CSC 496-80 Course Project – Installation and Deployment

Dimitra Deliopoulos, Desislava Atanasova

Github – https://github.com/dimitra216/CSC496-Team-Cloud-Project-DA

Abstract

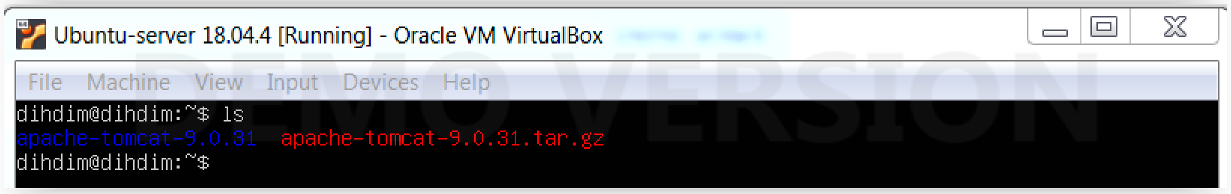
Install and deploy a multi-node computing service on CloudLab's Openstack Cloud. Then select a computing service consisting of at least two service nodes and deploy the corresponding VMs inside CloudLab's Openstack infrastructure.

Other goals include:

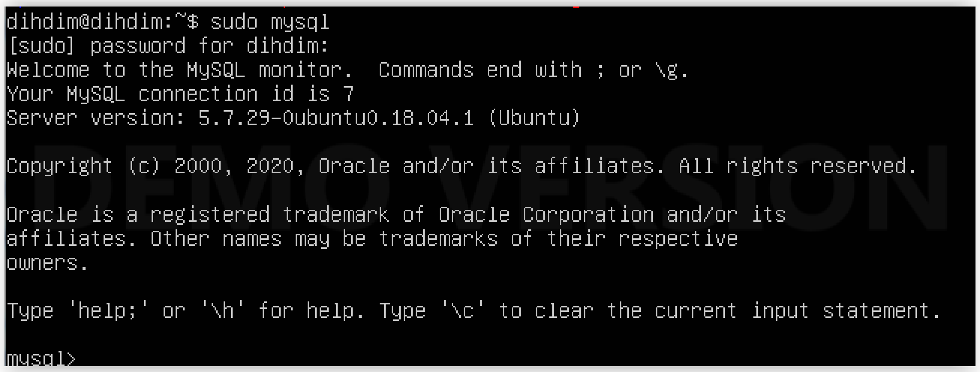
- Able to setup VM nodes that work together to provide a unified computing service to users.
- Able to move these VM nodes into the Cloud, setup proper network environment, and link them together.
- Able to setup at least 2-3 security groups, not using the default security groups on Openstack.

First Approach Project Achieved/Goals

The first webserver we attempted to use was the Tomcat server (version 9.0.31). After installing the webserver and the database server MySQL (server version 5.7.29) as well, it was up and running successfully on Ubuntu 18.04.4 VM.

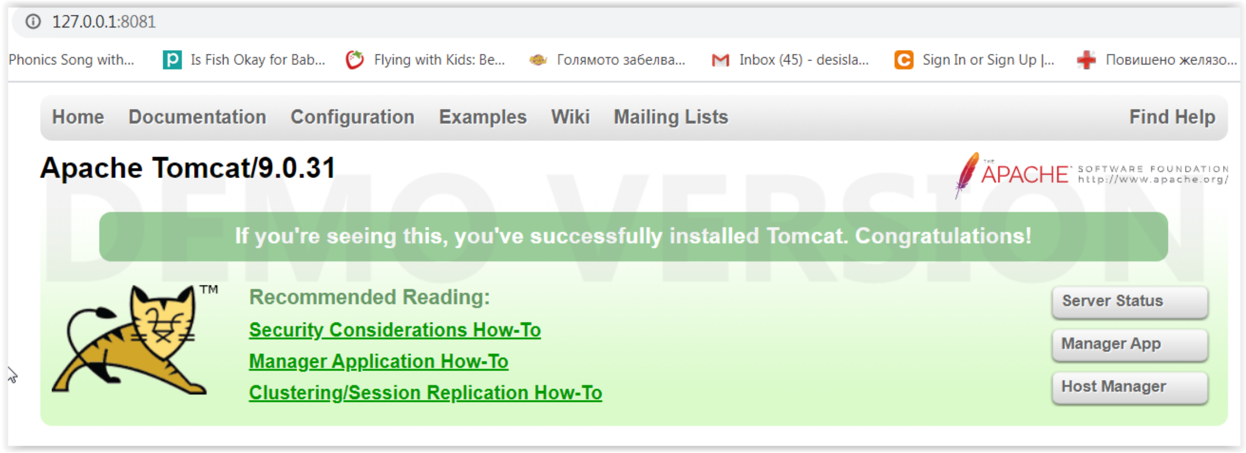

Ubuntu-server 18.04.4 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

```
dihdim@dihdim:~$ ls
apache-tomcat-9.0.31  apache-tomcat-9.0.31.tar.gz
dihdim@dihdim:~$
```



```
dihdim@dihdim:~$ sudo mysql
[sudo] password for dihdim:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.29-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

We have a ready to use Tomcat application server installed and successfully running when it is open in the browser.


127.0.0.1:8081

Phonics Song with... | Is Fish Okay for Bab... | Flying with Kids: Be... | Голямото забелва... | Inbox (45) - desisla... | Sign In or Sign Up |... | Повишено желязо...

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.31

APACHE SOFTWARE FOUNDATION http://www.apache.org/

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:

Security Considerations How-To

Manager Application How-To

Clustering/Session Replication How-To

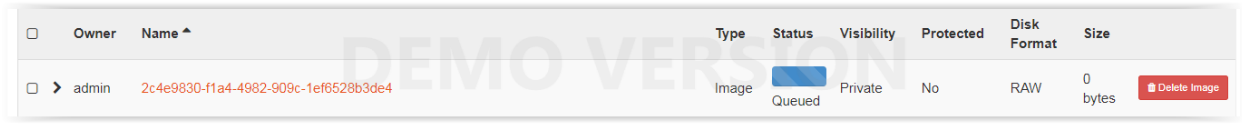
Server Status

Manager App

Host Manager

First Approach - Complications

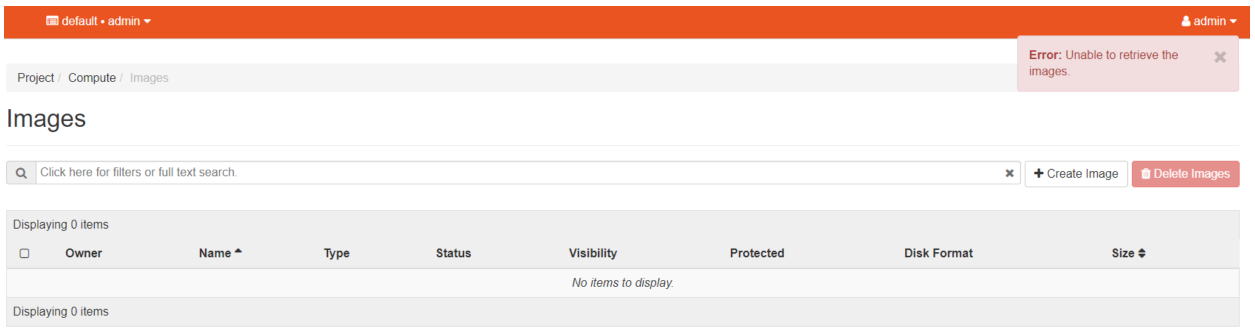
The problem we encountered was to create an image in OpenStack for one of our machines. State “Queued” lasts forever.



☐	Owner	Name ▲	Type	Status	Visibility	Protected	Disk Format	Size	
☐ ›	admin	2c4e9830-f1a4-4982-909c-1ef6528b3de4	Image	Queued	Private	No	RAW	0 bytes	Delete Image

First Approach - Conclusion

Even though we were successfully able to install the webserver and the SQL database on our local computers, OpenStack ended up not being able to upload an image successfully. It gave an extremely long Queued time that eventually prompted an error explaining an image cannot be retrieved. Multiple attempts were made to add a different type of image with different settings, but none were successful through OpenStack.


default • admin ▾ admin ▾

Error: Unable to retrieve the images.

Project / Compute / Images

Images

Click here for filters or full text search.

Create Image Delete Images

Displaying 0 items

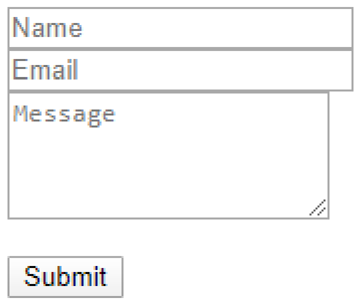
☐	Owner	Name ▲	Type	Status	Visibility	Protected	Disk Format	Size ⬍
				No items to display.				

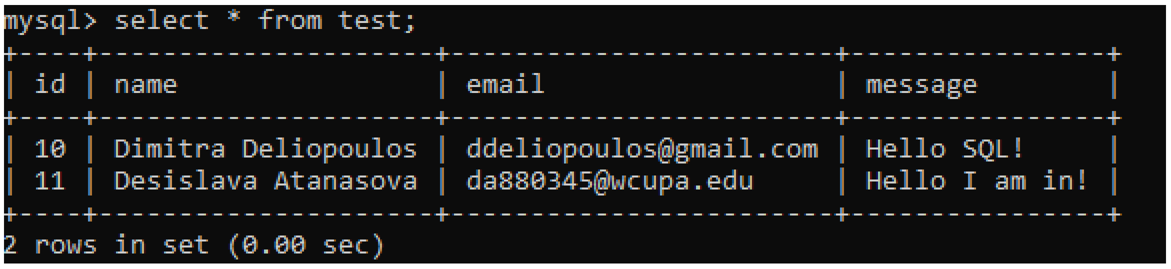
Displaying 0 items

Second Approach - Project Achieved/Goals

On a different week, OpenStack seemed to work and was able to upload images/instances. The problem from the first approach seemed to be the OpenStack/CloudLab service itself. After some time working with the second approach, the achieved goal was an accessible SQL form that can be filled out and submitted. This data was held in the backend SQL server that was already implemented.

IP address: http://128.105.146.179/ (May change due to floating IP issues involving CloudLab)

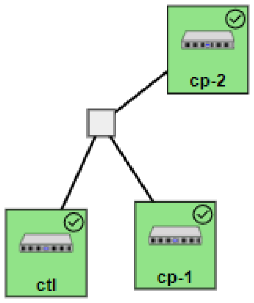




```
mysql> select * from test;
+----+---------------------+-----------------------+----------------+
| id | name                | email                 | message        |
+----+---------------------+-----------------------+----------------+
| 10 | Dimitra Deliopoulos | ddeliopoulos@gmail.com | Hello SQL!     |
| 11 | Desislava Atanasova | da880345@wcupa.edu    | Hello I am in! |
+----+---------------------+-----------------------+----------------+
2 rows in set (0.00 sec)
```

CloudLab - Profile

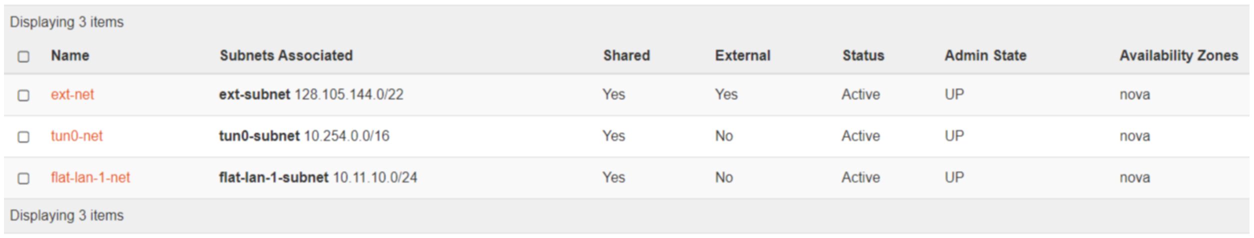
The profile provides a highly configurable OpenStack instance with a controller and one or more compute nodes (potentially at multiple Cloudlab sites) (and optionally a network manager node, in a split configuration). This profile runs x86, arm64, and POWER8 (Queens and up) nodes. It sets up OpenStack Stein (Ubuntu 18.04, python3), Rocky, Queens (Ubuntu 18.04, python2), Pike, Ocata, Newton, or Mitaka (Ubuntu 16.04, python2) (Liberty on 15.10, Kilo on 15.04, and Juno on 14.10, python2, deprecated) according to your choice, and configures all OpenStack services, pulls in some VM disk images, and creates basic networks accessible via floating IPs.



OpenStack - Network Configuration

Created allocation pools in the subnet ext-net that include:

[128.105.146.176], [128.105.146.177], [128.105.146.178], [128.105.146.179]



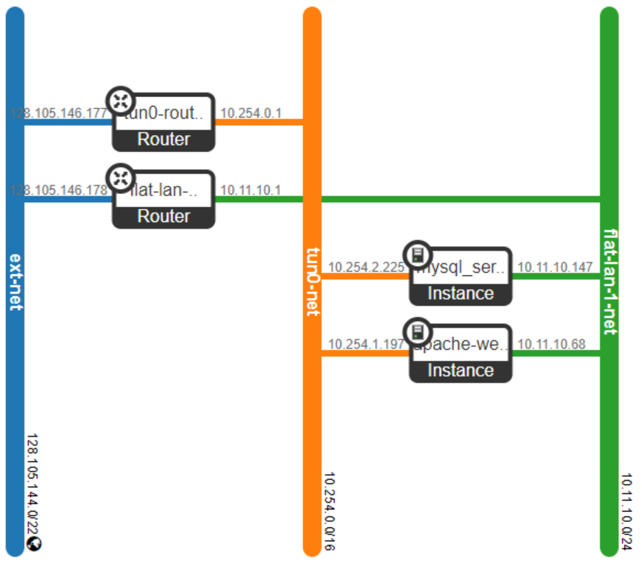
Displaying 3 items

Name	Subnets Associated	Shared	External	Status	Admin State	Availability Zones
ext-net	**ext-subnet** 128.105.144.0/22	Yes	Yes	Active	UP	nova
tun0-net	**tun0-subnet** 10.254.0.0/16	Yes	No	Active	UP	nova
flat-lan-1-net	**flat-lan-1-subnet** 10.11.10.0/24	Yes	No	Active	UP	nova

Displaying 3 items

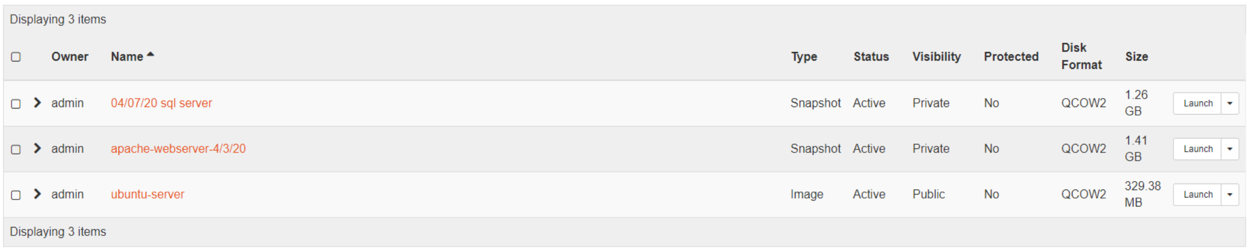
The network includes two routers, one router connects to the *tun0-net* network and the other connects to the *flat-lan-1-net* network, and both routers are connected to the *ext-net* network. The instances mysql_server and the apache-webserver have interfaces on both *tun0-net* and *flat-lan-1-net* networks.

Topology of network connections:



OpenStack – Images

Created an image using, “bionic-server-cloudimg-amd64”, named “ubuntu-server”, which has a disk format of QCOW2 and is just a standard ubuntu-server 18.04 used specifically for cloud purposes. Launched both instances using that image, then created a few snapshots of the same image after installing the essential components to the servers.



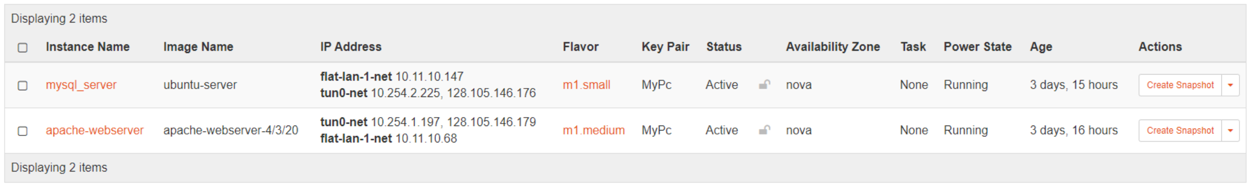
Displaying 3 items

	Owner	Name	Type	Status	Visibility	Protected	Disk Format	Size	
▢	admin	04/07/20 sql server	Snapshot	Active	Private	No	QCOW2	1.26 GB	Launch
▢	admin	apache-webserver-4/3/20	Snapshot	Active	Private	No	QCOW2	1.41 GB	Launch
▢	admin	ubuntu-server	Image	Active	Public	No	QCOW2	329.38 MB	Launch

Displaying 3 items

OpenStack – Instances

Created the instances and went through the OpenStack options to create its infrastructure which in our case specifically include: Details, source, flavor, networks, security groups, and key pair. The details were providing a name for the instance and a description (mysql_server, apache-webserever). The source includes the image using (ubuntu-server). The flavor for the apache-webserver was m1.medium and m1.small for the mysql_server. These allocate specific resources to the server. The networks connected to the instances include the *flat-lan-1-net* and *tun0-net.* A key pair was added to be able to connect to the servers remotely from the local computer.

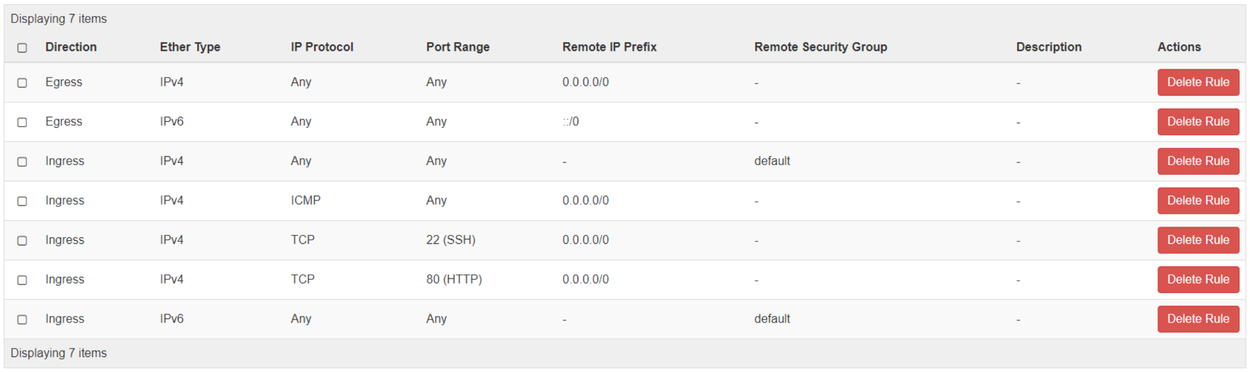


Displaying 2 items

	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
▢	mysql_server	ubuntu-server	**flat-lan-1-net** 10.11.10.147 **tun0-net** 10.254.2.225, 128.105.146.176	m1.small	MyPc	Active	nova	None	Running	3 days, 15 hours	Create Snapshot
▢	apache-webserver	apache-webserver-4/3/20	**tun0-net** 10.254.1.197, 128.105.146.179 **flat-lan-1-net** 10.11.10.68	m1.medium	MyPc	Active	nova	None	Running	3 days, 16 hours	Create Snapshot

Displaying 2 items

As for the security groups, I kept most the default rules, but added the TCP port 80 to be able to access the HTTP pages and be able to view the webserver created.

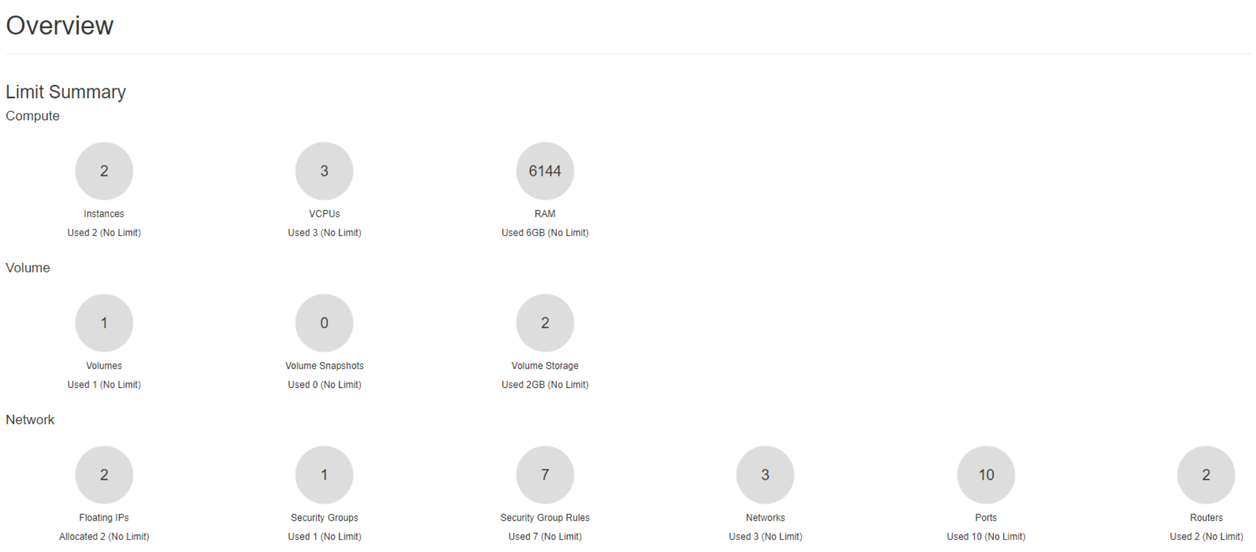


Displaying 7 items

	Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description	Actions
▢	Egress	IPv4	Any	Any	0.0.0.0/0	-	-	Delete Rule
▢	Egress	IPv6	Any	Any	::/0	-	-	Delete Rule
▢	Ingress	IPv4	Any	Any	-	default	-	Delete Rule
▢	Ingress	IPv4	ICMP	Any	0.0.0.0/0	-	-	Delete Rule
▢	Ingress	IPv4	TCP	22 (SSH)	0.0.0.0/0	-	-	Delete Rule
▢	Ingress	IPv4	TCP	80 (HTTP)	0.0.0.0/0	-	-	Delete Rule
▢	Ingress	IPv6	Any	Any	-	default	-	Delete Rule

Displaying 7 items

OpenStack – Overview



Overview

Limit Summary

Compute

2	3	6144
Instances	VCPUs	RAM
Used 2 (No Limit)	Used 3 (No Limit)	Used 6GB (No Limit)

Volume

1	0	2
Volumes	Volume Snapshots	Volume Storage
Used 1 (No Limit)	Used 0 (No Limit)	Used 2GB (No Limit)

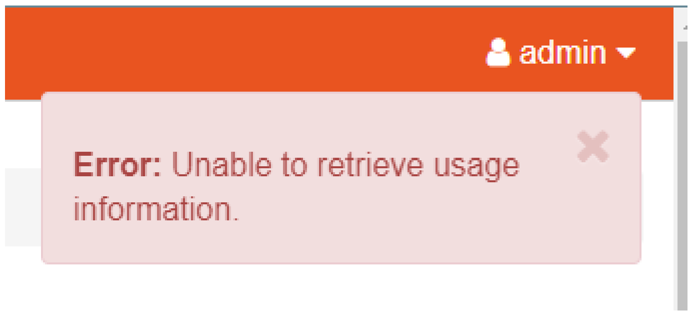
Network

2	1	7	3	10	2
Floating IPs	Security Groups	Security Group Rules	Networks	Ports	Routers
Allocated 2 (No Limit)	Used 1 (No Limit)	Used 7 (No Limit)	Used 3 (No Limit)	Used 10 (No Limit)	Used 2 (No Limit)

Second Approach Problems

The goal to add a better UI for the SQL database and login service did not work out as planned because the free PHP scripts used did not comply with the webserver used. Attempts were made by changing the backend webserver settings to read the PHP scripts uploaded but ended up doing more harm to the original project. A snapshot of the original project was preserved and eventually became the final project.

Also, after about three weeks of having the CloudLab/OpenStack project up, access to the OpenStack dashboard came with complications. Every login would show an error, stating it was unable to retrieve usage information. The floating IP's were still up and running though and access to the project was still available. We were just unable to see our images/instances in the dashboard.



Error: Unable to retrieve usage information.

Conclusion

In conclusion, the project was successfully implemented and works well. Installation of all the necessary components on the servers were finished and adjusted to fit the project idea. There were plenty of complications and different approaches along the way, but we succeeded in our goal of implementing the webserver and SQL server and having them connect to the given network together.

References

https://docs.openstack.org/train/
https://dev.mysql.com/doc/refman/5.6/en/drop-user.html
https://dev.to/alwaysup/create-a-web-server-and-save-form-data-into-mysql-database-using-php-beginners-guide-fah
https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-18-04
https://support.rackspace.com/how-to/mysql-connect-to-your-database-remotely/
https://www.configserverfirewall.com/ubuntu-linux/enable-mysql-remote-access-ubuntu/
https://code-boxx.com/php-mysql-search/
https://tomcat.apache.org/download-90.cgi