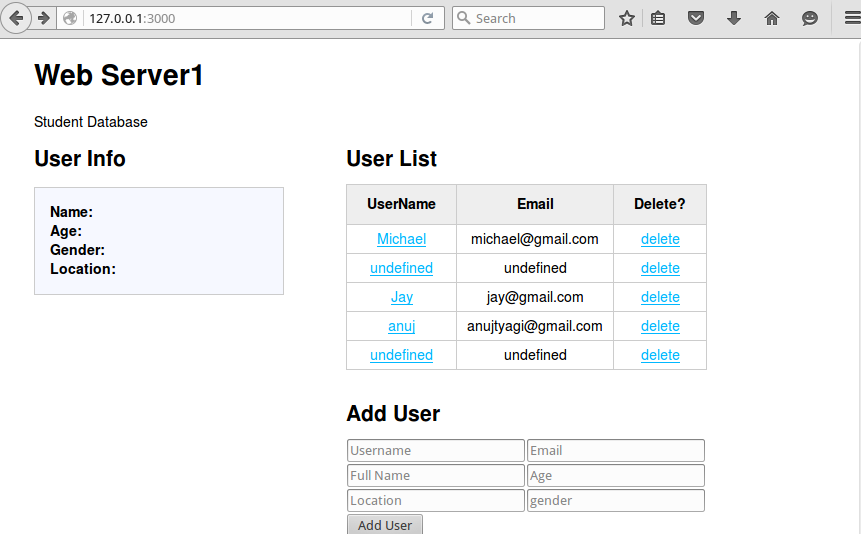
**Assignment**

1. **REST Web API:**

Using Nodejs REST WEB API to communicate on webservers connected to MongoDB server.

Important code to check for the REST operations available in routes/users.js



**GET Operation:** Data of student information is saved in **users/userlist** json format.

router.get('/userlist', function(req, res) {

var db = req.db;

var collection = db.get('userlist');

collection.find({},{},function(e,docs){

res.json(docs);

});

});

**POST Operation:** To add the data, we need to use URL: localhost:3000/users/adduser

router.post('/**adduser**', function(req, res) {

var db = req.db;

var collection = db.get('userlist');

collection.insert(req.body, function(err, result){

res.send(

(err === null) ? { msg: '' } : { msg: err }

);

});

});

**Delete Operation:**

router.delete('/deleteuser/:id', function(req, res) {

var db = req.db;

var collection = db.get('userlist');

var userToDelete = req.params.id;

collection.remove({ '\_id' : userToDelete }, function(err) {

res.send((err === null) ? {msg: '' } : { msg:'error: ' + err });

});

});

module.exports = router;

1. **Interact With API (Python scripts to perform CRUD Operations**

**GET: (Extracting the data from database)**

import urllib, json

url = <http://10.103.45.50:3000/users/userlist>

response = urllib.urlopen(url)

data = json.loads(response.read())

index = -1

for i in range(len(a)):

if a[i]['email']==email:

index = i

break

if index == -1:

print "Not found"

print "Student Details: ",data

**POST: (Adding/uploading data)**

import requests

import json

payload = {}

#create an empty dictionary

url = "http://10.103.45.50:3000/users/adduser"

#Insert the data in dictionary

payload['username'] = str(raw\_input("Enter username : "))

payload['fullname'] = str(raw\_input("Enter fullname : "))

payload['email'] = str(raw\_input("Enter email : "))

payload['age'] = str(raw\_input("Enter username : "))

payload['location'] = str(raw\_input("Enter username : "))

payload['gender'] = str(raw\_input("Enter username : "))

r = requests.post(url, data=json.dumps(payload))

print(r.text)

**Delete: (Removing data)**

import urllib, json

url = "http://10.103.45.50:3000/users/userlist"

response = urllib.urlopen(url)

data = json.loads(response.read())

# removing the user information by taking it's email-id as input

def db\_search(a):

email = raw\_input("Enter the email id of user:\n")

index = -1

for i in range(len(a)):

if a[i]['email']==email:

index = i

break

if index == -1:

print "Not found"

a.remove(a[index])

print "Old database: ",data

db\_search(data)

print "New database: ",data

1. **Scale Up application with NGINX:** Nginx can be used as web server, proxy server, load balancer.

**Load Balancing: /etc/nginx/site-available**

1. Install NGINX: sudo apt­get install ­y nginx

upstream web\_backend {

    server 10.0.0.15;

    server 10.0.0.16;

}

server {

    listen 80;

    location / {

        proxy\_set\_header X­Forwarded­For $proxy\_add\_x\_forwarded\_for;

        proxy\_pass http://web\_backend;

    }

}Here, we have added two servers to the NGINX and it’s balancing the traffic by using round robin algorithm.

1. **IP Tables:** Here, IP table will drop all common ports which are unused. Also, accepting traffic from NGINX Servers and DB server only. All dropped packets by IP tables will be logged into file.

root@web1:/home/anuj#iptables -N LOGGING

root@web1:/home/anuj# sudo iptables -A INPUT -p tcp --match multiport --dports 23,100,25,20,21 -j DROP

root@web1:/home/anuj#iptables -I INPUT -s 10.102.62.236 -j ACCEPT

root@web1:/home/anuj#iptables -I INPUT -s 10.102.62.94 -j ACCEPT

root@web1:/home/anuj#iptables -I INPUT -s 10.102.61.30 -j ACCEPT

root@web1:/home/anuj# iptables -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT

root@web1:/home/anuj#iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT

root@web1:/home/anuj#iptables -A LOGGING -j DROP

root@web1:/home/anuj# sudo iptables -A INPUT -p ICMP --icmp-type 8 -j DROP

root@web1:/home/anuj#iptables -A INPUT -j DROP

1. **Automation with Ansible:**Ansible is a agentless configuration management tool and works with ssh connection only. To install Nginx, we need to provide ssh communication between devices without query of password. It can perform almost any operation to the remote device support ssh. There are 3 ways to configure device using Ansible: Adhoc Mode – Module, Playbook and Roles. I have used playbook to install NGINX.

**#Ansible playbook to install Nginx on web servers:**

**/etc/ansible/ansible-playbook.yml**

---

- host: all

sudo: True

tasks:

- name: Install Nginx

apt : pkg=nginx state=present

handlers:

- name: start nginx

service: name=nginx state=started

