo
import json
from bson import json_util
from bson import objectid
import re
app = Flask(__name__)

@app.route('/')
def employees():
        #setup the connection
        conn = pymongo.Connection
        (os.environ['OPENSHIFT_MONGODB_DB_URL'])
        db = conn.myapp

        #query the DB for all the employee
        result = db.employee.find()

    #Now turn the results into valid JSON
        return str(json.dumps(
        {'results': list(result)},default=json_util.default))

if __name__ == '__main__':
    app.run()
```

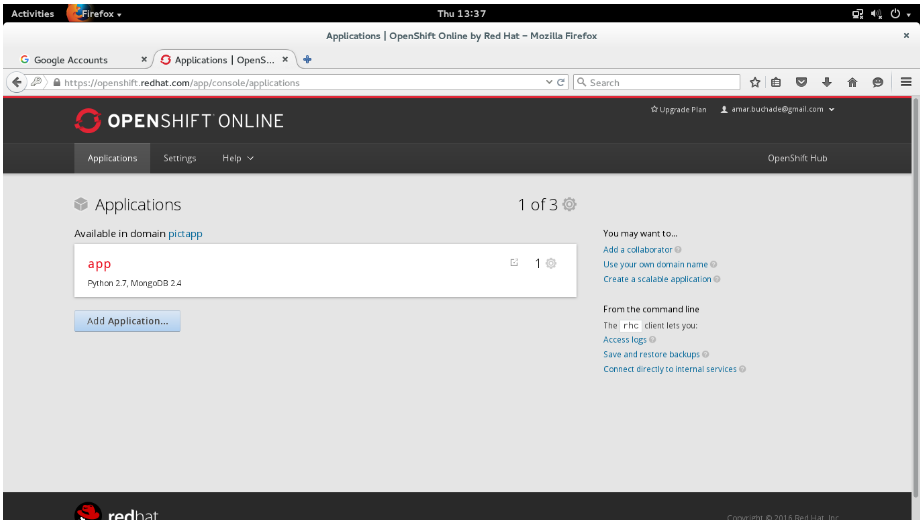
```
//setup.py

from setuptools import setup

setup(name='YourAppName',
      version='1.0',
      description='OpenShift App',
      author='Your Name',
      author_email='example@example.com',
      url='http://www.python.org/sigs/distutils-sig/',
#      install_requires=['Django>=1.3'],
     )

//wsgi.py

#!/usr/bin/env python
import os

from myapp import app as application


#
# Below for testing only
#
if __name__ == '__main__':
    from wsgiref.simple_server import make_server
    httpd = make_server('localhost', 8051, application)
    # Wait for a single request, serve it and quit.
    httpd.serve_forever()

//Client_Codde.py

import httplib2
import json

resp, content = httplib2.Http().
request("https://myapp-restfuldemo.rhcloud.com/")
jobj = json.loads(content)
count = 1
for i in jobj['results']:
    print 'employee:'
    print 'name: '+ i['name']
    print 'salary: ' + i['salary']
    print 'address: ' + i['address']
    print 'date of joining: ' + i['Date_Of_Joining']
    print 'position: ' + i['position']
    print count
    count=count+1
```

Output

```
[pict@localhost pythoncodev2]$ python c1client.py
employee :
name: prasad
salary: 2400000
address: pune
date of joining: july 2016
position: developer
1
employee :
name: sid
salary: 2400
address: pune
date of joining: july 2016
position: tester
2
employee :
name: sid
salary: 2400
address: pune
date of joining: july 2016
position: tester
3
employee :
name: sid
salary: 2400
address: pune
date of joining: july 2016
position: tester
4
employee :
name: mandy
salary: 24
address: nashik
date of joining: july 2016
position: management
5
employee :
name: mandy
salary: 24
```

```
address: nashik
date of joining: july 2016
position: management
6
employee :
name: mandy
salary: 24
address: nashik
date of joining: july 2016
position: management
7
employee :
name: mandy
salary: 24
address: nashik
date of joining: july 2016
position: management
8
[pict@localhost pythoncodev2]$
```

Output Screenshots



Available in domain pictapp

Python 2.7, MongoDB 2.4

You may want to...

Add a collaborator

Use your own domain name

Create a scalable application

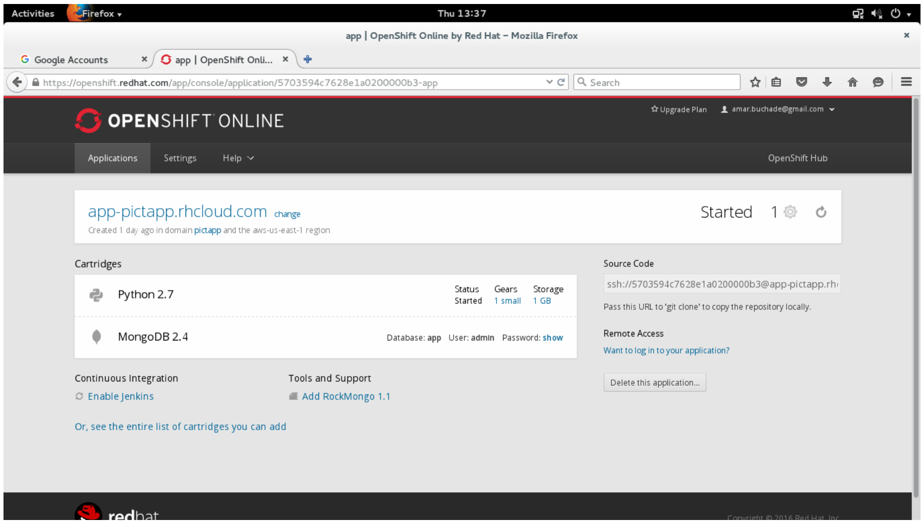
From the command line

The `rhc` client lets you:

Access logs

Save and restore backups

Connect directly to internal services



https://openshift.redhat.com/app/console/application/5703594c7628e1a0200000b3-app

app-pictapp.rhcloud.com

Created 1 day ago in domain pictapp and the aws-us-east-1 region

Started 1

Cartridges

	Status	Gears	Storage
Python 2.7	Started	1 small	1 GB

MongoDB 2.4 Database: app User: admin Password: show

Continuous Integration

Tools and Support

Source Code

ssh://5703594c7628e1a0200000b3@app-pictapp.rh

Pass this URL to 'git clone' to copy the repository locally.

Remote Access

Conclusion

Thus, we have successfully setup a Open Shift cloud PaaS, created an application to connect PaaS and retrieved data from Cloud.